

**PROJECT NAME: GREENHOUSE AUTOMATION SYSTEM**

**GROUP NAME: NEXUS BRIGADE**

**GROUP MEMBERS**

|  |  |  |
| --- | --- | --- |
|  | **Reg No** | **Name** |
| **1** | **COM/B/01-00124/2022** | **Sabastian Chanzu** |
| **2** | **COM/B/01-00092/2022** | **Victor Wanyungu** |
| **3** | **COM/B/01-00086/2022** | **Byrone Kingsly** |
| **4** | **COM/B/01-00113/2022** | **Antony Kariuki** |
| **5** | **COM/B/01-00095/2022** | **Faith Munyithya** |

**Supervisors:**

Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date Submitted:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date Approved:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Declaration**

We, the undersigned, declare that this project proposal is our original work and has not been submitted for any academic award in any institution. This proposal has been prepared and conducted under the guidance and supervision of our project supervisor.

**Students’ Signatures:**

1. ANTONY KARIUKI: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. BYRONE KINGSLY: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. VICTOR WANYUNGU: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. FAITH MUNYITHYA: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. SEBASTIAN CHANZU: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Supervisors’ Signatures:**

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Table of Contents**

Declaration ii

Abstract 1

Chapter One: Introduction 2

1.1 Background 2

1.2 Statement of the Problem 2

1.3 Objectives 2

Specific objectives: 2

1.4 Justification 2

Chapter Two: Literature Review 3

2.1 Introduction 3

2.2 Overview of HappyNest 3

2.3 Challenges of existing caregiving platforms 3

2.5 Identified Gaps in existing platforms 3

2.6 Justification for the proposed mobile application 3

Chapter Three: Methodology 4

3.1 Research Design 4

3.2 Data Collection Method 4

3.3 System Development Approach 4

3.4 Phases of development 4

3.5 Tools and Technologies 5

Requirements Analysis and Specifications 6

4.1 Functional Requirements 6

4.2 Non-Functional Requirements 6

SYSTEM DESIGN 7

5.1 Conceptual Design 7

5.1.1 Key System Components 8

5.2 Functional Design 8

5.2.1 System Features 8

5.2.2 System Architecture 8

5.3 Database Design 8

5.3.1 Table and Relationships 8

5.4 wireframes 8

Development Environment 9

6.1 Hardware Requirements 9

6.2 Software Requirements 9

Development Schedule and budget 10

7.1 Gantt Chart 10

7.2 Development Budget 10

References 11

**ABSTRACT**

An IoT-enabled greenhouse automation system is proposed and evaluated to improve agricultural efficiency and sustainability. The system leverages sensor data and cloud-based analytics to automate environmental control, resulting in optimized plant growth conditions. By minimizing manual intervention and providing real-time insights, the system reduces water and energy waste, enhances crop yield, and empowers growers with remote management capabilities. This paper details the system's architecture, implementation, and performance evaluation, demonstrating the potential of IoT for modern greenhouse management.

By integrating a network of sensors, including temperature, humidity, light intensity, and soil moisture sensors, the system continuously collects environmental data. This data is processed by a microcontroller and transmitted to a cloud platform for storage and visualization. Automated control algorithms are implemented to adjust actuators, such as water pumps, fans, and lighting systems, based on predefined thresholds and real-time data analysis. The system aims to enhance crop yield, reduce resource consumption, and provide remote monitoring capabilities for greenhouse operators.

**INTRODUCTION**

**Proposal for an IoT-Based Greenhouse Automation System for Small-Scale Farmers**

**Background Information**

Smallholder farming plays a crucial role in global food production, contributing approximately

70% of the world's food supply while utilizing less than 25% of all agricultural land.

However, smallholder farmers, particularly in Kenya, face significant challenges that hinder

productivity and sustainability.

Pests, diseases, high operational costs, lack of technical knowledge, and climate variability

continue to affect yields and economic viability.

Greenhouse farming has emerged as a solution to mitigate some of these challenges by

providing controlled environments for crop production.

Despite its success in large-scale horticulture, smallholder adoption has been slow, with failure

rates between 30% and 70% due to inadequate knowledge, high setup costs, water scarcity, and

poor market access.

