Chapter 1
INTRODUCTION

- ## Preamble

Farmers are the backbone of the Indian economy, yet they face persistent challenges in securing fair profits for their hard work. The dominance of intermediaries and the lack of pricing transparency often compel them to sell their produce at unreasonably low prices, while intermediaries and vendors reap higher margins. This disparity not only undermines the financial stability of farmers but also discourages the adoption of farming as a sustainable profession.

The Harvest Auction Hub envisions transforming this reality by creating a transparent, technology-driven platform that directly connects farmers with buyers. By eliminating unnecessary middlemen, ensuring fair pricing, and introducing modern agricultural practices, the Harvest Auction Hub strives to empower farmers, enhance their profitability, and make farming a viable and respected profession.

Together, we aim to revolutionize the agricultural ecosystem, bridging the gap between farmers and markets while fostering prosperity for those who feed the nation

- ## Problem Definition

Farmers face significant challenges in earning fair profits for their hard work due to the dominance of intermediaries (middleman) and the lack of pricing transparency.

Despite being the backbone of the Indian economy, farmers are often forced to sell their crops at low prices, while intermediaries and vendors gain higher margins.

These issues are exacerbated by limited farming technologies making farming a financially unviable profession for many.

3. Objectives

1. **Facilitate Transparent Pricing:**
   Establish a fair and transparent bidding process that ensures farmers receive competitive prices for their produce.

2. **Eliminate Middlemen:**
   Directly connect farmers with retailers and buyers, minimizing the role of intermediaries and maximizing profits for farmers.

3. **Promote Technological Integration:**
   Introduce and implement advanced agricultural and digital technologies to improve efficiency in farming and market access.

4. · **Ensure Sustainability and Profitability:**
   Empower farmers with a sustainable and financially viable model for selling their produce, ensuring long-term economic stability.

5. · **Build a Farmer-Centric Ecosystem:**
   Create a supportive ecosystem that addresses farmers' challenges holistically, fostering trust and mutual benefit between stakeholders

1.4 Motivation

The motivation behind Harvest Auction Hub lies in the urgent need to empower farmers and restore equity in the agricultural value chain. By establishing a centralized platform that connects farmers directly with retailers, we aim to foster transparent bidding processes, ensure fair pricing, and remove the inefficiencies caused by middlemen.

This solution aspires to not only improve the livelihoods of farmers but also create a ripple effect of growth and innovation in the agricultural ecosystem. Through the adoption of technology and fair trade practices, Harvest Auction Hub seeks to build a future where farming is profitable, sustainable, and valued as the cornerstone of national progress.

Chapter 2
SOFTWARE REQUIREMENT SPECIFICATIONS

2.1 Functional Requirements and use case diagram

1. Bidding System

- Farmers can list their crops on the platform, specifying a starting bid price for each product.
- Retailers can view the listed crops and place competitive bids within the auction timeline.
- This process ensures dynamic pricing where farmers receive fair compensation based on market demand and competition among buyers.
- The system records all bids, providing transparency and fair access for both farmers and retailers.

2. Product Management

- Farmers have tools to manage their crop listings, allowing them to:
- Add new products they wish to sell.
- Update details such as quantity, quality, and pricing.

- Remove products that are no longer available.
- These features ensure accurate and up-to-date product information for buyers.

3. Email Notifications

- The platform provides timely updates to users through automated emails, such as:
- Notification to farmers when new bids are placed on their crops.
- Alerts to buyers when their bids are successful or outbid.
- Updates for both parties on order placements, delivery schedules, and status changes.
- This keeps all stakeholders informed and engaged throughout the process.

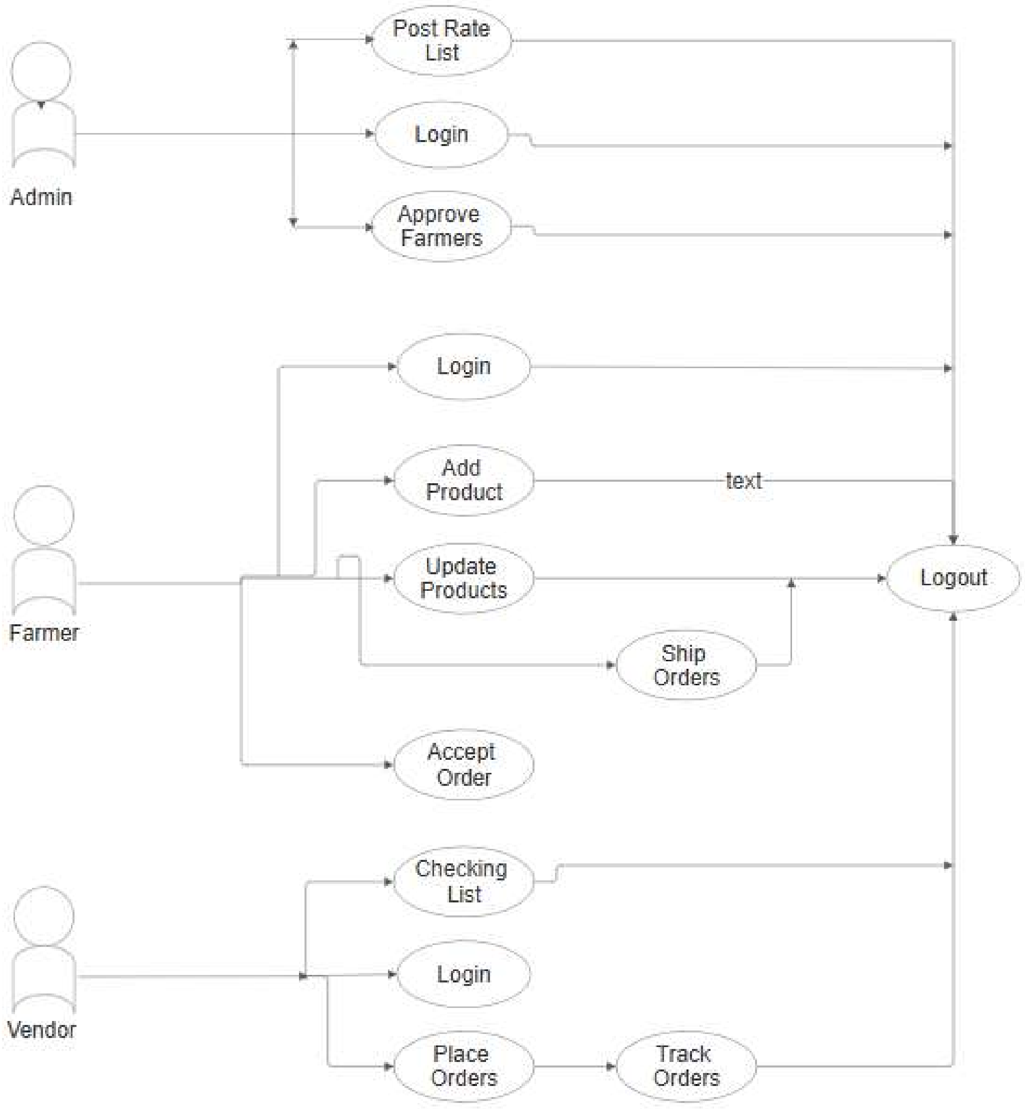
4. Admin Dashboard

- The administrative interface offers oversight and control over platform activities, including:
- Monitoring sales performance and transaction data in real time.
- Setting pricing caps to prevent exploitative practices and ensure market stability.
- Tracking user activity to detect issues, improve

operations, and maintain platform integrity.
- This dashboard helps ensure the system operates efficiently and aligns with its mission to support farmers.

