**PROG7311 POE PART 3**

**Imaan Ebrahim**

**ST10021922**



**Report: Enhancements for Agri-Energy Connect**

# 1. Optimizing Performance for the Prototype and Final Software

The success of the Agri-Energy Connect platform depends heavily on its performance. This part describes methods to enhance the prototype and final software, guaranteeing a system that is responsive, scalable, and efficient.

Current Prototype Optimizations:

The model uses the following enhancements:

* **Effective Management of Data:**  
  Uses in-memory storage to quickly retrieve and manipulate data while developing.
* **Asynchronous Programming:**

Programming that allows tasks to be executed independently of the main program flow, allowing for more efficient use of resources.

* **Improved MVC Design:**  
  Divides responsibilities into Model, View, and Controller to lower processing burden.

## Guidelines for Final Software Performance

To guarantee satisfactory performance on the end platform, development should be guided by the following principles:

1. **Backend Optimizations:**

* **Database Optimization:**

-Utilize indexes and optimization strategies to enhance retrieval speeds.  
-Utilize database partitioning to efficiently manage large datasets.  
-Utilize tools such as Redis to store commonly used queries in order to improve performance.

* **Scalable Infrastructure:**

-Use cloud platforms like Azure and AWS that have auto-scaling capabilities for managing increases in traffic.  
-Utilize containerization (for example, Docker) in order to deploy microservices that have the ability to scale separately.

1. **Frontend Performance:**

* **Static Asset Optimization**:

- Reduce load times by minifying CSS, JavaScript, and images.  
-Deploy CDNs to improve the delivery speed of static content.

* **Lazy Loading:**

- Load content that is seen on the screen first and delay loading less important resources.

1. **Network Efficiency:**

- Utilize compression such as Gzip to decrease the size of data exchanged between server and client.  
-Reduce the size of API payloads by utilizing efficient data serialization formats like JSON and Protobuf.

1. **Asynchronous Processing**

-Transfer time-consuming tasks (such as data analytics) to background workers through queues such as RabbitMQ or Azure Service Bus.  
-Utilize event-driven programming to receive real-time updates and decrease bottlenecks

1. **Performance Monitoring:**

Incorporate tools like Application Insights or New Relic for monitoring performance metrics, such as:

* Latency of requests.
* Utilization of resources such as CPU and memory.
* Times taken to query a database.

Consistently perform load testing with tools such as Apache JMeter to pinpoint bottlenecks

Benefits of these Guidelines:

* **Improved User Experience**:

Quicker reaction times guarantee that farmers and employees can engage with the platform promptly.

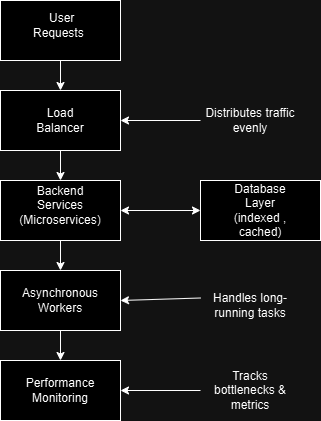
* **Scalability:**

As the community grows, the system can accommodate a growing number of users.

* **Reliability:**

Continuous monitoring of performance ensures early identification of problems, decreasing downtime.

Diagram showing what the final optimization of the software will look like:



Made using: <https://app.diagrams.net/>

# 2. Recommended Software Development Methodology

**Recommendation: Agile Methodology**

The Agile methodology is strongly advised for creating the Agri-Energy Connect platform. Agile is a contemporary and adaptable software development method that encourages continual progress, regular feedback, and flexibility to changing demands.

**Why Agile?**

The strengths of Agile methodology make it well-suited for the requirements and goals of the Agri-Energy Connect project.  
**Iterative Development**:

* Agile divides the project into smaller sprints to ensure that a functional version of the platform is delivered at the end of each iteration.

**Flexibility and Adaptability**:

* Agile offers flexibility to pivot without disrupting overall progress when faced with changing requirements, such as integrating more green energy technologies or expanding features for farmers.

**Stakeholder Involvement**:

* Agile focuses on fostering cooperation among developers, marketing teams, and stakeholders.

**Focus on Delivering Value**:

* Agile prioritizes features according to their business value, ensuring that essential functionalities like farmer and employee dashboards are delivered promptly, therefore decreasing time-to-market.