**Problem statement**

As per the research done by Kenya Agricultural and livestock Research Organization in Kisii,

some of the challenges that lead to inefficiencies and economic losses include;

**1. Environmental Monitoring Issues**

● Difficulty maintaining optimal temperature and humidity levels

● Limited ability to monitor soil moisture consistently

● Challenges in tracking greenhouse conditions remotely

**2. Crop Health Challenges**

● Early disease detection difficulties

● Limited expertise in pest identification

● Delayed response to crop stress

**3. Resource Management Problems**

● Inefficient water usage patterns

● Suboptimal fertilizer application

● Limited technical support (1:1093 extension officer-to-farmer ratio)

**Project Objectives**

The proposed IoT-based greenhouse automation system aims to:

● Automate greenhouse monitoring and control processes to improve productivity.

● Reduce operational costs by optimizing water and input usage.

● Provide real-time data and alerts to farmers for proactive decision-making.

● Enhance knowledge-sharing through a digital platform integrated with agricultural

extension services.

● Improve market access by linking farmers directly to buyers and cooperatives.

● Develop an AI-powered image recognition system that allows farmers to upload

crop images via for instant disease diagnosis and tailored treatment

recommendations.

● Link farmers with nearby agro-vets for personalized crop treatment solutions

based on real-time disease diagnostics.

**CHAPTER TWO: LITERATURE REVIEW**

**2.1 Introduction**

This chapter reviews existing greenhouse farming solutions, their advantages, limitations, and the gap this project aims to fill. The review explores different technological approaches temperature and moisture control, their effectiveness, and the challenges faced in providing comprehensive and accessible support for a farmers.

**2. IoT in Agriculture and Precision Farming:**

* **Overview of IoT Applications:**
  + Discuss the general applications of IoT in agriculture, including monitoring, control, and data analysis.
  + Explore the concept of precision agriculture and how IoT enables it.
* **Benefits of IoT in Agriculture:**
  + Discuss the potential benefits, such as increased crop yield, reduced resource consumption, and improved efficiency.
  + Provide examples of successful IoT implementations in agriculture.
* **Challenges of IoT in Agriculture:**
  + Address the challenges, such as connectivity issues, data security, and cost.
  + Discuss the problems of power consumption in remote areas.

**3. Greenhouse Automation Systems:**

* **Traditional Greenhouse Control:**
  + Describe traditional methods of greenhouse control and their limitations.
* **Automated Greenhouse Systems:**
  + Review existing automated greenhouse systems, focusing on their control strategies, sensor networks, and actuator systems.
  + Discuss the use of microcontrollers and single-board computers in greenhouse automation.
* **Wireless Sensor Networks (WSNs) in Greenhouses:**
  + Explore the use of WSNs for environmental monitoring in greenhouses.
  + Discuss different wireless communication protocols .
* **Cloud-Based Greenhouse Monitoring and Control:**
  + Review the use of cloud platforms for data storage, analysis, and remote control of greenhouse systems.
  + Discuss the benefits of cloud computing in terms of scalability and accessibility.

**4. Sensor Technologies and Actuators:**

* **Environmental Sensors:**
  + Review different types of sensors used for monitoring temperature, humidity, soil moisture, light intensity, CO2, and pH.
  + Discuss the accuracy, reliability, and cost of different sensor technologies.
* **Actuator Systems:**
  + Review different types of actuators used for irrigation, ventilation, lighting, and heating/cooling.
  + Discuss the control methods and energy efficiency of different actuator systems.

**5. Data Analysis and Machine Learning:**

* **Data Analysis Techniques:**
  + Explore data analysis techniques used for processing and interpreting sensor data.
  + Discuss the use of statistical analysis and data visualization tools.
* **Machine Learning Applications:**
  + Review the application of machine learning algorithms for predicting crop yield, optimizing environmental conditions, and detecting anomalies.
  + Discuss the use of AI to create predictive models for disease and pest control.

**6. User Interface and User Experience (UI/UX):**

* **Web-Based Dashboards:**
  + Review the design and functionality of web-based dashboards for greenhouse monitoring and control.
* **Mobile Applications:**
  + Explore the development of mobile applications for remote access and control.
  + Discuss the importance of user-friendly interfaces and intuitive navigation.