5. Product Search

- Vendors (retailers and other buyers) can easily locate crops using advanced search and filtering options.
- Filters may include parameters such as crop type, location, quality grade, price range, and availability.
- This functionality saves time, enhances the user experience, and facilitates faster transactions.



2.2 Non-Functional Requirements

· Scalability

- The platform should be designed to accommodate an increasing number of users, product listings, and transactions over time.
- It must support growth without degrading performance, ensuring that new users and higher transaction volumes do not impact the user experience.

- This can be achieved through scalable infrastructure such as cloud computing, load balancing, and modular architecture.

· Security

- Passwords are hashed with bcrypt and stored securely in the database.
- Sensitive user data is encrypted during communication.

· Performance

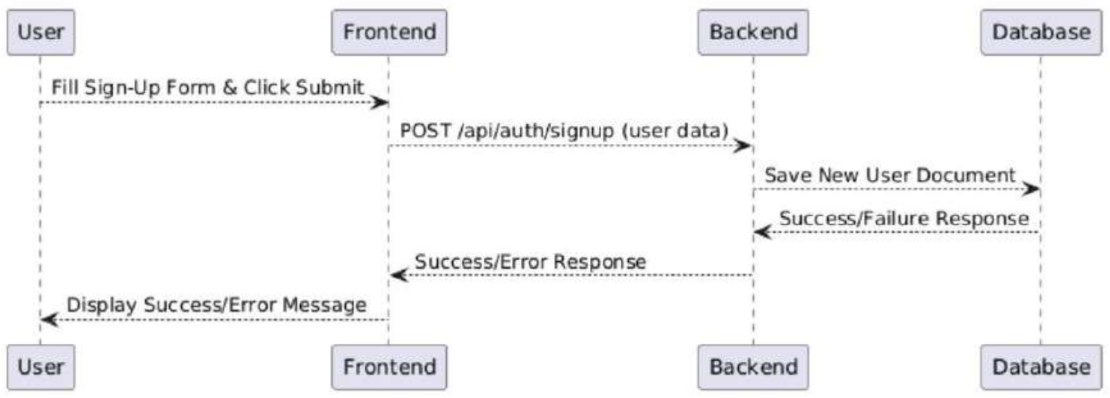
- The platform must efficiently handle high volumes of concurrent users and transactions, especially during peak periods such as harvest seasons or bidding wars.
- Performance optimization techniques, including caching, database indexing, and efficient query handling, should be implemented.
- Real-time responsiveness for actions like bidding updates and notifications is essential to maintaining user satisfaction.

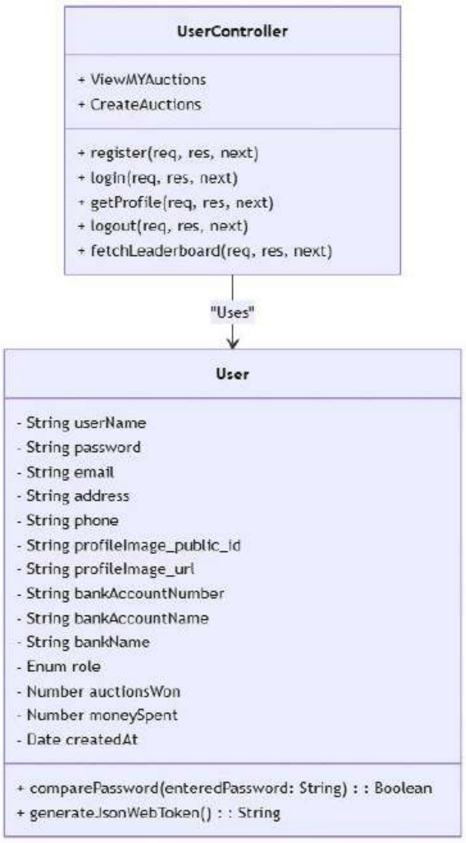
· Usability

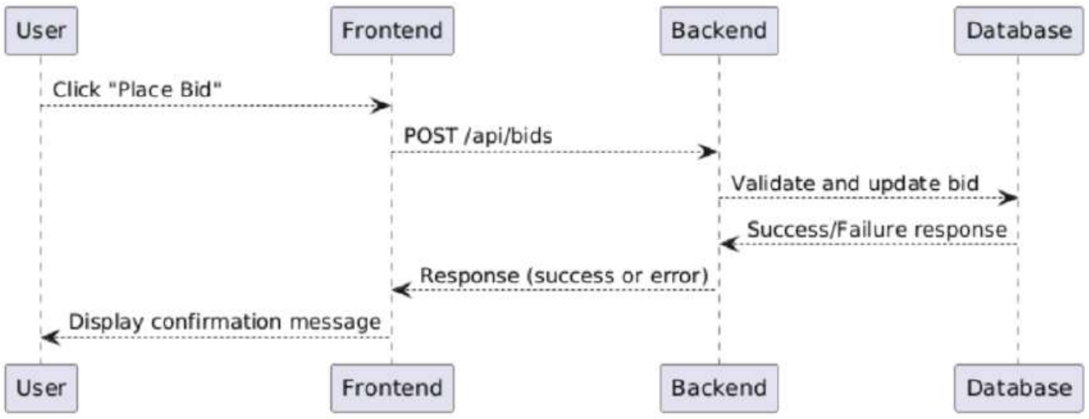
- The platform's UI/UX design should prioritize simplicity, accessibility, and ease of navigation for all users, including farmers with limited digital experience.
- Features such as intuitive menus, guided workflows, and localized language support enhance usability.
- Continuous feedback loops and usability testing ensure the platform evolves based on user needs.

· Reliability

- The system must provide consistent and dependable performance, even during heavy workloads or unexpected spikes in activity.
- Features like redundancy, failover mechanisms, and regular backups ensure the system remains operational and minimizes downtime.
- Reliable operations install confidence in users, making the platform a trusted tool for their business needs.







2.3 Hard ware and software requirements

Hardware Requirements

1. **Processor:** Intel i5/i7 or equivalent AMD processor (or higher)
2. **RAM:** Minimum 8GB (16GB recommended for smoother performance)
3. **Storage:**

- 500GB HDD or 256GB SSD (minimum)
- Additional storage for project files and dependencies (~10GB)

4. **Graphics:** Integrated GPU is sufficient, but a dedicated GPU (e.g., NVIDIA/AMD) is recommended for high-resolution designs or large datasets.
5. **Operating System:** Windows 10/11, macOS 10.15+ (Catalina or later), or a Linux distribution like Ubuntu 20.04+.

Software Requirements

1. **Operating System:**
   - Windows 10/11, macOS, or Linux
2. **IDE/Code Editor:**
   - [VS Code](https://code.visualstudio.com/) (recommended)
   - Other options: Atom, WebStorm, or Sublime Text
3. **Node.js and npm:**
   - npm (comes with Node.js) or yarn for package management
4. **Framework and Libraries:**
   - React.js (latest version)
   - React Router for navigation
   - State management library (e.g., Redux or Context API)
   - Tailwind CSS/ Vite
5. **Backend Requirements (if applicable):**

- Node.js with Express.js or other backend frameworks
- Database: MongoDB.