**Implementation Guidelines for Agile**

1. Framework: Scrum:

* The Agile project benefits from the structured sprints, defined roles, and regular reviews provided by the Scrum framework.
* Sprints consist of cycles lasting 2-3 weeks, with each cycle providing a new functional addition to the platform, such as role-based dashboards in Sprint 1 and product management in Sprint 2.

Roles:

* Product Owner: Makes sure the backlog is prioritized to meet business requirements for the platform.
* Scrum Master: Helps with the process and eliminates obstacles for the team.
* Development Team: Executes the characteristics and capabilities.

Mapping Agile to Client Needs

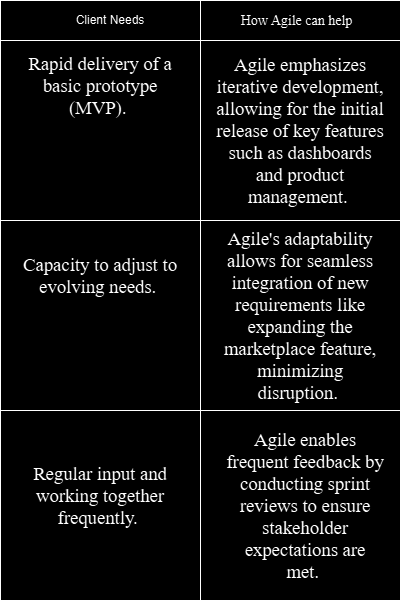
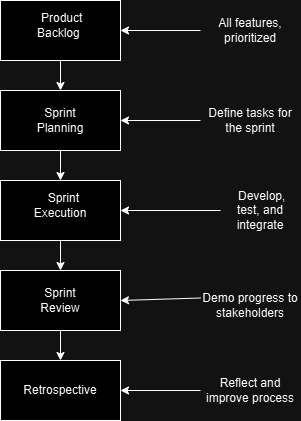


Diagram for Agile Workflow



# 3. DevOps Recommendation

**Recommendation: Implement DevOps Practices**

Strongly suggested for the Agri-Energy Connect platform is the implementation of DevOps to improve development and align with Agile methodology. DevOps stresses the importance of teamwork between development and operations departments to guarantee fast, trustworthy, and uniform software delivery.

**Why DevOps?**

* **Continuous Integration (CI)**:

Automatically incorporate and verify code modifications upon committing.  
Utilize GitHub Actions or Jenkins to streamline the CI pipeline through automation.

* **Continuous Delivery (CD)**:

Automatically roll out the most recent code updates to staging or production environments following successful testing.

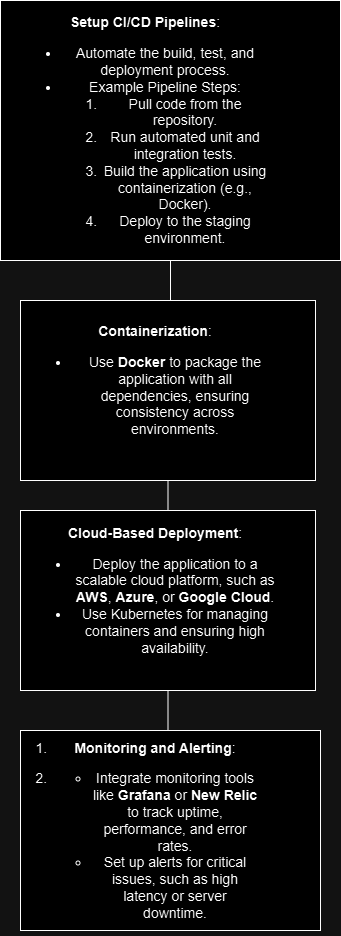
* **Infrastructure as Code (IaC)**:

Utilize Terraform or AWS CloudFormation tools for defining infrastructure setups as code, guaranteeing consistency and repeatability.

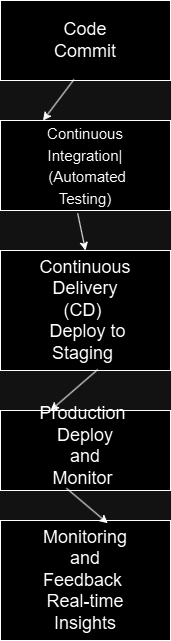
* **Monitoring and Logging:**

Utilize real-time monitoring tools like ELK Stack and Prometheus for tracking system performance.