**7. Security and Privacy:**

* **IoT Security Challenges:**
  + Discuss the security challenges associated with IoT devices and networks.
  + Review security measures for protecting sensor data and preventing unauthorized access.
* **Data Privacy:**
  + Address the importance of data privacy and compliance with relevant regulations.

**8. Gaps in the Literature and Research Opportunities:**

* **Identify Gaps:**
  + Identify gaps in the existing research and areas where further investigation is needed.
  + Point out any limitations of existing systems or approaches.
* **Research Opportunities:**
  + Suggest potential research directions and opportunities for innovation.
  + Discuss how your proposed system addresses the identified gaps.

**9. Conclusion:**

* Summarize the key findings of the literature review.
* Reinforce the significance of your project and its potential contributions.
* Transition to the next chapter, which outlines your proposed system design and methodology.

**Key Sources to Consider:**

* Academic journals (e.g., Computers and Electronics in Agriculture, Precision Agriculture).
* Conference proceedings (e.g., IEEE conferences on IoT, agricultural technology).
* Research papers and technical reports.
* Industry publications and reports.
* Online databases (e.g., IEEE Xplore, ScienceDirect, Google Scholar).

**Potential benefits for project justification.**

1. Increased greenhouse productivity due to optimized environmental control.
2. Reduction in crop losses from pests and diseases through early detection.
3. Lower water and input costs through automation and precise monitoring.
4. Enhanced access to technical knowledge through digital learning tools.
5. Improved income for smallholder farmers through better market connections.

**CHAPTER THREE: METHODOLOGY**

This chapter outlines the methodology that will be used to develop the  **Greenhouse Automation** application. It covers the research design, system development approach, data collection methods, and tools to be used throughout the development process.

3.1 Research Design

A robust methodology for developing an IoT-based greenhouse automation system ensures a structured and efficient process. Here's a comprehensive methodology breakdown:

**1. Requirements Gathering and Analysis:**

* **Stakeholder Identification:** Identify all stakeholders (e.g., greenhouse operators, farmers, researchers).
* **Needs Assessment:** Conduct interviews, surveys, and workshops to understand their specific needs and pain points.
* **Functional Requirements:** Define the functionalities the system must perform (e.g., automated irrigation, remote monitoring, data logging).
* **Non-Functional Requirements:** Define quality attributes like performance, security, reliability, and usability.
* **Use Case Development:** Create use cases to describe how users will interact with the system.

**2. System Design:**

* **Architecture Design:**
  + Define the overall system architecture, including hardware and software components.
  + Choose appropriate communication protocols (e.g., MQTT, HTTP).
  + Select cloud platform and database technologies.
* **Hardware Design:**
  + Choose sensors and actuators based on accuracy, range, and cost.
  + Design the sensor network and actuator placement.
  + Select microcontrollers/SBCs based on processing power and connectivity.
  + Create circuit diagrams and PCB layouts.
* **Software Design:**
  + Design the backend API (Node.js).
  + Design the frontend UI/UX (React and Flutter).
  + Develop control algorithms for automated actions.
  + Plan the database schema.
* **Security Design:**
  + Implement security measures at each layer (device, network, cloud).
  + Use encryption and authentication protocols.

**3. Development and Prototyping:**

* **Hardware Prototyping:**
  + Build a prototype of the sensor network and actuator system.
  + Test sensor accuracy and actuator functionality.
* **Software Development:**
  + Develop the backend API using Node.js and Express.js.
  + Develop the web-based dashboard using React.
  + Develop the mobile app using Flutter.
  + Implement data storage and retrieval.
  + Integrate MQTT communication.
* **Iterative Development:**
  + Use an agile development approach with short sprints.
  + Conduct regular code reviews and testing.

**4. Testing and Validation:**

* **Unit Testing:** Test individual software components.
* **Integration Testing:** Test the interaction between different components.
* **System Testing:** Test the entire system in a simulated or real greenhouse environment.
* **User Acceptance Testing (UAT):** Involve stakeholders in testing the system to ensure it meets their requirements.
* **Performance Testing:** Evaluate the system's performance under various loads.
* **Security Testing:** Conduct vulnerability assessments and penetration testing.
* **Environmental testing:** Test the system under various environmental conditions that could be expected in the greenhouse.

