A

Project Report On

Deal Plus

B.Tech CE Sem-V

Subject :(CE-520)

Advanced Technologies

Prepared by:

Meet Antala (CE092) (22CEUOS096)

Guided by:

Prof. Prashant M. Jadav

Associate Professor

Faculty of Technology

Department of Computer Engineering

Dharmsinh Desai University

Nadiad

Dharmsinh Desai University

Faculty of Technology, College Road, Nadiad – 387001, Gujarat

CERTIFICATE

This is to certify that the term work carried out in the subject of (**CE520**) **Advanced Technologies** and submitted is the bonafide work of **Mr. Meet Antala** Roll No.: **CE092** Identity No.: **22CEUOS096** of B.Tech. **Semester V** in the branch of **Computer Engineering** during the academic year 2024-2025.

Prof. P. M. Jadav	Dr. C. K. Bhensdadia
Associate Professor	Head, CE Department

Table of Contents

1.Abstract

DealPlus is an online marketplace developed using the MERN stack (MongoDB, Express.js, React, and Node.js). It is designed to connect buyers and sellers in a convenient and user-friendly way. The platform allows users to easily register and log in. Once users are logged in, they can create and manage their product listings. This includes adding new products, updating existing details, and removing items they no longer want to sell. If they want to buy some product then they can make request to seller. The site has a clean design, making it easy for users to navigate.

.

2. Introduction

2.1 Brief Introduction:

DealPlus is an online marketplace designed to connect buyers and sellers in a simple and effective way. The platform allows sellers to create and manage product listings while buyers can easily browse, search, and purchase items. Key features include product management, user accounts, and tools that enhance the online shopping experience.

2.2 Scope:

The DealPlus marketplace aims to connect users with a wide range of products, enabling sellers to showcase their items and buyers to find what they need with ease. Core features include user registration, product listings, search functionality, and secure transactions. The platform supports multimedia uploads for product images, user reviews, and real-time messaging between buyers and sellers, creating a seamless shopping environment. Administrators will manage operations, ensuring smooth interactions between users and efficient course management.

2.3 Technology/Platform/Tools Used:

- **Frontend:** React.js
- **Backend:** Node.js with Express.js
- **Authentication:** JWT (JSON Web Tokens)
- **Database:** MongoDB
- **File Storage:** Clouldinary for image uploads
- **Development Tools:** VS Code, Postman
- **Deployment Tools:** Localhost