Implementation plan diagram:



DevOps Workflow Diagram:



Using DevOps will make the development and deployment of the Agri-Energy Connect platform more efficient, resulting in quicker delivery, reliability, and scalability. Aligning with Agile methodology is crucial for reaching the project’s objectives.

# Recommended Frameworks: ITIL, Zachman, or TOGAF

The best choice for the Agri-Energy Connect platform would be a mix of ITIL and TOGAF.

**Why ITIL?**

ITIL, or IT Infrastructure Library, is a comprehensive framework for managing IT services. It includes strategies for aligning services with

* business needs
* designing and transitioning services
* operations support
* continual service improvement.

ITIL supports Agri-Energy Connect by

* designing services
* deploying them to production
* managing day-to-day operations
* continually improving platform performance and user experience.

**Why TOGAF?**

TOGAF is a comprehensive architecture framework by The Open Group for defining Agri-Energy Connect's

* business
* information
* application
* technology architecture

**ITIL supports Agri-Energy Connect by**

TOGAF's ADM provides a step-by-step process for

* developing enterprise architecture
* creating an architecture vision
* defining it and implementing it
* ACF organizes architectural elements such as data and technology standards.

**Combining TOGAF and ITIL:**

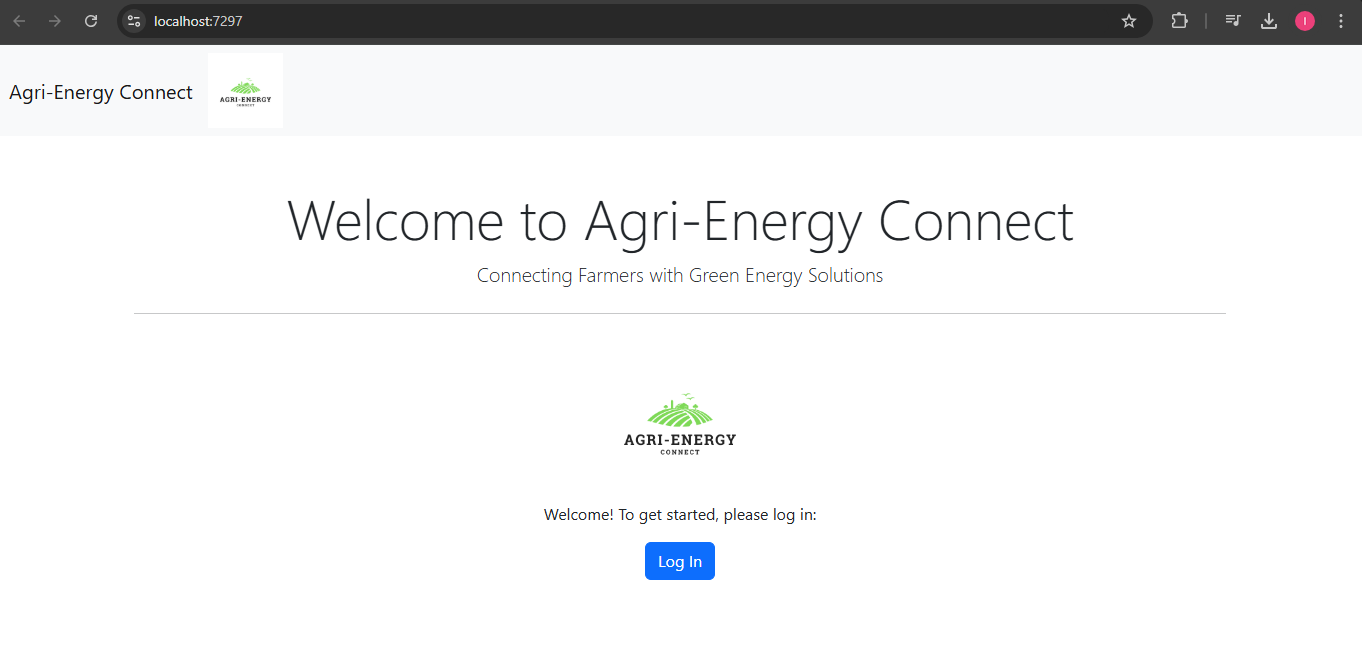
The platform Agri-Energy Connect uses ITIL and TOGAF frameworks to oversee daily tasks and develop structures that align with business goals. Advantages consist of improved coordination, efficiency in service delivery, decreased risk, and enhanced performance for users.