**5. Deployment and Implementation:**

* **Pilot Deployment:** Deploy the system in a small-scale greenhouse environment.
* **Data Migration:** Migrate existing data to the new system.
* **User Training:** Provide training to greenhouse operators on how to use the system.
* **Full-Scale Deployment:** Deploy the system in all greenhouses.

**6. Monitoring and Maintenance:**

* **Continuous Monitoring:** Monitor system performance and identify potential issues.
* **Data Analysis:** Analyze sensor data to optimize greenhouse conditions.
* **Regular Maintenance:** Perform software updates and hardware maintenance.
* **Feedback Collection:** Collect feedback from users and incorporate it into future updates.
* **System Optimization:** Continuously improve the system's performance and efficiency.

**7. Documentation:**

* **System Documentation:** Create detailed documentation of the system architecture, design, and implementation.
* **User Manual:** Provide a user-friendly manual for greenhouse operators.
* **API Documentation:** Document the backend API for developers.
* **Maintenance Manual:** Create a manual for system maintenance and troubleshooting.

**Key Methodological Principles:**

* **User-Centered Design:** Focus on the needs and requirements of the end-users.
* **Data-Driven Approach:** Use sensor data and analytics to optimize greenhouse conditions.
* **Iterative Development:** Continuously improve the system based on feedback and data.
* **Collaboration:** Foster collaboration between developers, greenhouse operators, and researchers.
* **Modular Design:** Design the system with modular components for easy maintenance and updates.

3.2. System Development Tools

**Frontend**:

* **React Native (Expo)** for building the mobile and web interfaces.
* **CSS** for styling components.

**Backend**:

* for handling data management, authentication, and API integrations.

**Database**:

* **Firebase** for storing user data securely.

**APIs**:

* Integration with wearable devices for real-time health data.
* **Firebase** for real-time notifications and alerts.

**CHAPTER 4: REQUIREMENTS ANALYSIS AND SPECIFICATIONS**

**Functional Requirements**

* **4.3.1 Sensor Data Acquisition:**
  + **Temperature Monitoring:**
    - Requirement: The system shall monitor ambient air temperature within the greenhouse.
    - Specification: Range: -10°C to 50°C, Accuracy: ±0.5°C, Resolution: 0.1°C, Sampling Rate: 1 sample per minute.
    - Requirement: The system shall log temperature data with a timestamp.
  + **Humidity Monitoring:**
    - Requirement: The system shall monitor relative humidity within the greenhouse.
    - Specification: Range: 0% to 100% RH, Accuracy: ±3% RH, Resolution: 1% RH, Sampling Rate: 1 sample per minute.
    - Requirement: The system shall log humidity data with a timestamp.
  + **Soil Moisture Monitoring:**
    - Requirement: The System shall monitor soil moisture levels.
    - Specification: measurement range 0-100% volumetric water content, accuracy +/- 5%, sampling rate 5 minutes.
    - Requirement: The system shall log soil moisture data with a timestamp.
  + **Light Intensity Monitoring:**
    - Requirement: The System shall monitor light intensity.
    - Specification: measurement range 0-1000 lux, accuracy +/- 10 lux, sampling rate 1 minute.
    - Requirement: The system shall log light intensity data with a timestamp.
  + **CO2 Monitoring:**
    - Requirement: The System shall monitor CO2 levels.
    - Specification: measurement range 0-2000 ppm, accuracy +/- 50 ppm, sampling rate 5 minutes.
    - Requirement: The system shall log CO2 data with a timestamp.
  + **pH Monitoring:**
    - Requirement: The System shall monitor the pH of the soil.
    - Specification: measurement range 4-9, accuracy +/- 0.2, sampling rate 1 hour.
    - Requirement: The system shall log pH data with a timestamp.
* **4.3.2 Actuator Control:**
  + **Automated Irrigation:**
    - Requirement: The system shall automatically activate the irrigation system when soil moisture falls below a user-defined threshold.
    - Specification: Thresholds adjustable through the web/mobile interface.
    - Requirement: The system shall allow for scheduled irrigation cycles.
    - Requirement: The system shall allow manual override of automated irrigation.
  + **Automated Ventilation:**
    - Requirement: The system shall automatically activate ventilation fans when temperature or humidity exceeds user-defined thresholds.
    - Specification: Thresholds adjustable through the web/mobile interface.
    - Requirement: The system shall allow manual override of automated ventilation.
  + **Automated Heating/Cooling:**
    - Requirement: The system shall automatically activate heating or cooling systems based on temperature thresholds.
    - Specification: Thresholds adjustable through the web/mobile interface.
    - Requirement: The system shall allow manual override of automated heating/cooling.
* **4.3.3 Data Processing and Storage:**
  + Requirement: The system shall filter sensor data to remove noise.
  + Requirement: The system shall store sensor data in a cloud-based database (e.g., MongoDB).
  + Requirement: The system shall retain sensor data for a minimum of one year.
  + Requirement: The system shall perform daily backups of the database.
* **4.3.4 Remote Monitoring and Control:**
  + Requirement: The system shall provide a web-based dashboard and mobile app for remote monitoring.
  + Requirement: The web/mobile interface shall display real-time sensor data in graphical and tabular formats.
  + Requirement: The web/mobile interface shall allow remote control of all actuators.
  + Requirement: The web/mobile interface shall display historical sensor data.
* **4.3.5 Alert and Notification System:**
  + Requirement: The system shall generate alerts for critical conditions (e.g., high/low temperature, low soil moisture, equipment malfunction).
  + Requirement: Alerts shall be delivered via email, SMS, and/or push notifications.
  + Requirement: Users shall be able to configure alert thresholds.
* **4.3.6 Reporting and Analytics:**
  + Requirement: The system shall generate daily, weekly, and monthly reports on environmental conditions.
  + Requirement: The system shall provide basic crop yield analysis based on environmental data.
  + Requirement: The system shall allow users to export data in CSV format.
* **4.3.7 User Management:**
  + Requirement: The system shall implement role-based access control (admin, operator, viewer).
  + Requirement: The system shall use secure user authentication (username/password, two-factor authentication).