3. System Analysis

3.1 Use-case diagram

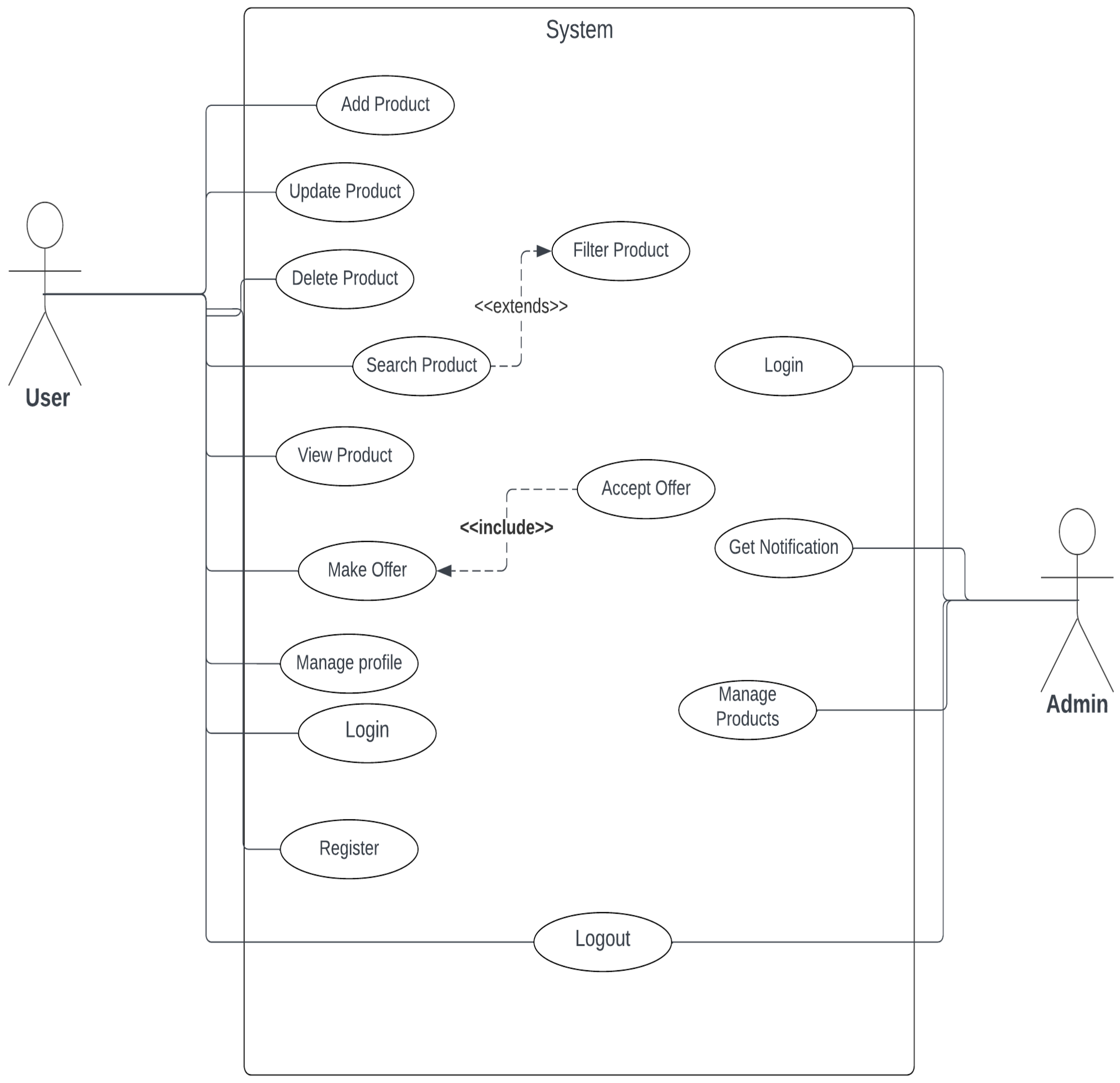


Figure 3.1 Use Case

3.1 Sequence diagram (Add product and make bid)

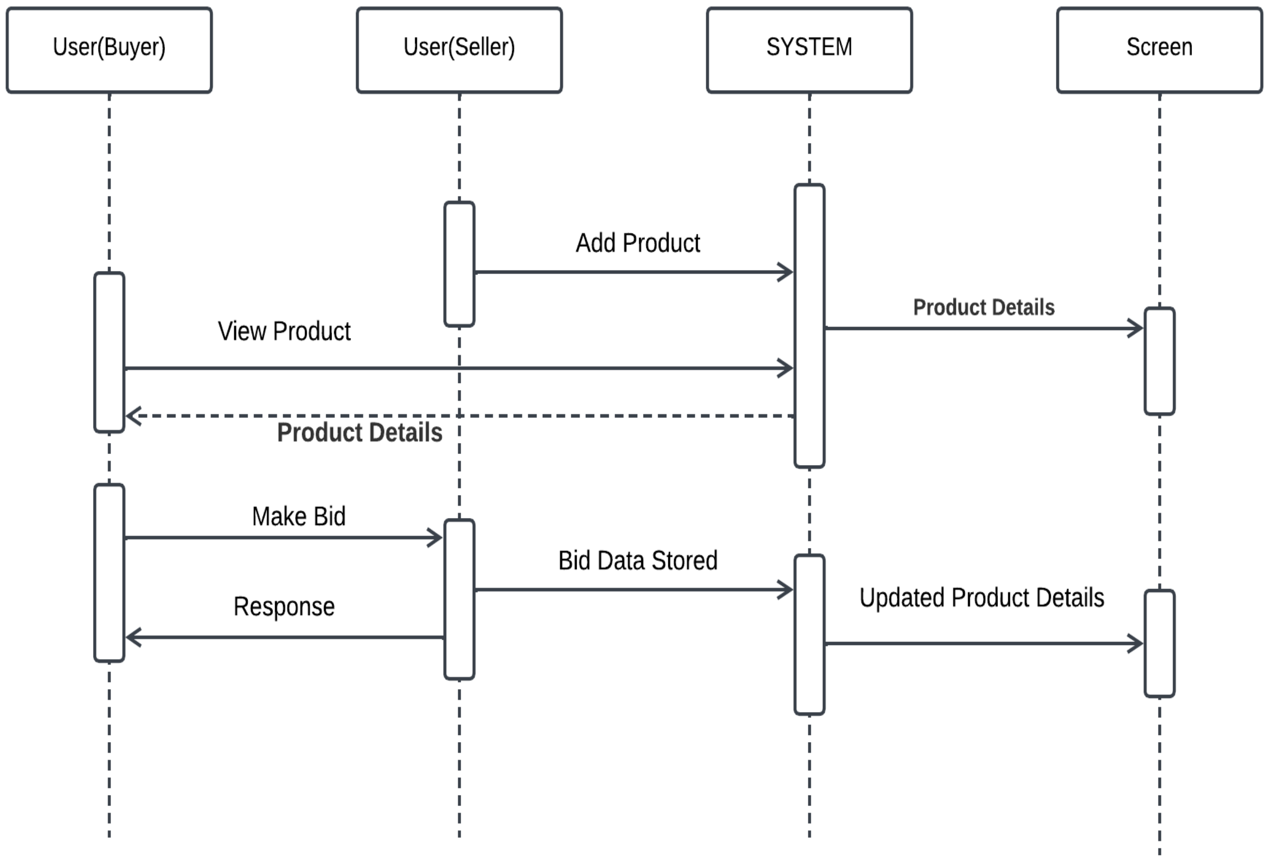


Figure 3.1 Sequence Diagram

4. Software Requirements Specifications :

4.1 Overall Description

4.1.1 Product Perspective

DealPlus is a standalone web application that integrates with various third-party services for user authentication. It will be accessible via web browsers on desktop and mobile devices.

4.1.2 Product Functions

- User registration and authentication
- Profile management
- Viewing products
- Product listing creation, editing, and deletion
- Product search and filtering
- Price negotiation between buyers and sellers
- Admin dashboard

4.1.3 User Classes and Characteristics

- **Buyers**: Users looking to purchase goods or services.
- **Sellers**: Users looking to sell goods or services.
- **Admins**: Users with elevated privileges to manage the platform.