# Technical Solution Overview:

**Introduction**

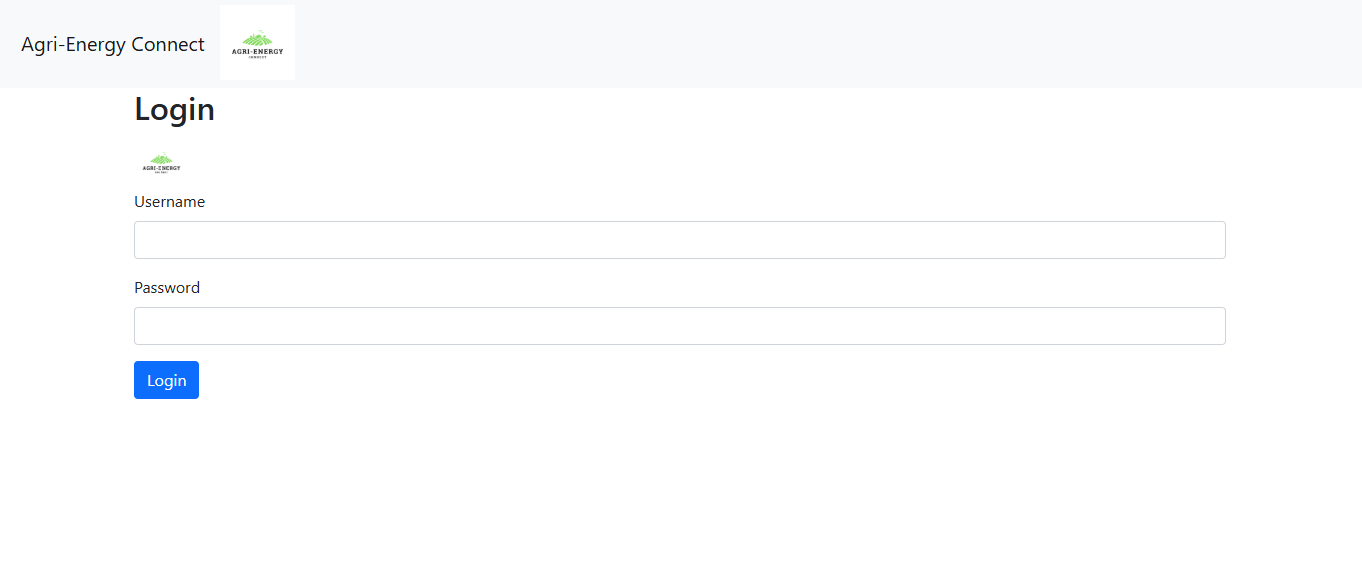
The Agri-Energy Connect platform is a digital tool that connects farmers with renewable energy suppliers. It provides an all-encompassing solution for sustainable farming and clean energy.

App Key Functionalities:  
Homescreen:



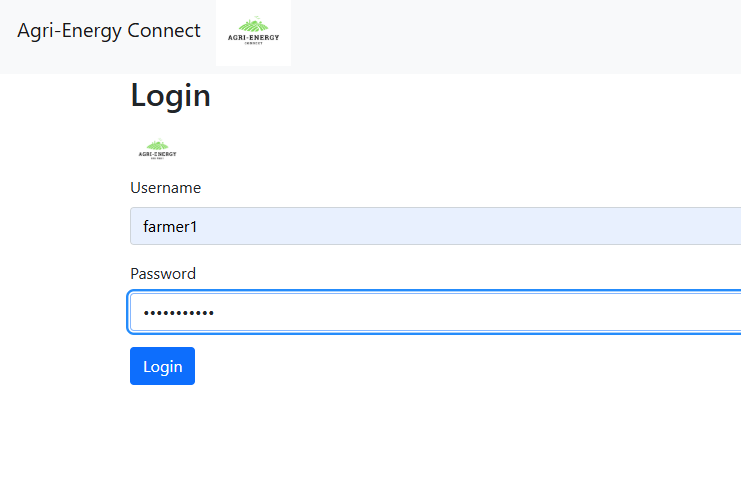
**Authentication and Authorization:**

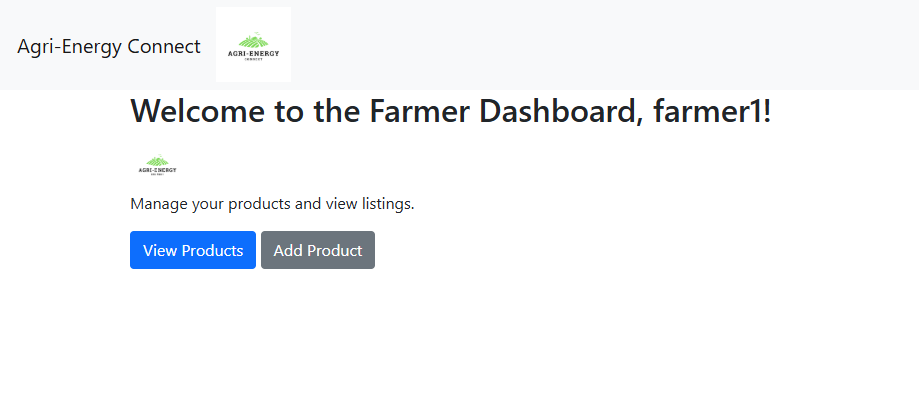
* Secure user login and role-based access control