2.4 Test plan and Test cases

1)Test Cases for User Registration and Authentication Module:

Test Case ID	Description	Steps	Expected Outcome	Status

TC001	Verify that users can register successfully	1. Open the application and navigate to the sign-up page 2. Enter a valid full name, email, phone number, address, role, and password 3. Upload a valid profile image 4. Submit the form	The user account is created, and the user is redirected to the home page/dashboard	Pass
TC002	Verify that users can register successfully	1. Open the application and navigate to the sign-up page 2. Enter a valid full name, email, leave other fields blank 3. Upload a valid profile image 4. Submit the form	Form is not submitted. User is not registered	Pass
TC003	Verify that users can log in using valid credentials	1. Open the application and navigate to the login page 2. Enter a valid email address and password 3. Click the login button	The user is logged in successfully and redirected to the dashboard	Pass

2) Test Cases for Product Management Module (Auction):

Test Case ID	Description	Steps	Expected Outcome	Status
TC004	Add a new product listing	1. Log in as a Auctioneer 2. Navigate to “Create Auction" 3. Enter product details 4. Click “Create Auction”	Product added to the auction list	Pass
TC005	Republish existing product details	1. Log in as a Auctioneer 2. Navigate to "My Auctions” 3. Select a product 4. Edit details 5. Save changes	Product Republished successfully	Pass
TC006	Delete a product listing	1. Log in as a Auctioneer 2. Navigate to "My Auctions” 3. Select a product 4. Click "Delete"	Product removed from the auction list	Pass

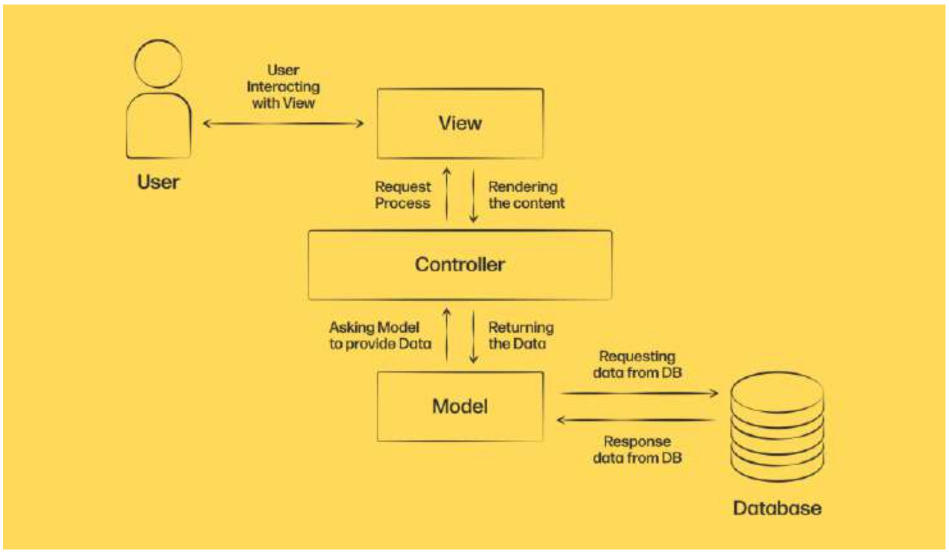
3) Test Cases for Bidding System Module:

Test Case ID	Description	Steps	Expected Outcome	Status
TC007	Place a bid on a product	1. Log in as a Bidder 2. View active auctions 3. Select a product 4. Enter bid amount 5. Submit bid	Bid successfully placed	Pass
TC008	Attempt bid lower than the starting price	1. Log in as a Bidder 2. View active auctions 3. Select a product 4. Enter bid amount lower than starting price 5. Submit bid	Error message: "Bid must be higher than starting price"	Pass

4) Test Cases for Notification and Communication Module:

Test Case ID	Description	Steps	Expected Outcome	Status
TC009	Receive email notification for successful bid winning	1. Log in as a Bidder 2. Place a valid bid 3. Check email	Email notification received for winning	Pass
TC010	Notify Admin of mails from users	1. Log in as a Bidder or Auctioneer 2. Navigate to contact us page and fill the details with message 3. Admin receives notification	Notification received: “mail from user”	Pass

Chapter 3
SYSTEM DESIGN & IMPLEMENTATION
3.1 MVC Architecture



1.Model(Data & Business Logic Layer)

- Database: MongoDB to store user profiles, product details, bidding data, and notifications.
- Business Logic: Manages product listings, bidding workflows, notifications, and user authentication.
- Data Handling: Supports CRUD operations for all modules.

  Purpose: Handles all data-related tasks, enforces business rules, and updates the Controller with data for the View.

2.View (User Interface Layer)

Components:

- HTML/CSS/JavaScript: Provides page structure and styling.

- React.js: Builds interactive UI elements like real-time bids, search filters, and user dashboards.

  Purpose: Displays data and collects user inputs for actions such as bidding, product management, and notifications.

3. Controller (Mediator Layer)

   Components:

- Node.js & Express.js: Handles API routes for authentication, CRUD operations, and notifications.
- API Logic: Mediates between View and Model to process requests and deliver responses.

  Purpose: Facilitates communication between the View and Model, processes user actions, and ensures smooth data flow.

3.2 MERN framework

1.MongoDB (Database Layer):

- Stores user profiles, product listings, bids, and notifications.
- Enables scalable and efficient data management for CRUD operations like adding products or recording bids.

2. Express.js (Backend Framework)

- Manages server-side APIs for user authentication, product management, and bidding.
- Handles routing, data validation, and secure communication with the database.

3. React.js (Frontend Framework)

- Builds an interactive and responsive user interface.

- Features include real-time bid updates, search filters, and dashboards for farmers and vendors.

4. Node.js (Runtime Environment)

- Executes server-side JavaScript and handles high user loads.
- Ensures non-blocking, fast processing of requests from React.js and database interactions.

MERN work flow:

1. React.js sends user inputs (e.g., adding a product, placing a bid) to the Express.js API.

2.Node.js processes the inputs, interacts with the MongoDB database to perform the necessary operations, and returns responses to the frontend.

3.The React.js frontend dynamically updates based on responses, ensuring a smooth user experience.

3.3 Detailed Database Description(Mongo DB/ MySQL)

1. Database Structure

- Users: Stores details of farmers and vendors.
- Auctions: Maintains information about listed crops/products.
- Bids: Records all bidding activities and winning bids.

2. Key Collections and Schema

➢Users Collection

- userId (Primary Key): Unique identifier for each user.
- name: User’s full name.
- email: Contact email for login and notifications.

- password: Encrypted password for authentication.
- role: Specifies role (Auctioneer or Bidder).
- contactDetails: Stores phone number and address.

➢Products Collection

- productId (Primary Key): Unique identifier for each product.
- CreatedBy (Foreign Key): Links to the farmer listing the product.
- productName: Name of the crop/product.
- description: Details about the product.
- Startingbid: Starting bid price for the product.
- Bids: Array of all bids
- createdAt: Timestamp for when the product was listed.

➢Bids Collection

- bidId (Primary Key): Unique identifier for each bid.
- productId (Foreign Key): Links to the product being bid on.
- UserId (Foreign Key): Links to the vendor placing the bid.
- bidAmount: The amount offered by the vendor.
- createdAt: Timestamp for when the bid was placed.