**4.4 Non-Functional Requirements**

* **4.4.1 Performance:**
  + Requirement: The system shall have a maximum response time of 2 seconds for actuator control.
  + Requirement: The system shall update sensor data on the web/mobile interface at least once per minute.
  + Requirement: The system shall be able to handle at least 10 concurrent users.
* **4.4.2 Reliability:**
  + Requirement: The system shall have an uptime of 99%.
  + Requirement: Sensor and actuator components shall have a minimum lifespan of 2 years.
  + Requirement: The system shall have a failover mechanism in case of internet connectivity loss.
  + Requirement: The system shall have an ability to log errors.
* **4.4.3 Security:**
  + Requirement: The system shall use encryption for data transmission and storage.
  + Requirement: The system shall implement secure user authentication and authorization.
  + Requirement: The system shall undergo regular security audits.
  + Requirement: The system shall follow the principle of least privilege.
* **4.4.4 Usability:**
  + Requirement: The web/mobile interface shall be intuitive and user-friendly.
  + Requirement: The system shall provide clear and concise data visualization.
  + Requirement: User training documentation shall be provided.
* **4.4.5 Scalability:**
  + Requirement: The system shall be able to accommodate the addition of more sensors and actuators.
  + Requirement: The system shall be designed using a scalable cloud-based architecture.
* **4.4.6 Maintainability:**
  + Requirement: The system shall be designed with modular components for easy maintenance.
  + Requirement: The system shall have comprehensive documentation.
  + Requirement: The system shall provide remote diagnostics capabilities.
* **4.4.7 Environmental Requirements:**
  + Requirement: Sensors and actuators shall operate reliably within the greenhouse's temperature and humidity ranges.
  + Requirement: The system shall be designed for energy efficiency.
  + Requirement: Devices exposed to moisture or dust shall have appropriate IP ratings.

**CHAPTER 5: SYSTEM DESIGN**

**5.1 Conceptual design**

The conceptual design for system outlines the overall structure and interaction between the system’s components ensuring seamless greenhouse operation. It incorporates the following key elements:

**1. Sensor Network:**

* **Environmental Sensors:**
  + Temperature and humidity sensors: To monitor air conditions.
  + Soil moisture sensors: To track soil hydration levels.
  + CO2 sensors: To monitor carbon dioxide levels.
  + pH sensors: To assess soil acidity/alkalinity.
* **Sensor Placement:**
  + Strategically position sensors throughout the greenhouse to obtain representative data.
  + Consider zoning the greenhouse for more granular control.
* **Sensor Communication:**
  + Wireless sensor networks (WSNs) using protocols
  + Wired connections may be preferred for reliability in certain applications.