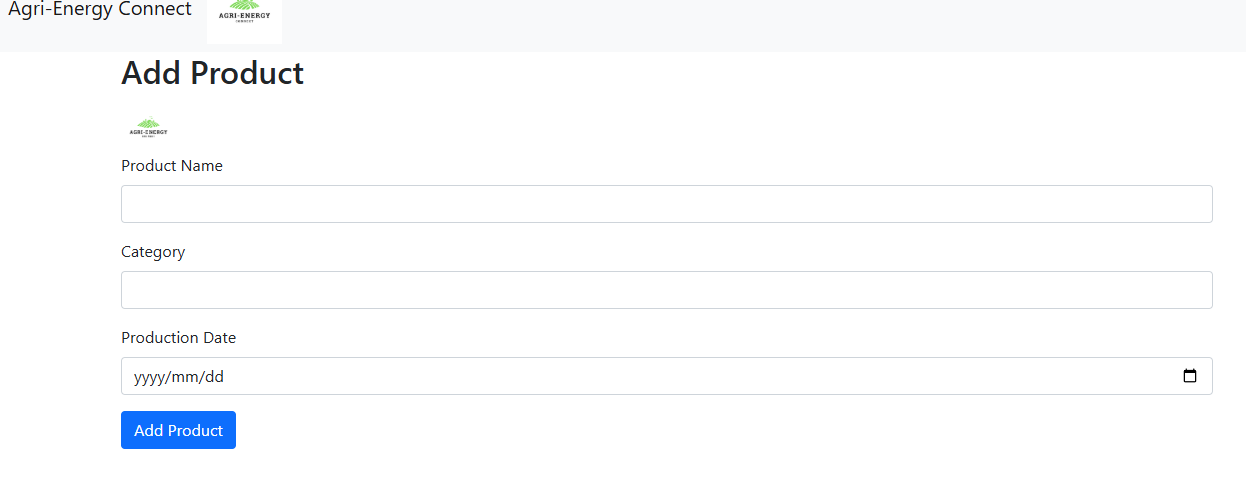


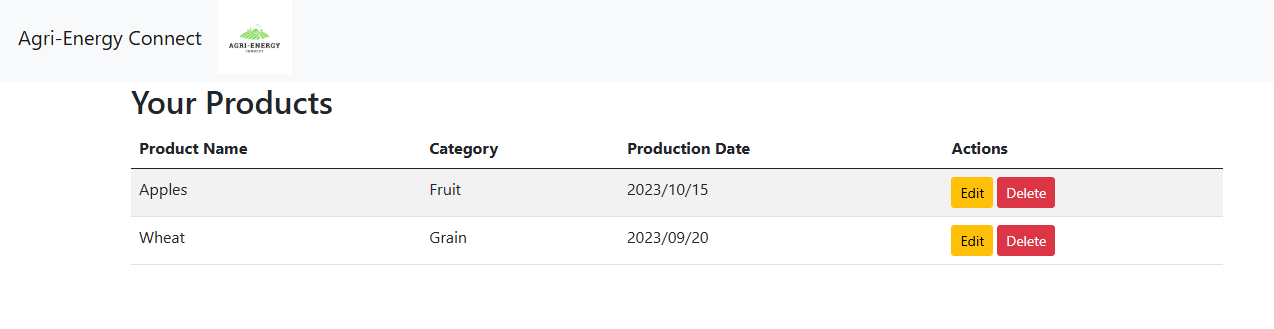
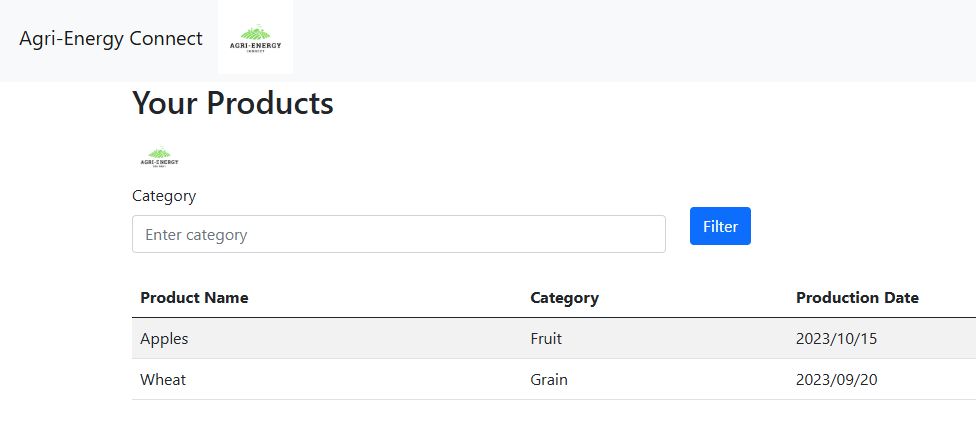
**Farmer Dashboard:**