4.1.4 Operating Environment

DealPlus will operate in a web environment and will be compatible with major browsers (Chrome, Firefox, Safari, Edge).

4.1.5 Design and Implementation Constraints

- Use of the MERN stack (MongoDB, Express.js, React, Node.js)
- Compliance with security and data privacy regulations
- Scalability to handle increasing user load

4.1.6 User Documentation

User manuals and help guides will be provided to assist users in navigating and utilizing the platform effectively.

4.1.7 Assumptions and Dependencies

- Users have access to the internet.
- Users have basic knowledge of web navigation.
- The application will use third-party services for payment and authentication.

4.2 System Features

- User Authentication
- Profile Management
- View Products
- Product Listings
- Search and Filters
- Product Details View
- Price Negotiation by providing Bid
- Admin Dashboard

4.3 Functional Requirements

4.3.1 Signup / Login

4.3.1.1 Login

- **Description:** Registered users can log in by entering their credentials. The system verifies user details for authentication.
  - **4.3.1.1.1: Provide Login Details**
    - **State:** User is already registered successfully.
    - **Input:** Enter username and password.
    - **Output:** Login if details are correct.

4.3.1.2 Signup

- **Description:** The system facilitates user registration, allowing new users to create accounts securely. Users need to provide details like Name, username, password, email, etc. Upon registration, user details are stored in the database.
  - **4.3.1.2.1: Enter Registration Details**
    - **State:** Text fields for different details are presented.
    - **Input:** User enters name, username, password, email, and mobile number.
    - **Output:** Registration successful, now User can Login.

4.3.2 View Products

4.3.2.1View Product

- **Description:** Users can view detailed information about products, including images and descriptions, but cannot access exclusive products.
  - **4.3.2.1.1 : Display Product Details**
    - **Input:** Access the product page.
    - **Output:** Display product details available to all users.

4.3.3 Explore Products

4.3.3.1 Search Product

- **Description:** Users can search for products by entering keywords. The system matches keywords with items in the product list and displays the most relevant results.
  - **4.3.3.1.1: Select Search Option**
    - **Input:** Click on the "Search" option.
    - **Output:** Prompt to enter keywords.
  - **4.3.3.1.2 : Perform Search and Display**
    - **Input:** Enter keywords for searching item.
    - **Output:** Display details of items based on entered keywords.

4.3.3.2 Filter Search

- **Description:** Users can filter search results based on categories, price ranges, and other criteria.
  - **4.3.3.2.: Apply Filters**
    - **Input:** Select filter options (e.g., category, price range).
    - **Output:** Display filtered search results.

4.3.4 Manage Product

4.3.4.1 Upload New Product

- **Description:** Sellers can upload new products to their store profile, providing details such as an image, price, description, and brand name.
  - **4.3.4.1.1: Upload Product**
    - **Input:** Seller uploads an image, adds price, description, and brand name.
    - **Output:** Product successfully uploaded.

4.3.4.2 Update Product Information

- **Description:** Sellers can edit existing product information, including price.
    - **4.3.4.2.1: Select Product to Edit**
        - **Input:** Seller selects the item to be edited.
        - **Output:** Product details are displayed for editing.
    - **4.3.4.2.2: Update Product**
        - **Input:** Seller edits suitable details.
        - **Output:** Product information updated successfully.

4.3.5 Price Negotiation

4.3.5.1 Buyer Offers Price

- **Description:** Buyers can offer their own prices for products, which sellers can accept or reject.
    - **4.3.5.1.1: Submit Offer**
        - **Input:** Buyer submits an offer price for a product.
        - **Output:** Seller receives the offer.
    - **4.3.5.1.2: Accept or Reject Offer**
        - **Input:** Seller reviews the offer and accepts or rejects it.
        - **Output:** Buyer is notified of the seller's decision.

4.3.6 Profile Management

4.3.6.1 View Profile

- **Description:** Users can view their profiles and make changes if desired.
    - **4.3.6.1.: Go to Profile**
        - **State:** User is logged in.
        - **Input:** Click on the profile photo.
        - **Output:** User redirected to the profile page.

4.3.6.2 Edit Profile Information

- **Description:** Users can edit profile information and save changes.
    - **4.3.6.2.1: Select Field to Edit**
        - **Input:** User selects the field to edit.
        - **Output:** User enters new information for the selected field.
    - **4.3.6.2.2: Save Changes**
        - **Input:** Click on the "Update Profile" button.
        - **Output:** User profile updated with a confirmation message.

4.4. Other Non-Functional Requirements

4.4.1 Performance Requirements

The website/app must be fast, interactive and must provide a quick response. In the case of opening applications, forms and saving the settings or sessions there must not be a delay of more than 1 second. On connection to databases, sorting categories and ordering of products there should be no delays and the operation must be performed in less than 1 second for opening, sorting, computing, etc.

4.4.2 Safety Requirements

Only authorised administrators should be able to query and change the contents of the database. The website users should be authenticated before they can use the feature exclusively for them. The usernames and passwords of users should be encrypted and protected from unauthorised access. Data security should be maintained. If the payments get cancelled, there must be provision for credit of the amount in question to the customer.

4.4.3 Security Requirements

- HTTPS protocols are used for communication due to their secure nature.
- Fraud transaction detection systems can be used to combat fake transactions.
- Security plugins provide protection against bad bots, code injections and hundreds of other severe attacks.
- Automatic backup service can be used so that even if data is lost, all data will be backed up automatically. Another option is to choose a managed e-commerce web hosting service that automatically creates backups like Cloudways.