**2. Actuator System:**

* **Irrigation System:**
  + Automated water pumps and solenoid valves for precise irrigation.
  + Drip irrigation systems for efficient water usage.
* **Ventilation System:**
  + Automated fans and vents to regulate temperature and humidity.
* **Heating/Cooling System:**
  + Heaters or coolers controlled by temperature sensors.

**3. Microcontroller/Edge Computing:**

* **Data Acquisition and Processing:**
  + Microcontrollers to collect sensor data and control actuators.
  + Edge computing capabilities for local data processing and real-time control.
* **Communication Interface:**
  + Wi-Fi, Ethernet, or cellular modules for internet connectivity.
* **Control Logic:**
  + Implement control algorithms to automate actuator operation based on sensor data.

**4. Cloud Platform:**

* **Data Storage:**
  + Cloud databases to store sensor data for analysis and historical records.
* **Data Visualization:**
  + Web-based dashboards or mobile apps to display real-time and historical data.
* **Remote Monitoring and Control:**
  + Enable remote access to the system for monitoring and adjusting settings.
* **Data Analytics:**
  + Implement algorithms to analyze sensor data and optimize greenhouse conditions.
  + Machine learning can be used to improve crop yields.
* **Alerts and Notifications:**
  + System to send alerts to the user when certain tresholds are crossed.

**5. Communication Network:**

* **Internet Connectivity:**
  + Reliable internet connection for data transmission to the cloud.
* **Network Security:**
  + Implement security measures to protect the system from unauthorized access.

**Key Design Considerations:**

* **Scalability:** The system should be designed to accommodate future expansion.
* **Reliability:** Redundant components and robust software design are essential.
* **Energy Efficiency:** Minimize energy consumption through intelligent control algorithms.
* **Cost-Effectiveness:** Balance system functionality with cost considerations.
* **User-Friendliness:** Provide an intuitive interface for greenhouse operators.

**CHAPTER 6: Development Environments**

**6.1 Hardware requirements**

To ensure the successful development and deployment of the system, the following hardware requirements are needed:

**Core Technologies and Their Roles:**

* **Node.js:**
  + This will primarily handle the backend logic. It will:
    - Receive sensor data from IoT devices.
    - Process and store data in a database.
    - Implement the application's business logic.
    - Provide APIs for the React web application and the Flutter mobile app.
    - Manage communication with IoT devices using protocols like MQTT.
* **React:**
  + This will be used to build the web-based dashboard for monitoring and controlling the greenhouse. It will:
    - Visualize sensor data in real-time.
    - Provide a user interface for adjusting settings and configurations.
    - Display historical data and analytics.
* **Flutter:**
  + This will be used to create the mobile application, allowing users to monitor and control the greenhouse from their smartphones. It will:
    - Provide a mobile-friendly interface for accessing sensor data.
    - Enable remote control of actuators.
    - Deliver notifications and alerts.

**2. Development Environment Setup:**

* **Node.js Environment:**
  + **Node.js and npm (Node Package Manager):** Install the latest LTS (Long Term Support) version of Node.js.
  + **IDE:** Visual Studio Code (VS Code) is highly recommended due to its versatility and excellent support for JavaScript and Node.js.
  + **Database:** Choose a database based on your needs (e.g., MongoDB, PostgreSQL, MySQL).
  + **Frameworks:** Express.js is a popular framework for building Node.js web applications and APIs.
  + **MQTT Broker:** Install and configure an MQTT broker (e.g., Mosquitto) for IoT device communication.
* **React Environment:**
  + **Node.js and npm:** React relies on Node.js.
  + **Create React App:** Use this tool to quickly set up a new React project.
  + **IDE:** VS Code is excellent for React development.
  + **Browser:** Chrome or Firefox for testing and debugging.
  + **React Developer Tools:** Install the browser extension for debugging React applications.
* **Flutter Environment:**
  + **Flutter SDK:** Download and install the Flutter SDK.
  + **Dart SDK:** Flutter uses the Dart programming language, which is included in the Flutter SDK.
  + **IDE:** VS Code with the Flutter and Dart extensions, or Android Studio with the Flutter and Dart plugins.
  + **Android SDK and/or Xcode:** Depending on whether you're developing for Android or iOS.
  + **Emulators/Simulators:** Set up Android emulators and/or iOS simulators for testing.
* **General Tools:**
  + **Git:** For version control.
  + **Postman/Insomnia:** For testing APIs.