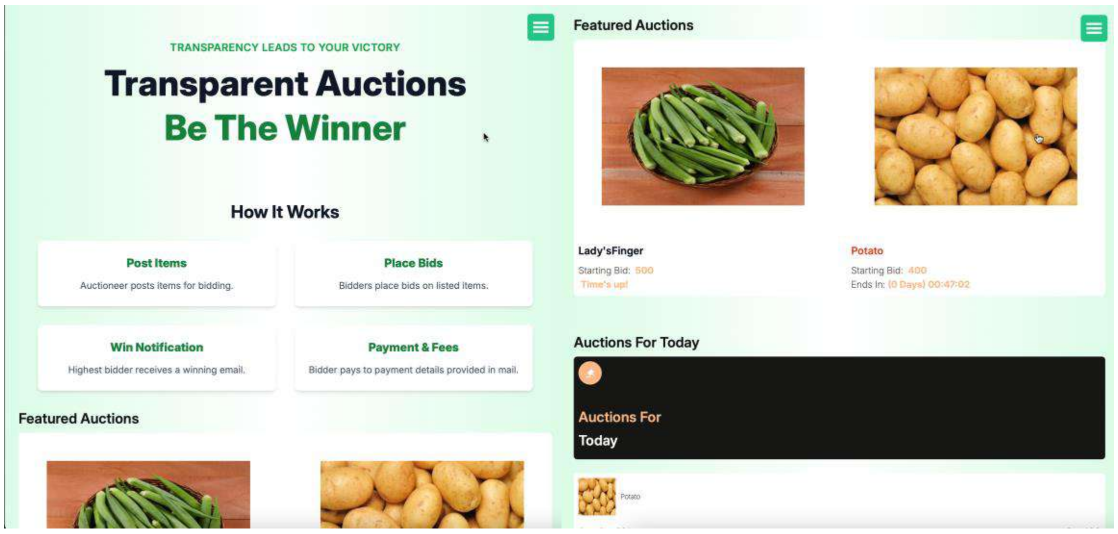
3.4 Modules Description & Implementation

Implementation:

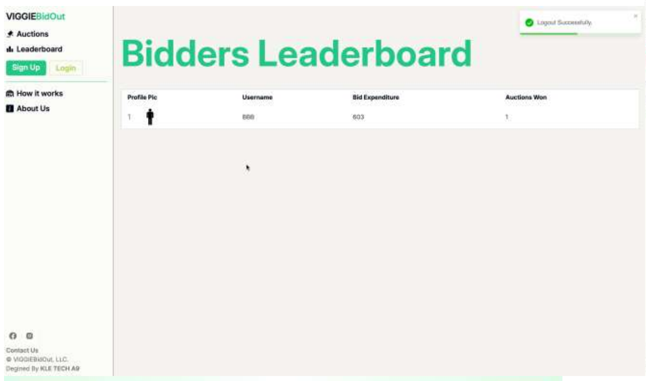
The "VIGGIEBidOut" project is designed to directly connect farmers with retailers, eliminating intermediaries and ensuring fair pricing for agricultural products. The platform leverages a combination of web technologies to provide an intuitive and secure solution.

Technologies Used

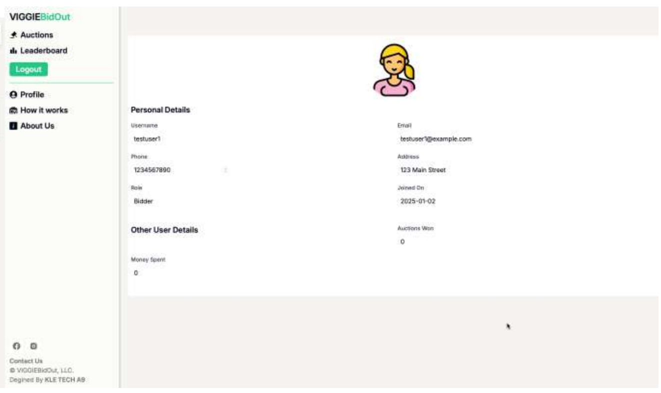
1. **Frontend Technologies:**
    - **HTML, CSS, JavaScript:** To design a visually appealing and user-friendly interface.
    - **React.js:** Used for dynamic frontend development, enabling smooth transitions and modular UI components like product listings and bidding forms.
2. **Backend Technologies:**
    - **Express.js / Node.js:** The backend framework handles routing for APIs such as user authentication, product listing, bidding, and notifications.
    - **MongoDB:** A NoSQL database used for storing user data, product details, and bid history in a structured format.
3. **Security Features:**
    - **Authentication:** The system uses secure login mechanisms and role-based access to differentiate between farmers and vendors.
    - **Encryption:** All sensitive data, including passwords, is hashed and stored securely to prevent breaches.
4. **Communication and Notifications:**
    - Email notifications are sent for critical actions like bid updates, order placements, and delivery confirmations.



[HOME PAGE]



Profile Pic	Username	Bid Expenditure	Auctions Won
1	BBB	603	1



Personal Details

Username: testuser1	Email: testuser1@example.com
Phone: 1234567890	Address: 123 Main Street
Role: Bidder	Joined On: 2025-01-02

Other User Details

Money Spent: 0	Auctions Won: 0

Profile Pic	Username	Bid Expenditure	Auctions Won
1	BBB	603	1

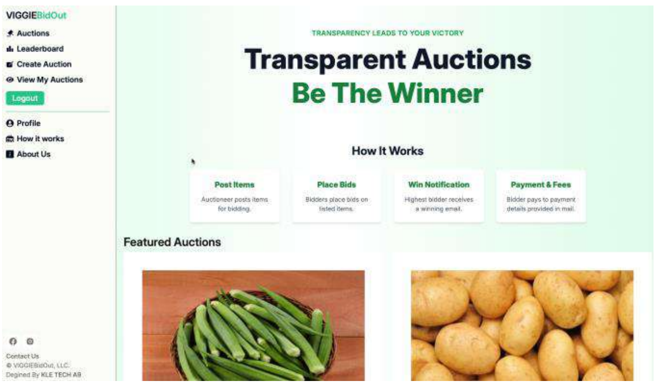
[PROFILE AND LEADERBOARD PAGE]

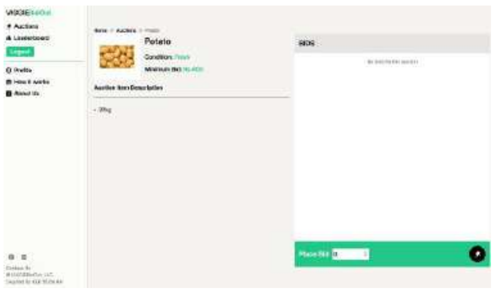
Modules and Functionalities:

1. User Registration and Authentication:

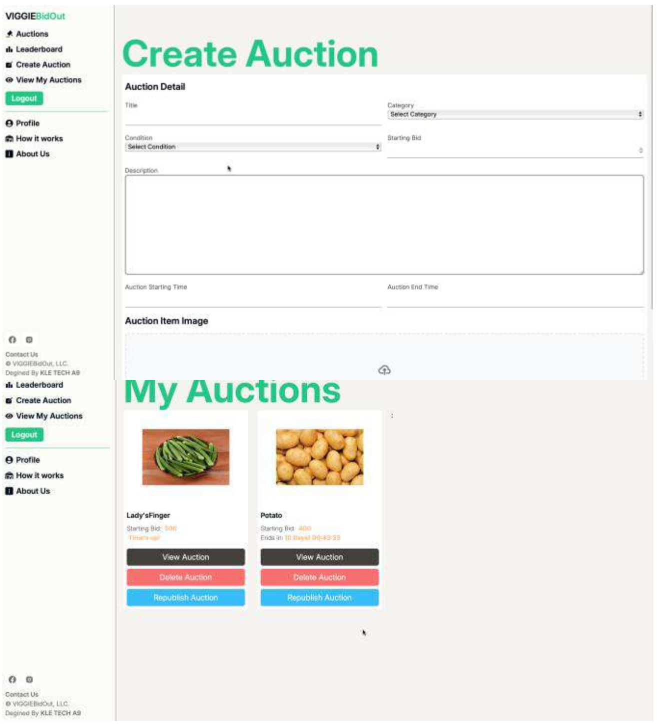
- ◦ Role assignment during registration (Auctioneer or Bidder).
- ◦ Secure login and password hashing to protect sensitive data.