4.4.4 Software Quality Attributes

- Performing Security and Vulnerability Assessments
- Checking Compatibility with Web Browsers
- Testing for Mobile Responsiveness
- Checking Compatibility with Android versions

4.4.5 Business Rules

- Taxes
- Payment Gateways
- Trademark and Copyright
- Shipping Restrictions

5. Database Design

5.1. Models

- **User**: Stores user information, including name, email, password, role (default: user), status (default: active), and profile picture URL.

- **Product**: Stores product details such as name, description, price, category, age, months/years of use, images, and additional details (bill available, warranty available, accessories available, box available, product damage, first owner, scratches, status). It also references the seller (User).

- **Notification**: Stores notifications with a title, message, a URL to be triggered on click, the user it's directed to, and a read status.

- **Bid**: Stores bidding information, including product reference, seller, buyer, bid amount, buyer's mobile number, and message.

5.2. Relationship

- One-to-many: A user can have multiple products, bids, and notifications.

- One-to-many: A product can have multiple bids.

- One-to-one: Each bid is linked to a product, seller, and buyer.

5.3 Schema

- **User**

```
{
  name: { type: String, required: true },
  email: { type: String, required: true, unique: true, trim: true },
  password: { type: String, required: true },
  role: { type: String, default: 'user' },
  status: { type: String, default: 'active' },
  profilePicture: { type: String, default: '' },
  createdAt: { type: Date, default: Date.now },
  updatedAt: { type: Date, default: Date.now }
}
```

- **Bid**

```
{
  product: { type: Schema.Types.ObjectId, ref: 'Product', required: true },
  seller: { type: Schema.Types.ObjectId, ref: 'User', required: true },
  buyer: { type: Schema.Types.ObjectId, ref: 'User', required: true },
  bidAmount: { type: Number, required: true },
  mobile: { type: Number, required: true },
  message: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
}
```

- **Product**

```
{
  name: { type: String, required: true },
  description: { type: String, required: true },
  price: { type: Number, required: true },
  category: { type: String, required: true },
  age: { type: Number, required: true },
  monYears: { type: String, required: true },
  images: { type: [String], required: true },

  billAvailable: { type: Boolean, default: false },
  warrantyAvailable: { type: Boolean, default: false },
  accessoriesAvailable: { type: Boolean, default: false },
  boxAvailable: { type: Boolean, default: false },
  productdamage: { type: Boolean, default: false },
  firstowner: { type: Boolean, default: false },
  scratches: { type: Boolean, default: false },
  status: { type: String, default: 'pending' },

  seller: { type: Schema.Types.ObjectId, ref: 'User', required: true },
  createdAt: { type: Date, default: Date.now },
  updatedAt: { type: Date, default: Date.now }

}
```

- **Notification**

```
{
  title: { type: String, required: true },
  message: { type: String, required: true },
  onClick: { type: String, required: true },
  user: { type: Schema.Types.ObjectId, ref: 'User', required: true },
  read: { type: Boolean, default: false },
  createdAt: { type: Date, default: Date.now },
  updatedAt: { type: Date, default: Date.now }
}
```

6. Implementation Details

6.1. Modules Created and Brief Description

1. **Authentication Module:**
    - Implements JWT-based authentication for users. Provides login, registration, and secure access to protected routes. Ensures that only logged-in users can sell or bid on products.
2. **User Management Module:**
    - Handles user authentication, including login and registration. Also manages user sessions using JWT tokens stored in local storage for secure access to routes like adding products or bidding.
3. **Product Management Module:**
    - Allows users to add, edit, and delete products. Each product is linked to a seller, and users can upload multiple images using cloudinary Storage. Products are displayed on the home page, and users can view detailed product information.
4. **Bidding Module:**
    - Enables users to place bids on products. Users can submit a price and a message to the seller. Sellers can view all bids in their profile page, where they can manage incoming offers for their products.
5. **Search and Filter Module:**
    - Provides functionality to search for products by keywords and filter search results based on product categories, price ranges, and other filters.

6. **Messaging Module:**
    - Allows buyers and sellers to exchange messages regarding bids or other product-related queries. Messaging is initiated when a user places a bid on a product.
7. **Profile Management Module:**
    - Displays user-specific information like listed products and bids received. Sellers can manage products and view bids, while buyers can view their bidding history and product listings.

6.2. Function Prototype:

1. **registerUser(req, res)**
    - **Parameters:**
        - req (Object): The request object containing user data such as name, email, and password.
        - res (Object): The response object used to send data back to the client.
    - **Return Type: Promise<void>**
        - Returns a JSON object with a JWT token upon successful registration or an error message if registration fails.
    - **Description**: This function registers a new user in the system. It hashes the password using bcrypt, saves the user data in the database, and generates a JWT token for secure authentication.

2. **loginUser(req, res)**
   - **Parameters:**
     - req (Object): The request object containing user login data (email, password).
     - res (Object): The response object used to send data back to the client.
   - **Return Type: Promise<void>**
     - Returns a JSON object with a JWT token upon successful login or an error message if login fails.
   - **Description**: This function authenticates the user by checking the email and password. If valid, it generates a JWT token and allows access to protected routes.
3. **addProduct(req, res)**
   - **Parameters:**
     - req (Object): The request object containing product data like title, description, price, and images.
     - res (Object): The response object used to send data back to the client.
   - **Return Type: Promise<void>**
     - Returns a JSON object with success status upon product creation or an error message if the process fails.
   - **Description**: This function allows authenticated users to add products to the platform. Product details are stored in the database, and images are uploaded to Firebase Storage.