**3. Workflow:**

* **Backend Development (Node.js):**
  + Develop APIs to handle sensor data and control actuators.
  + Implement database interactions.
  + Set up MQTT communication.
* **Web Development (React):**
  + Create components to display sensor data and control devices.
  + Use API calls to communicate with the Node.js backend.
  + Implement real-time data visualization.
* **Mobile Development (Flutter):**
  + Build the mobile app's UI using Flutter widgets.
  + Use API calls to communicate with the Node.js backend.
  + Implement features like push notifications for alerts.
* **IoT Integration:**
  + Configure IoT devices to send data to the MQTT broker.
  + Ensure the Node.js backend can receive and process the data.

**Key Considerations:**

* **API Design:** A well-designed API is crucial for seamless communication between the frontend (React and Flutter) and the backend (Node.js).
* **Real-time Data:** Consider using WebSockets or MQTT for real-time data updates.
* **Security:** Implement security measures to protect your system from unauthorized access.
* **Scalability:** Design your system to handle increasing amounts of data and users.

**Conclusion**

This IoT-based greenhouse automation system has the potential to revolutionize small-scale

farming in Kenya by improving productivity, reducing costs, and increasing food security.

By leveraging technology, we can empower smallholder farmers to overcome existing challenges

and create a more sustainable agricultural future.

Success Stories

○ Improved crop yields through better climate control

○ Reduced water consumption

○ Extended growing seasons

○ Increased market access

**CHAPTER 7: DEVELOPMENT SCHEDULE AND BUDGET**

**7.1 Development schedule**

**Schedule**

|  |  |
| --- | --- |
| **Task** | **Period** |
| **Project planning** |  |
| **Requirement analysis** |  |
| **System Design** |  |
| **Protptype Design** |  |
| **Front-end Development** |  |
| **Backend development** |  |
| **Object detection intergration** |  |
| **Temperature and humidity sensors intergration** |  |
| **Testing and Debugging** |  |
| **Deployment** |  |

**Budget**

| **Category** | **Quantity** | **Unit Cost (KES)** | **Total Cost (KES)** |
| --- | --- | --- | --- |
|  |  |  |  |
| **IoT Sensors & Hardware** | 100 sets | 5,500 | **550,000** |
| - Temperature & Humidity Sensors (DHT22) | 100 | 1,500 | 150,000 |
| - Soil Moisture Sensors | 100 | 1,200 | 120,000 |
| - Raspberry Pi (Controller) | 10 | 9,500 | 95,000 |
| - Camera Module for Pest Detection | 10 | 7,500 | 75,000 |
| - Relay Modules for Smart Irrigation | 100 | 1,000 | 100,000 |
| **Mobile App Development** | - | - | **400,000** |
| - Frontend (Flutter) | - | - | 200,000 |
| - Backend | - | - | 100,000 |
| - UI/UX Design | - | - | 50,000 |
| - Testing & Deployment | - | - | 50,000 |
| **AI Model Training for Pest Detection** | - | - | **250,000** |
| - Data Collection & Model Training | - | - | 150,000 |
| - Server Hosting (1 Year) | - | - | 100,000 |
| **Cloud & SMS Alert System** | - | - | **300,000** |
| - Twilio API (SMS Notifications) | - | - | 150,000 |
| - Firebase Cloud Storage | - | - | 100,000 |
| - WhatsApp Business API Setup | - | - | 50,000 |
| **Training & Farmer Outreach** | - | - | **250,000** |
| - Training Workshops for Farmers | 5 sessions | 30,000 | 150,000 |
| - Training Manuals & Support Materials | - | - | 50,000 |
| - Travel & Logistics for Outreach | - | - | 50,000 |
| **Operational & Miscellaneous Costs** | - | - | **200,000** |
| **Total Estimated Budget** | - | - | **1,950,000 KES** |