* Add products
* View product listings
* Edit product
* Delete product



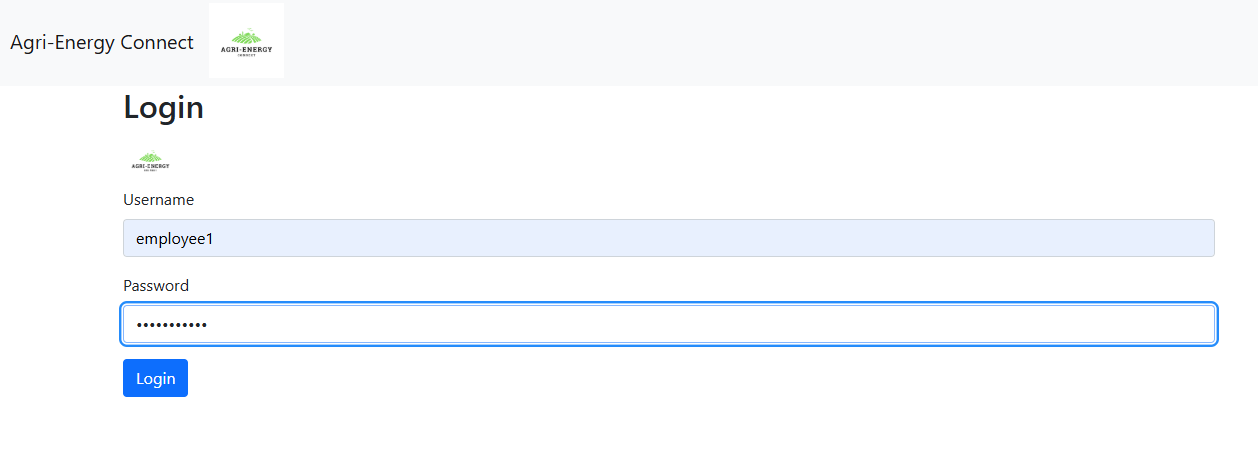


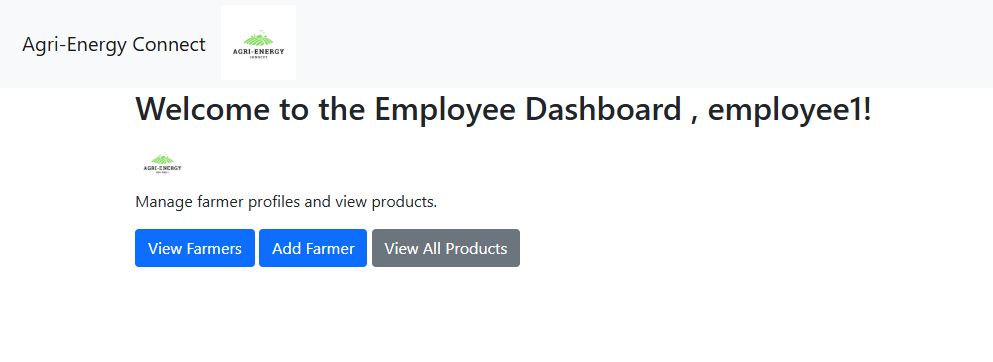


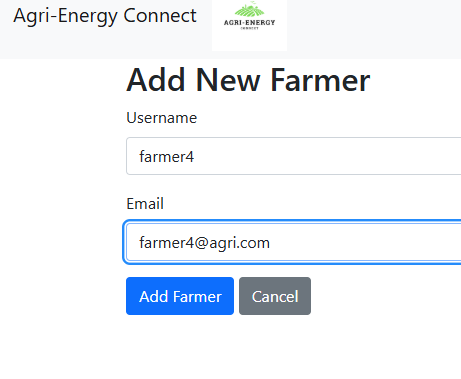
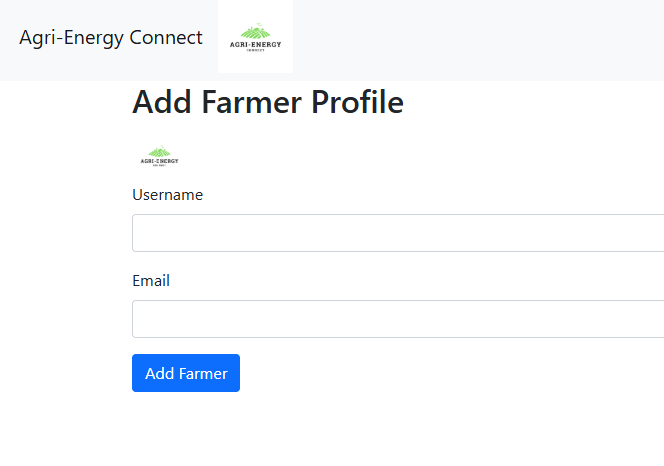