4. **placeBid(req, res)**
   - **Parameters**:
     - req (Object): The request object containing bid details such as product ID, bid price, and message.
     - res (Object): The response object used to send data back to the client.
   - **Return Type: Promise<void>**
     - Returns a JSON object confirming the bid submission or an error message if the bid fails.
   - **Description**: This function enables users to place bids on products. The bid price and message are stored in the database and linked to the product and user. Sellers can view all bids in their profile.
5. **filterProducts(req, res)**
   - **Parameters:**
     - req (Object): The request object containing filter criteria like category, price range, etc.
     - res (Object): The response object used to send filtered data back to the client.
   - **Return Type: Promise<void>**
     - Returns a JSON object with the filtered product list.
   - **Description**: This function filters products based on the user's input, such as category, price, or search keywords. The filtered results are then displayed on the home page.

6. **sendMessage(req, res)**
   - **Parameters:**
     - req (Object): The request object containing message content, recipient ID, and product ID.
     - res (Object): The response object used to send data back to the client.
   - **Return Type: Promise<void>**
     - Returns a JSON object with a success message if the message is sent successfully or an error message if it fails.
   - **Description**: This function allows users to send messages related to bids. It stores the messages in the database and ensures both buyers and sellers can view the messages in their profile.
7. **viewBids(req, res)**
   - **Parameters:**
     - req (Object): The request object containing the seller’s user ID.
     - res (Object): The response object used to send bid data back to the client.
   - **Return Type: Promise<void>**
     - Returns a JSON object with a list of bids on the seller's products.
   - **Description**: This function retrieves all bids placed on the products listed by the seller. The bids are displayed in the seller’s profile for review.

7. Testing

7.1 Testing Framework and Methodology:

- **Unit Testing**: Frontend forms, navigation, and responsiveness across multiple browsers.
- **Endpoint Testing**: Postman for API endpoints.

Table 7.1 (Test Cases)

Test Case ID	Description	Test Steps	Input	Expected Output	Actual Output	Status
TC01	Verify user registration and login functionality	1. Navigate to the registration page. 2. Enter valid details. 3. Click the "Register" button. 4. Log in.	Valid user details (email, password)	User registered and logged in successfully.	User registered and logged in successfully.	Pass
TC02	Test product listing on the home page	1. Log in. 2. Navigate to the home page.	N/A	Products listed on the home page.	Products displayed correctly on the home page.	Pass

TC03	Verify product search functionality	1. Log in. 2. Search for a product using the search bar.	Product keyword	Products matching the search query displayed.	Search results displayed correctly.	Pass
TC04	Test filter functionality for products	1. Log in. 2. Apply filters (e.g., category, price range) on the home page.	Filter options (category, price)	Products filtered as per the applied filters.	Products filtered correctly.	Pass
TC05	Verify adding a product via profile page	1. Log in. 2. Navigate to the profile page. 3. Click "Add Product" and fill the form.	Product details (name, price, etc.)	Product added and displayed in the profile and on the home page.	Product added and displayed correctly.	Pass
TC06	Validate bidding on a product	1. Log in. 2. View a product. 3. Enter bid amount and message. 4. Submit bid.	Bid amount and message	Bid submitted and visible in the seller's profile.	Bid submitted and visible to the seller.	Pass
TC07	Verify viewing bids in seller's profile	1. Log in as a seller. 2. Navigate to the profile page. 3. View bids on listed products.	N/A	All bids on seller's products displayed.	Bids displayed correctly in seller's profile.	Pass

TC08	Test for authentication required pages	1. Try accessing a restricted page (e.g., Add Product) without logging in.	N/A	Access denied, redirect to login page.	Access denied and user redirected to login page.	Pass
TC09	Test product detail viewing functionality	1. Log in. 2. Click on a product from the home page.	N/A	Product details page displayed.	Product details displayed correctly.	Pass
TC10	Test logout functionality	1. Log in. 2. Navigate to the profile page. 3. Click "Logout".	N/A	User logged out successfully and redirected to the login page.	User logged out and redirected to the login page.	Pass