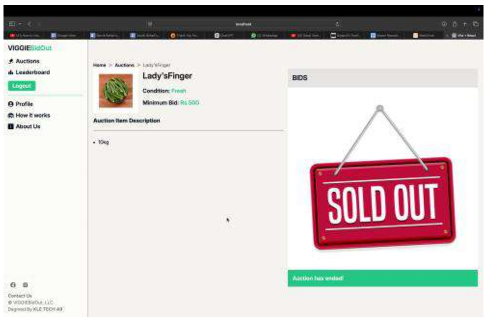
[Auctineer’s View]



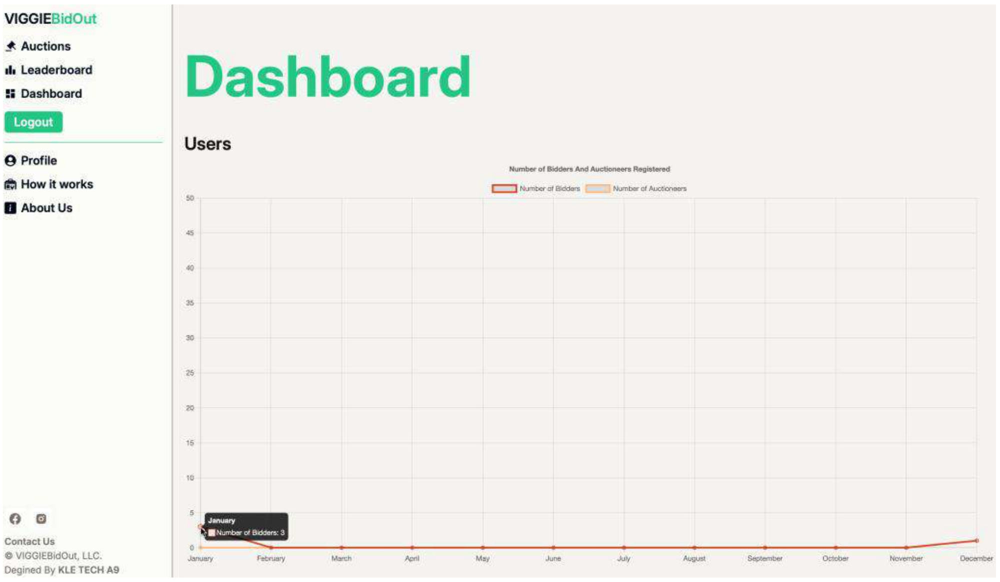


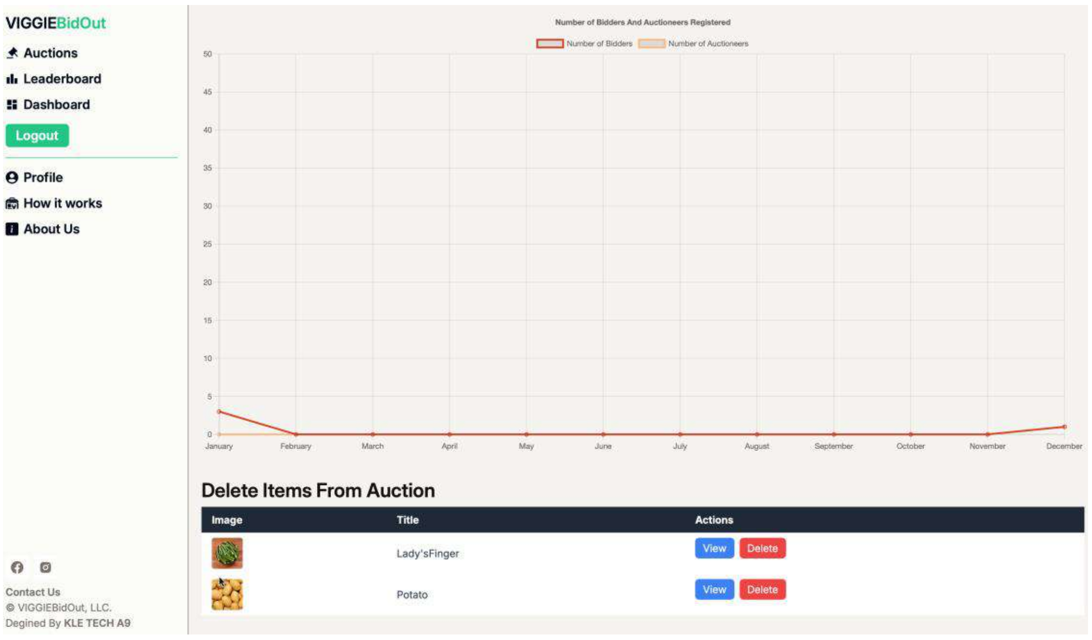
[BIDDER’s View]





[Admin View]





Delete Items From Auction

Image	Title	Actions
	Lady'sFinger	View Delete
	Potato	View Delete

Modules and Functionalities:

1. User Registration and Authentication:
    - Role assignment during registration (Auctioneer or Bidder).
    - Secure login and password hashing to protect sensitive data.
2. Product Management:

- Farmers can add, edit, or remove crop listings.

3. Bidding System:
   - Farmers set starting bids for their products.
   - Vendors place competitive bids.
   - Alerts notify bidders of bid status and order confirmation.
4. Admin Dashboard:
   - An interface for administrators to monitor sales, set pricing caps, and oversee system activity.
5. Notifications and Communication:
   - Email alerts for bid status, payment updates, and more ensure seamless interaction between all stakeholders.

Backend Features

1. **Security:**
   - Passwords are hashed with bcrypt and stored securely in the database.
   - Sensitive user data is encrypted during communication.
2. **Database Management:**
   - MongoDB ensures efficient storage and retrieval of data for users, products, and transactions.

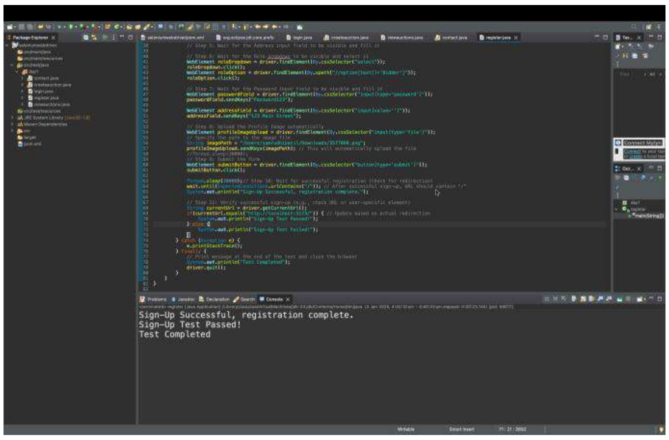
- Mongoose is used to streamline database interactions through schemas

3. **Routing and APIs:**

   - Express.js powers backend routing for user authentication, product management, and other operations.
   - RESTful APIs are implemented for secure communication between the frontend and backend.

