Short Capstone Project

Team 4

Organic Farm

**1. Overview:**

* The Organic Farms System is a web-based application designed to connect farming partners with customers, allowing the purchase of organic products through one-time orders or subscriptions. The system aims to facilitate quick delivery and a seamless user experience.

**2. Functional Requirements:**

* **User Authentication:**
  + Users must be able to sign up, log in, and manage their accounts securely.
  + Different roles (customers, admin) should have specific access rights.
* **Product Management:**
  + Admins can add, edit, or remove products.
  + Products include fruits, vegetables, and pulses with details like name, type, price, and availability.
* **Order Management:**
  + Customers can place orders for products with the option to subscribe on a weekly or monthly basis.
  + The system should calculate the total amount for the order based on product prices and quantities.
* **Payment Processing:**
  + Integration with a payment gateway for processing payments securely.
  + Users should receive confirmation of payment status.
* **Reviews and Ratings:**
  + Customers can leave reviews and ratings for products.
* **User Dashboard:**
  + Customers can view their order history and manage current subscriptions.

**3. Non-Functional Requirements:**

* **Performance:**
  + The application should be optimized for performance to handle multiple users and orders simultaneously.
* **Security:**
  + Implement JWT for secure user authentication and authorization.
  + Ensure sensitive data, such as passwords and payment information, is encrypted.
* **Scalability:**
  + The system should be designed to scale horizontally by adding more microservices as needed.
* **Usability:**
  + The user interface should be intuitive, responsive, and accessible on different devices.

**4. Technical Requirements:**

* **Frontend:**
  + Technologies: HTML5, CSS3, JavaScript, Angular.
  + Implement a responsive design using Bootstrap.
* **Backend:**
  + Technologies: Java, Spring Boot, Spring Data JPA, Hibernate, MySQL.
  + Implement RESTful APIs for CRUD operations.
* **Microservices:**
  + Design the system architecture to use microservices.
  + Implement an API Gateway for routing requests to the appropriate services.
* **Cloud Deployment:**
  + Deploy the application on Azure using Azure DevOps for CI/CD.
  + Use Docker for containerization and Kubernetes for orchestration.

**5. Testing Requirements:**

* Implement unit tests and integration tests using Mocha, Chai, and Jest to ensure code quality and functionality.
* Conduct performance testing to evaluate the system's responsiveness under load.

**6. Documentation:**

* Create comprehensive documentation covering system architecture, API specifications, user guides, and deployment instructions.