8. Screenshots

- **User registration and login page**

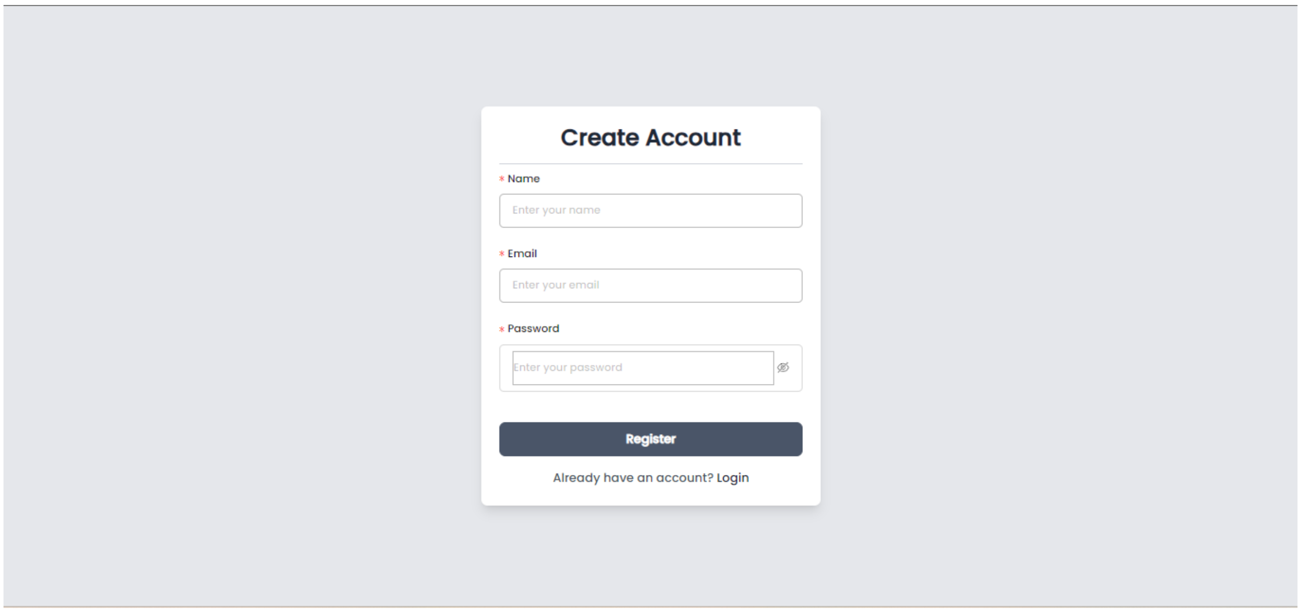


Figure 8.1 (Registration page)

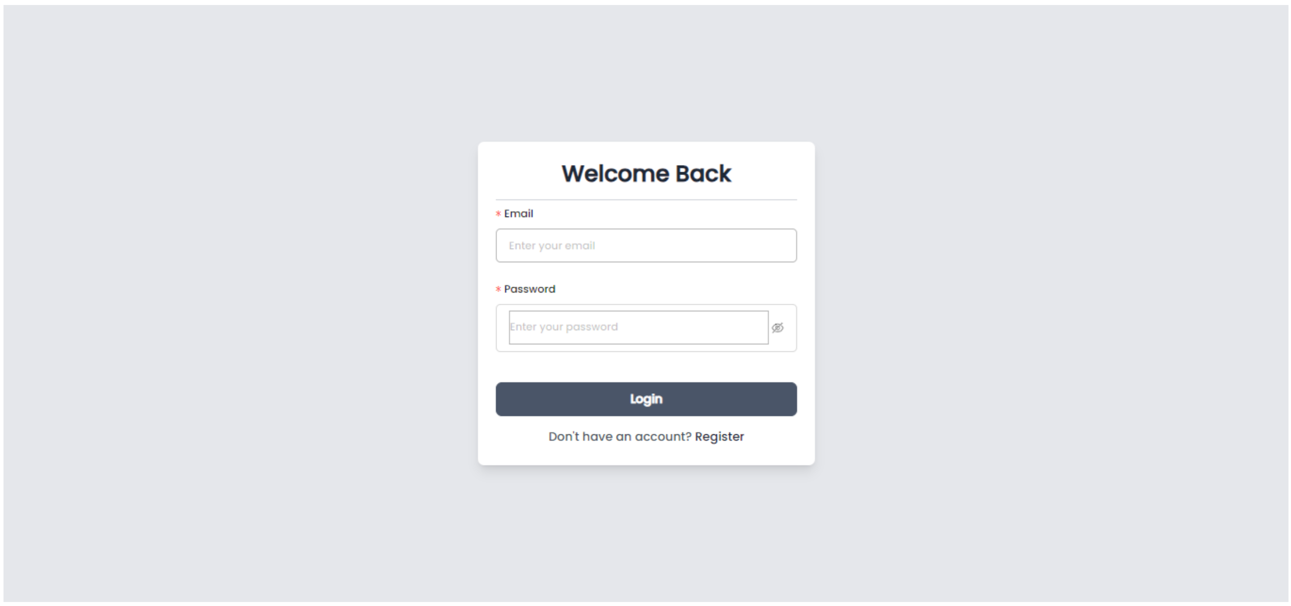
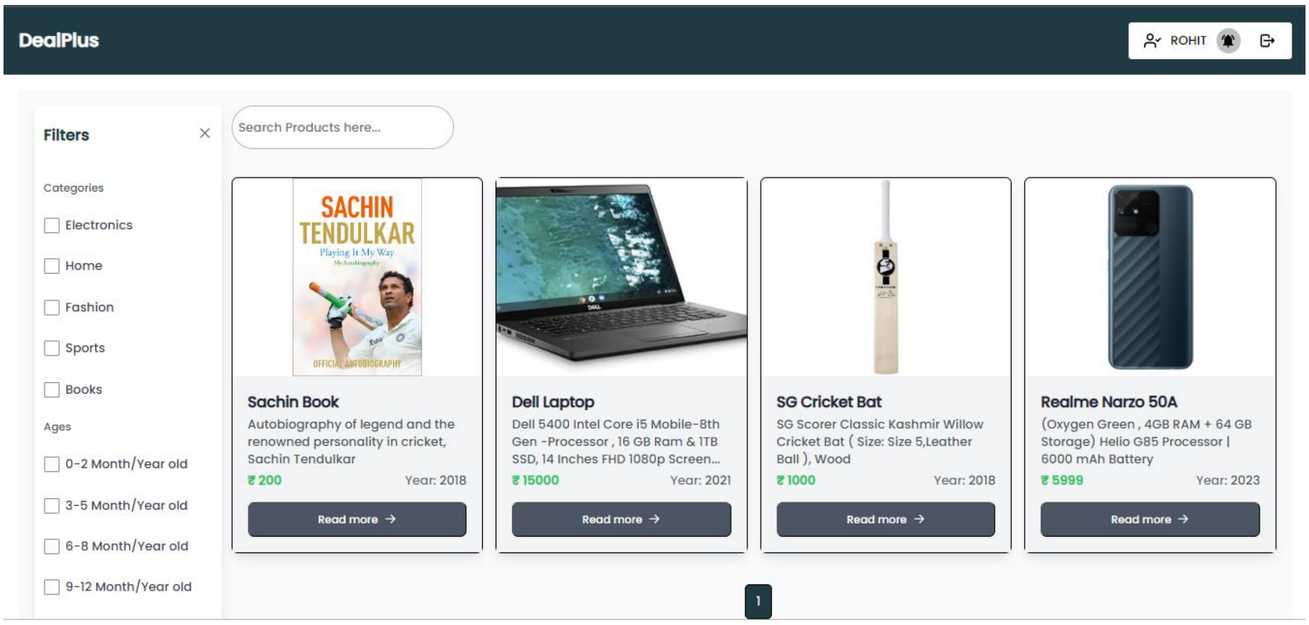