Chapter 4
RESULTS AND DISCUSSIONS

Test Case1: User Registration



```
        // Step 5: Wait for the Address input field to be visible and fill it

        // Step 6: Wait for the Role dropdown to be visible and select it
        WebElement roleDropdown = driver.findElement(By.cssSelector("select"));
        roleDropdown.click();
        WebElement roleOption = driver.findElement(By.xpath("//option[text()='Bidder']"));
        roleOption.click();

        // Step 7: Wait for the Password input field to be visible and fill it
        WebElement passwordField = driver.findElement(By.cssSelector("input[type='password']"));
        passwordField.sendKeys("Password123");

        WebElement addressField = driver.findElement(By.cssSelector("input[value='']"));
        addressField.sendKeys("123 Main Street");

        // Step 8: Upload the Profile Image automatically
        WebElement profileImageUpload = driver.findElement(By.cssSelector("input[type='file']"));
        // Specify the path to the image file
        String imagePath = "[illegible]";
        profileImageUpload.sendKeys(imagePath); // This will automatically upload the file
        //Thread.sleep(20000);
        // Step 9: Submit the form
        WebElement submitButton = driver.findElement(By.cssSelector("button[type='submit']"));
        submitButton.click();

        Thread.sleep(20000);// Step 10: Wait for successful registration (check for redirection)
        wait.until(ExpectedConditions.urlContains("/")); // After successful sign-up, URL should contain "/"
        System.out.println("Sign-Up Successful, registration complete.");

        // Step 11: Verify successful sign-up (e.g., check URL or user-specific element)
        String currentUrl = driver.getCurrentUrl();
        if(currentUrl.equals("http://localhost:5173/")) { // Update based on actual redirection
            System.out.println("Sign-Up Test Passed!");
        } else {
            System.out.println("Sign-Up Test Failed!");
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        // Print message at the end of the test and close the browser
        System.out.println("Test Completed");
        driver.quit();
    }
  }
}
```

```
Sign-Up Successful, registration complete.
Sign-Up Test Passed!
Test Completed
```

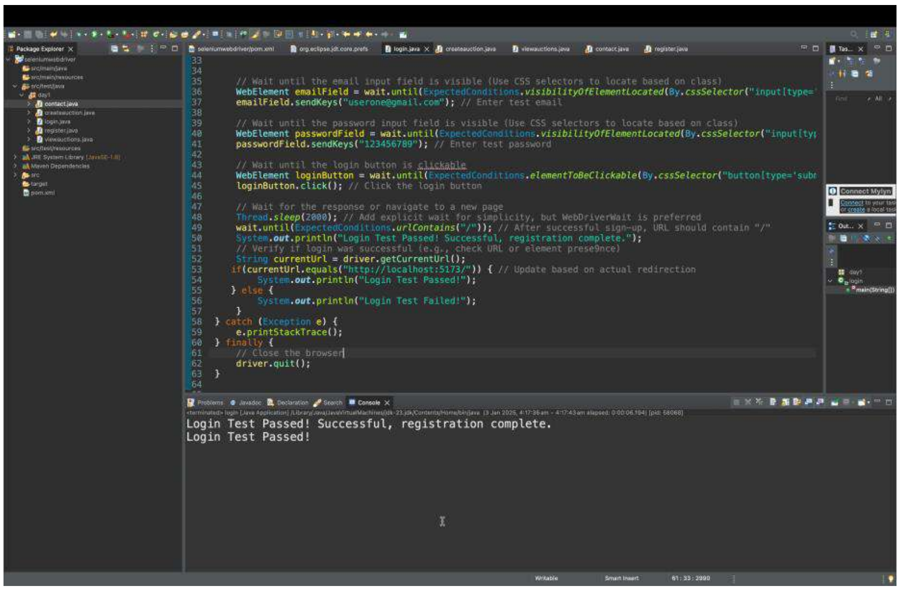
Description:
This test case verifies the registration functionality of the locally hosted website (`http://localhost:5173/sign-up`). It inputs valid data for all fields, uploads a valid profile image, and submits the form. The test checks if the user account is created successfully and if the user is redirected to the home page or dashboard.

Expected Output:
The user should be redirected to `http://localhost:5173/dashboard` or the homepage upon successful registration. A success message should be displayed, and no error messages should appear.

Actual Output:
The test passed as expected; the user was successfully redirected to the dashboard/homepage. The registration was completed without any errors, and a success notification was displayed.

Importance:
This test ensures that the registration functionality works correctly with valid inputs. It validates the proper functioning of user account creation, profile image upload, and redirection mechanisms, contributing to the seamless onboarding of new users.

Test Case: User Login



```
        // Wait until the email input field is visible (Use CSS selectors to locate based on class)
        WebElement emailField = wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector("input[type='
        emailField.sendKeys("userone@gmail.com"); // Enter test email

        // Wait until the password input field is visible (Use CSS selectors to locate based on class)
        WebElement passwordField = wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector("input[typ
        passwordField.sendKeys("123456789"); // Enter test password

        // Wait until the login button is clickable
        WebElement loginButton = wait.until(ExpectedConditions.elementToBeClickable(By.cssSelector("button[type='subm
        loginButton.click(); // Click the login button

        // Wait for the response or navigate to a new page
        Thread.sleep(2000); // Add explicit wait for simplicity, but WebDriverWait is preferred
        wait.until(ExpectedConditions.urlContains("/")); // After successful sign-up, URL should contain "/"
        System.out.println("Login Test Passed! Successful, registration complete.");
        // Verify if login was successful (e.g., check URL or element prese9nce)
        String currentUrl = driver.getCurrentUrl();
       if(currentUrl.equals("http://localhost:5173/")) { // Update based on actual redirection
            System.out.println("Login Test Passed!");
       } else {
            System.out.println("Login Test Failed!");
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        // Close the browser
        driver.quit();
    }
```

```
<terminated> login [Java Application] /Library/Java/JavaVirtualMachines/jdk-23.jdk/Contents/Home/bin/java (3 Jan 2025, 4:17:36 am – 4:17:43 am elapsed: 0:00:06.104) [pid: 58068]
Login Test Passed! Successful, registration complete.
Login Test Passed!
```

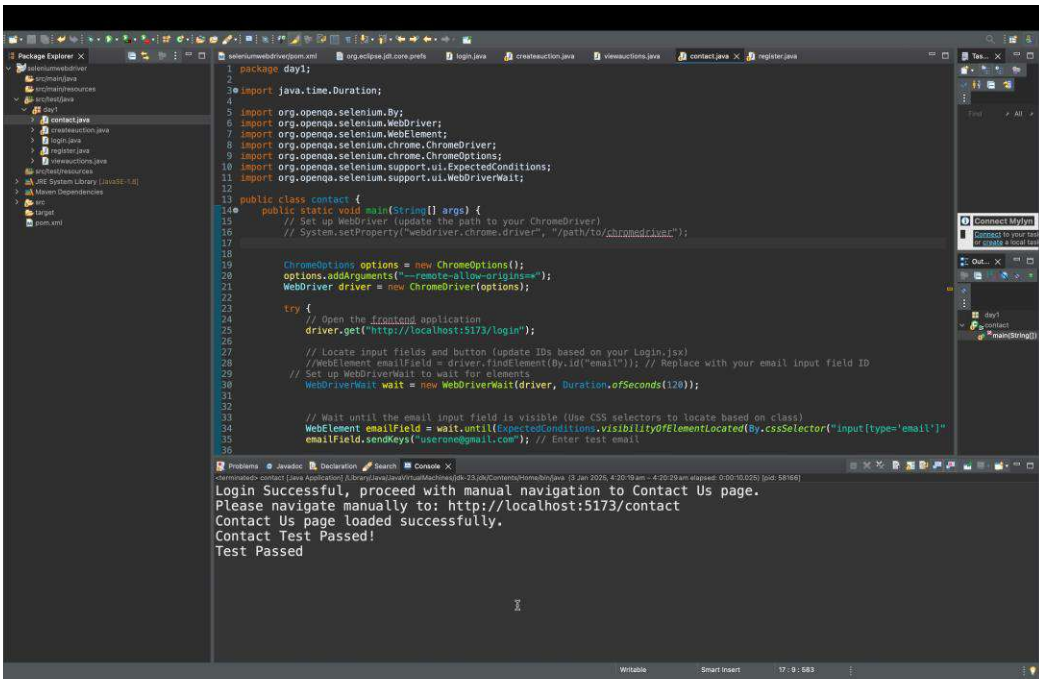
Description:
This test case verifies the login functionality of the locally hosted website (`http://localhost:5173/login`). It inputs valid credentials (e.g., `johndoe@example.com` and `Password123`), submits the login form, and checks if the user is redirected to the dashboard.