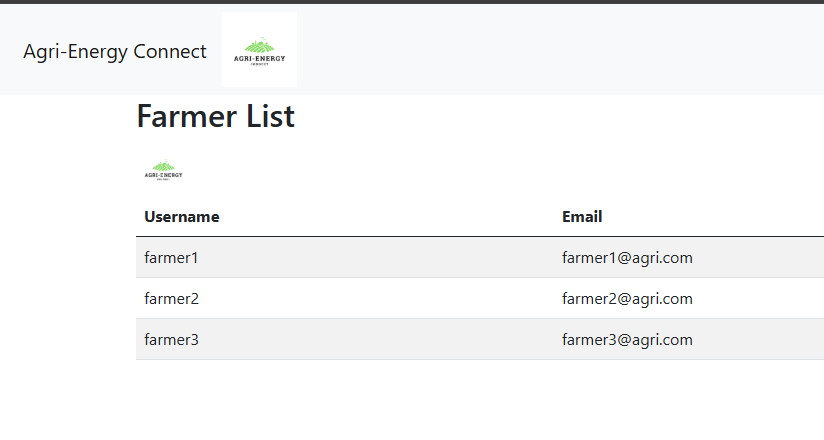
**Employee Dashboard:**

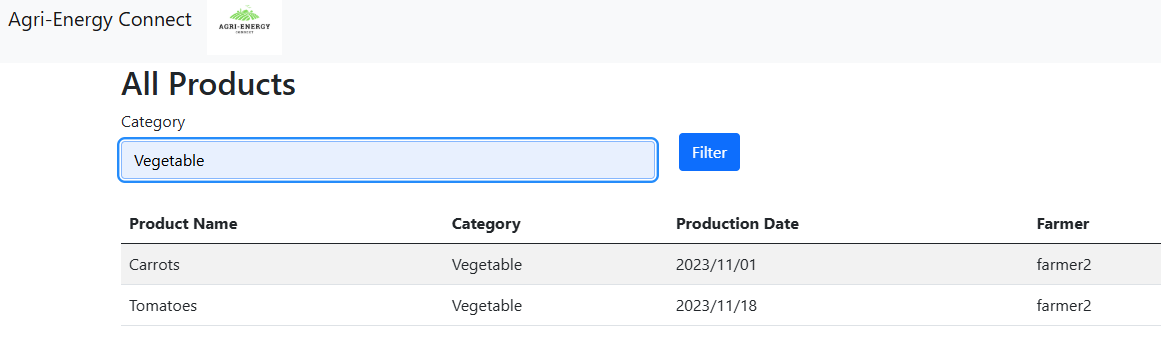
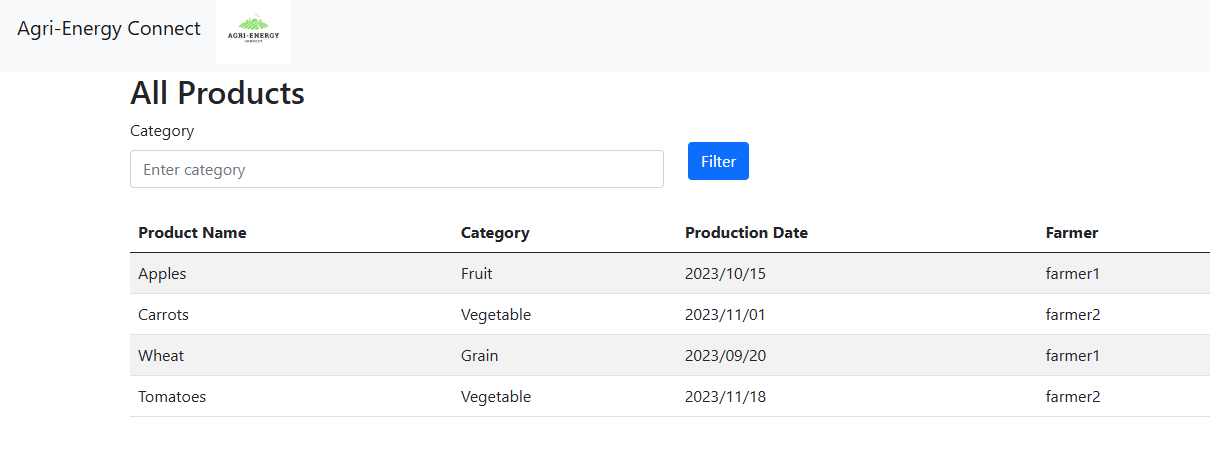
* View Farmers
* Add, edit, and delete farmer profiles
* View and filter all products











**Technical Solution:**

* System built on .NET, C#, modern front-end framework
* Hierarchical structure with clear responsibilities
* Model-View-Controller, Dependency Injection used for modularity.

**Business Value:**

* The Agri-Energy Connect platform offers advantages to farmers and energy providers through improved efficiency, lowered costs, enhanced sustainability, and better market access.
* The platform's goal is to build a sustainable and prosperous future for agriculture through technology and innovation.

References:

* **Clarion Technologies.** (n.d.). Cloud Computing Architecture: What is Front-End and Back-End? Retrieved from <https://www.clariontech.com/blog/cloud-computing-architecture-what-is-front-end-and-back-end>
* **Atlassian.** (2023). ITIL: The IT Infrastructure Library. Retrieved from <https://www.atlassian.com/itsm/itil>
* **Simplilearn.** (n.d.). TOGAF Certification Benefits. Retrieved from <https://www.simplilearn.com/togaf-certification-benefits-article>
* **Red Hat.** (n.d.). TOGAF: A Framework for Enterprise Architecture. Retrieved from <https://www.redhat.com/en/blog/togaf>
* **Atlassian.** (2023). Service Strategy. Retrieved from <https://www.atlassian.com/itsm/itil/service-strategy>
* **IT Process Maps.** (n.d.). ITIL Service Transition. Retrieved from <https://wiki.en.it-processmaps.com/index.php/ITIL_Service_Transition>
* **Troelsen, A., & Japikse, P. (2017). *Pro C# 7 with .NET and .NET Core.* 8th ed. Apress.**
* **Pressman, R. S. (2010). Software Engineering: A Practitioner's Approach.** McGraw-Hill Education.
* **Sommerville, I. (2016). Software Engineering, 10th Edition.** Pearson Education.
* **Lewis, J., & Fowler, M. (2017). Microservices: Designing Fine-Grained Systems.** O'Reilly Media.
* **Welling, L., & Fox, K. (2016). Cloud Native Java: Designing Reliable Systems with Patterns and Principles.** O'Reilly Media.