Figure 8.2 (Login page)

- **Dashboard pages**



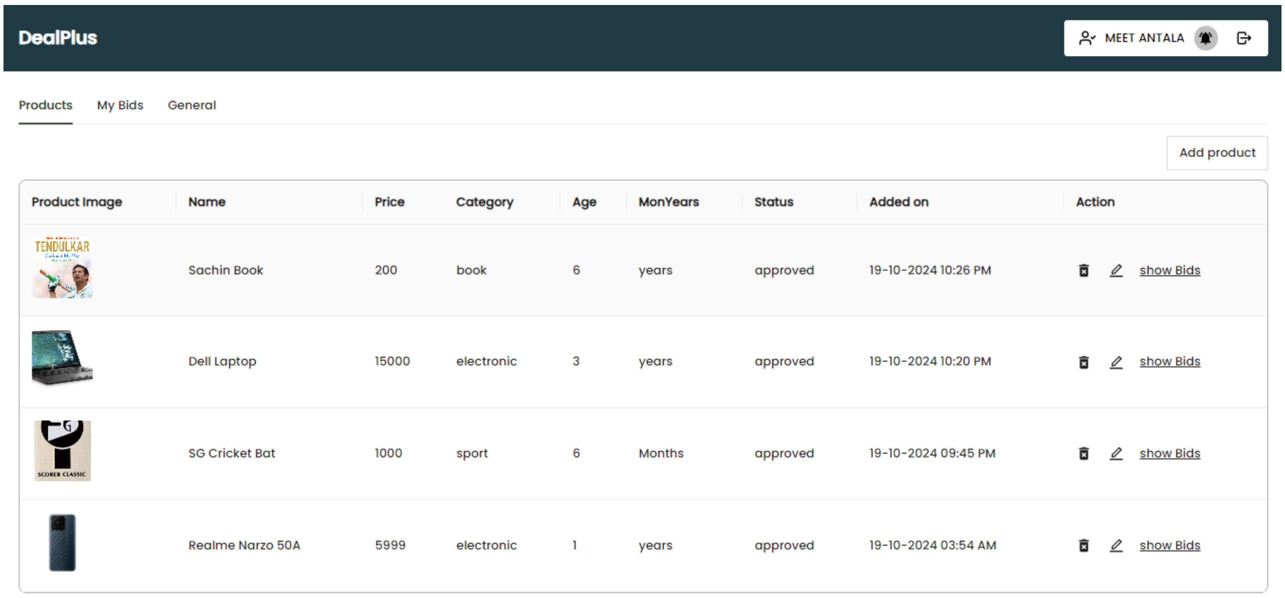
Sachin Book
Autobiography of legend and the renowned personality in cricket, Sachin Tendulkar
₹ 200 Year: 2018

Dell Laptop
Dell 5400 Intel Core i5 Mobile-8th Gen -Processor , 16 GB Ram & 1TB SSD, 14 Inches FHD 1080p Screen...
₹ 15000 Year: 2021

SG Cricket Bat
SG Scorer Classic Kashmir Willow Cricket Bat (Size: Size 5,Leather Ball), Wood
₹ 1000 Year: 2018

Realme Narzo 50A
(Oxygen Green , 4GB RAM + 64 GB Storage) Helio G85 Processor | 6000 mAh Battery
₹ 5999 Year: 2023

Figure 8.3 (Home page)



Product Image	Name	Price	Category	Age	MonYears	Status	Added on	Action
	Sachin Book	200	book	6	years	approved	19-10-2024 10:26 PM	show Bids
	Dell Laptop	15000	electronic	3	years	approved	19-10-2024 10:20 PM	show Bids
	SG Cricket Bat	1000	sport	6	Months	approved	19-10-2024 09:45 PM	show Bids
	Realme Narzo 50A	5999	electronic	1	years	approved	19-10-2024 03:54 AM	show Bids