Expected Output:
The user should be redirected to `http://localhost:5173/dashboard` upon successful login with valid credentials. No error messages should appear.

Actual Output:
The test passed as expected; the user was successfully redirected to the dashboard. No error messages were encountered during the test.

Importance:
This test ensures that the login functionality works correctly with valid credentials. It confirms that the authentication process, session management, and redirection mechanisms are functioning as intended, enhancing user experience and system reliability.

Test Case: Email Notification on Contact



```java
package day1;

import java.time.Duration;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

public class contact {
    public static void main(String[] args) {
        // Set up WebDriver (update the path to your ChromeDriver)
        // System.setProperty("webdriver.chrome.driver", "/path/to/chromedriver");


        ChromeOptions options = new ChromeOptions();
        options.addArguments("--remote-allow-origins=*");
        WebDriver driver = new ChromeDriver(options);

        try {
            // Open the frontend application
            driver.get("http://localhost:5173/login");

            // Locate input fields and button (update IDs based on your Login.jsx)
            //WebElement emailField = driver.findElement(By.id("email")); // Replace with your email input field ID
         // Set up WebDriverWait to wait for elements
            WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(120));


            // Wait until the email input field is visible (Use CSS selectors to locate based on class)
            WebElement emailField = wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector("input[type='email']"
            emailField.sendKeys("userone@gmail.com"); // Enter test email

```

```
<terminated> contact [Java Application] /Library/Java/JavaVirtualMachines/jdk-23.jdk/Contents/Home/bin/java (3 Jan 2025, 4:20:19 am – 4:20:29 am elapsed: 0:00:10.025) [pid: 58166]
Login Successful, proceed with manual navigation to Contact Us page.
Please navigate manually to: http://localhost:5173/contact
Contact Us page loaded successfully.
Contact Test Passed!
Test Passed
```

Description:
This test case verifies the "Contact Us" functionality of the website (`http://localhost:5173/contact`). It inputs valid data for name, email, subject, and message fields, submits the form, and checks if the email is sent successfully.

Expected Output:
A success notification should be displayed to the user, confirming that the email has been sent to the designated recipient. No error messages should appear.

Actual Output:
The test passed as expected; the email was sent successfully, and a success notification was displayed. No errors were encountered during the test.

Importance:
This test ensures that the email functionality works as intended. It validates the integration with the email service, enabling seamless communication between users and support, which is critical for customer satisfaction and issue resolution.

Chapter 5
CONCLUSION & FUTURE SCOPE

Conclusion

The Harvest Auction Hub successfully addresses the challenges faced by farmers by providing a transparent, centralized platform to connect them directly with vendors. It eliminates intermediaries, ensures fair pricing through a competitive bidding system, and empowers farmers with better financial prospects. The use of the MERN stack ensures scalability, efficiency, and a seamless user experience. This project has the potential to significantly improve the agricultural trade ecosystem by fostering trust and collaboration between farmers and vendors.

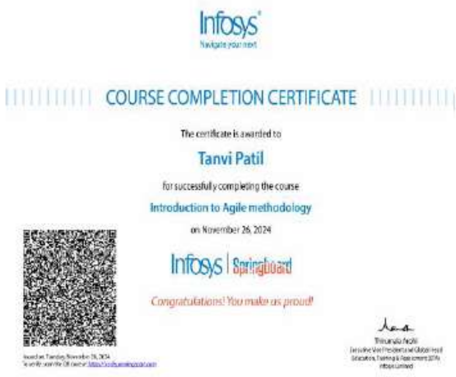
Future Scope

1. Mobile Application Development:
   - Expand the platform by introducing a mobile app for enhanced accessibility, especially for rural users.
2. Integration of Advanced Technologies:
   - AI and Machine Learning for predictive analytics on crop pricing and demand forecasting.
   - Blockchain to further enhance transparency and security in transactions.
3. Multi-language Support:
   - Implement support for multiple languages to cater to diverse user bases across regions.
4. Partnerships and Expansion:
   - Collaborate with logistics companies to offer delivery services.
   - Expand to new markets, enabling cross-region or international trade.
5. Enhanced Features:
   - Real-time chat functionality for seamless farmer-vendor communication.
   - Subscription plans for premium services like advanced analytics and advertising.

6. Government Integration: Partner with agricultural departments to offer subsidies, training, and government support schemes directly on the platform.

Chapter 6
INFOSYS SPRINGBOARD CERTIFICATION COURSE ON DEVOPS

Name : Tanvi Patil


Infosys
Navigate your next

COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Tanvi Patil

for successfully completing the course

Introduction to Agile methodology

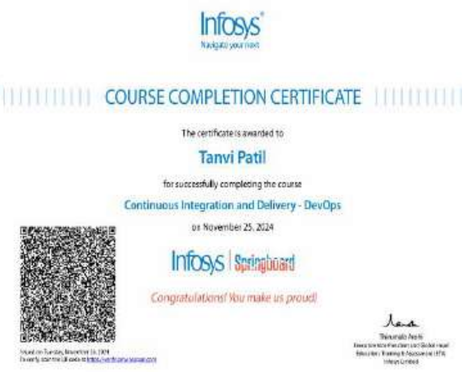
on November 26, 2024

Infosys | Springboard

Congratulations! You make us proud!

Issued on: Tuesday, November 26, 2024
To verify, scan the QR code at https://verify.onwingspan.com

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited


Infosys
Navigate your next

COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Tanvi Patil

for successfully completing the course

Continuous Integration and Delivery - DevOps

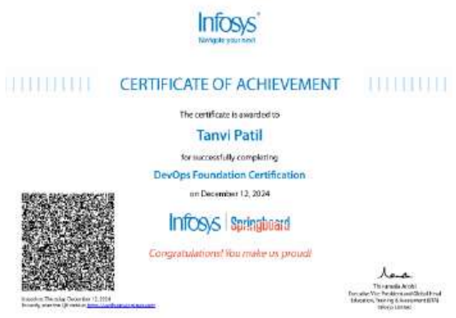
on November 25, 2024

Infosys | Springboard

Congratulations! You make us proud!

Issued on: Tuesday, November 26, 2024
To verify, scan the QR code at https://verify.onwingspan.com

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited


Infosys
Navigate your next

CERTIFICATE OF ACHIEVEMENT

The certificate is awarded to

Tanvi Patil

for successfully completing

DevOps Foundation Certification

on December 12, 2024

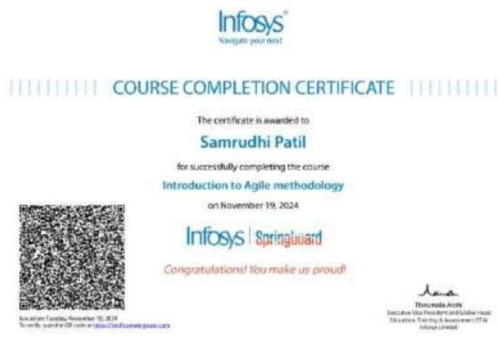
Infosys | Springboard

Congratulations! You make us proud!

Issued on: Thursday, December 12, 2024
To verify, scan the QR code at https://verify.onwingspan.com

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited

Name: Samrudhi Patil


Infosys
Navigate your next

COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Samrudhi Patil

for successfully completing the course

Introduction to Agile methodology

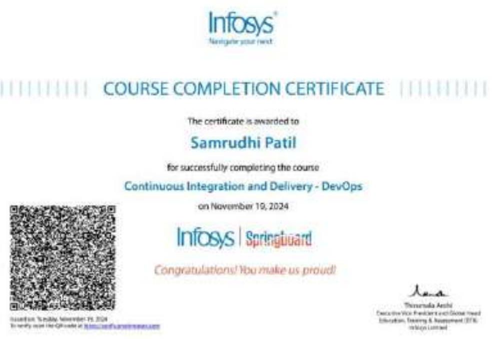
on November 19, 2024

Infosys | Springboard

Congratulations! You make us proud!

Issued on: Tuesday, November 19, 2024
To verify, scan the QR code at https://verify.onwingspan.com

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited


Infosys
Navigate your next

COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Samrudhi Patil

for successfully completing the course

Continuous Integration and Delivery - DevOps

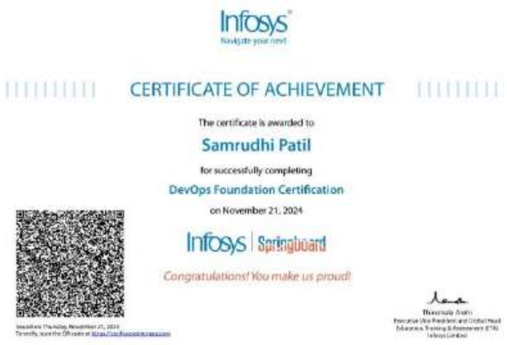
on November 19, 2024

Infosys | Springboard

Congratulations! You make us proud!

Issued on: Tuesday, November 19, 2024
To verify, scan the QR code at https://verify.onwingspan.com

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited



Infosys
Navigate your next

CERTIFICATE OF ACHIEVEMENT

The certificate is awarded to

Samrudhi Patil

for successfully completing

DevOps Foundation Certification

on November 21, 2024

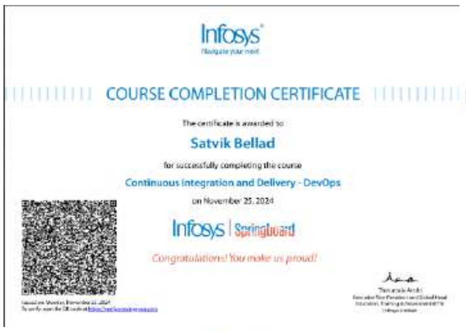
Infosys | Springboard

Congratulations! You make us proud!

Issued on: Thursday, November 21, 2024
To verify, scan the QR code at [illegible]

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited

Name:Satvik Bellad



Infosys
Navigate your next

COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Satvik Bellad

for successfully completing the course

Continuous Integration and Delivery - DevOps

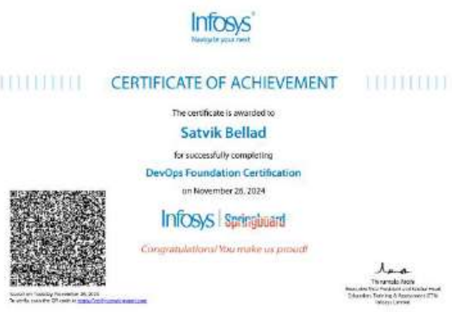
on November 25, 2024

Infosys | Springboard

Congratulations! You make us proud!

Issued on: Monday, November 25, 2024
To verify, scan the QR code at [illegible]

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited



Infosys
Navigate your next

CERTIFICATE OF ACHIEVEMENT

The certificate is awarded to

Satvik Bellad

for successfully completing

DevOps Foundation Certification

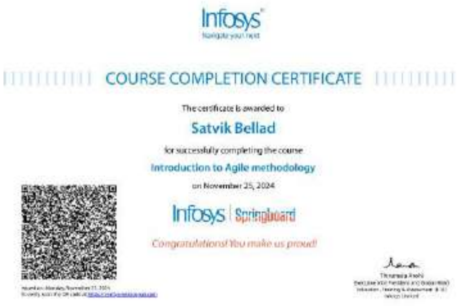
on November 26, 2024

Infosys | Springboard

Congratulations! You make us proud!

Issued on: Tuesday, November 26, 2024
To verify, scan the QR code at [illegible]

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited



Infosys
Navigate your next

COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Satvik Bellad

for successfully completing the course

Introduction to Agile methodology

on November 25, 2024

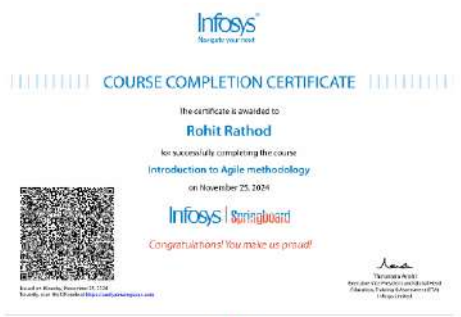
Infosys | Springboard

Congratulations! You make us proud!

Issued on: Monday, November 25, 2024
To verify, scan the QR code at [illegible]

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited

Name :Rohit Rathod



COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Rohit Rathod

for successfully completing the course

Introduction to Agile methodology

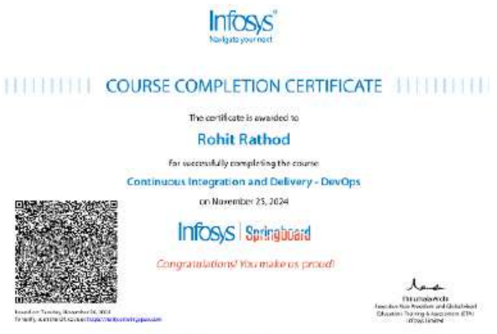
on November 25, 2024

Congratulations! You make us proud!

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited

[illegible]

<u>Introduction to Agile methodology</u>



COURSE COMPLETION CERTIFICATE

The certificate is awarded to

Rohit Rathod

for successfully completing the course

Continuous Integration and Delivery - DevOps

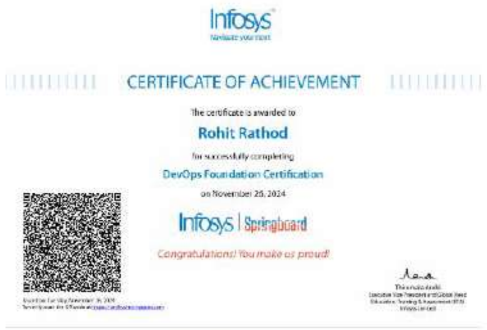
on November 25, 2024

Congratulations! You make us proud!

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited

[illegible]

<u>Continuous Integration and Delivery -DevOps</u>



CERTIFICATE OF ACHIEVEMENT

The certificate is awarded to

Rohit Rathod

for successfully completing

DevOps Foundation Certification

on November 26, 2024

Congratulations! You make us proud!

Thirumala Arohi
Executive Vice President and Global Head
Education, Training & Assessment (ETA)
Infosys Limited

[illegible]

<u>DevOps Foundation Certification</u>