Figure 8.4 (Profile dashboard)

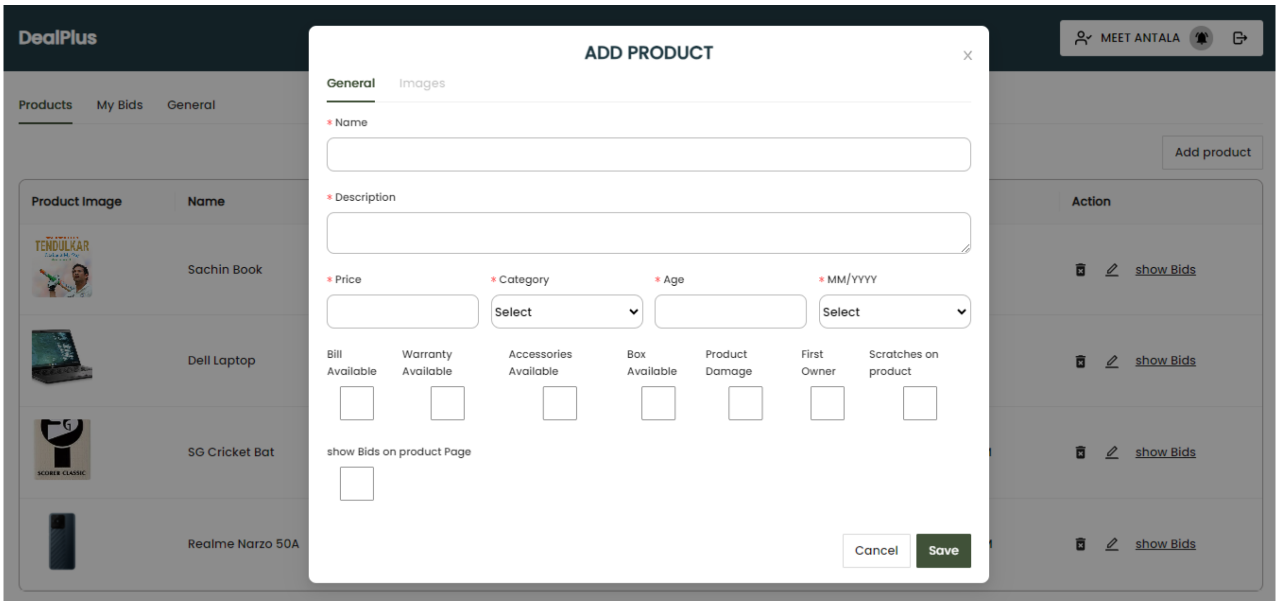
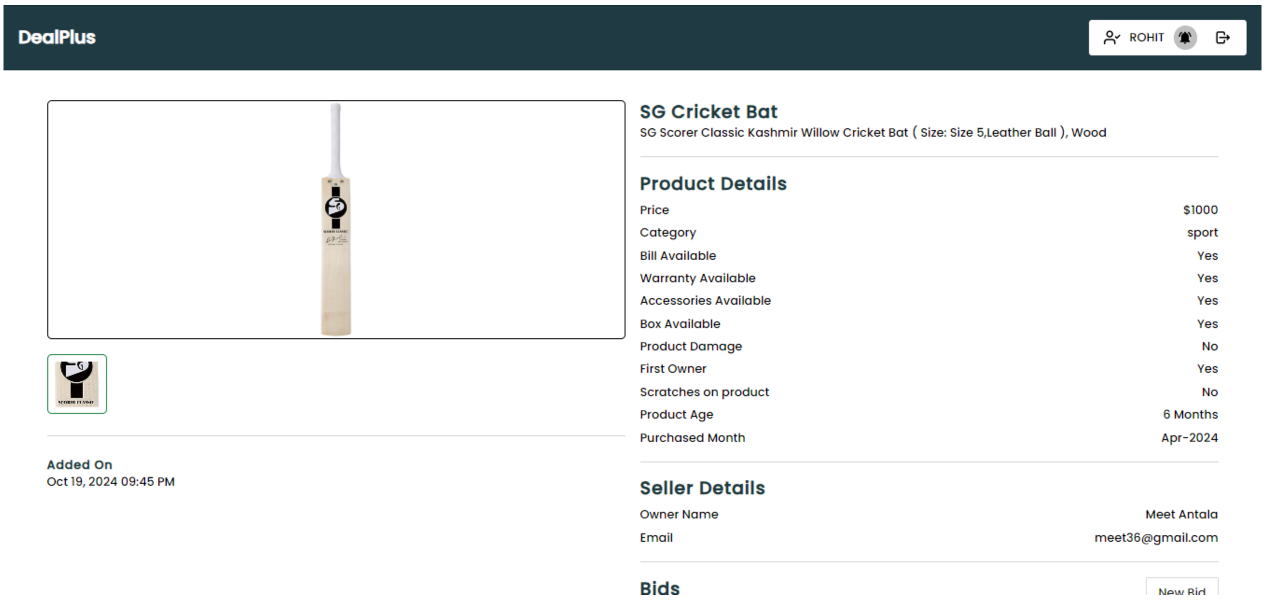


Figure 8.5 (Add Product Page)



Added On
Oct 19, 2024 09:45 PM

SG Cricket Bat
SG Scorer Classic Kashmir Willow Cricket Bat (Size: Size 5,Leather Ball), Wood

Product Details

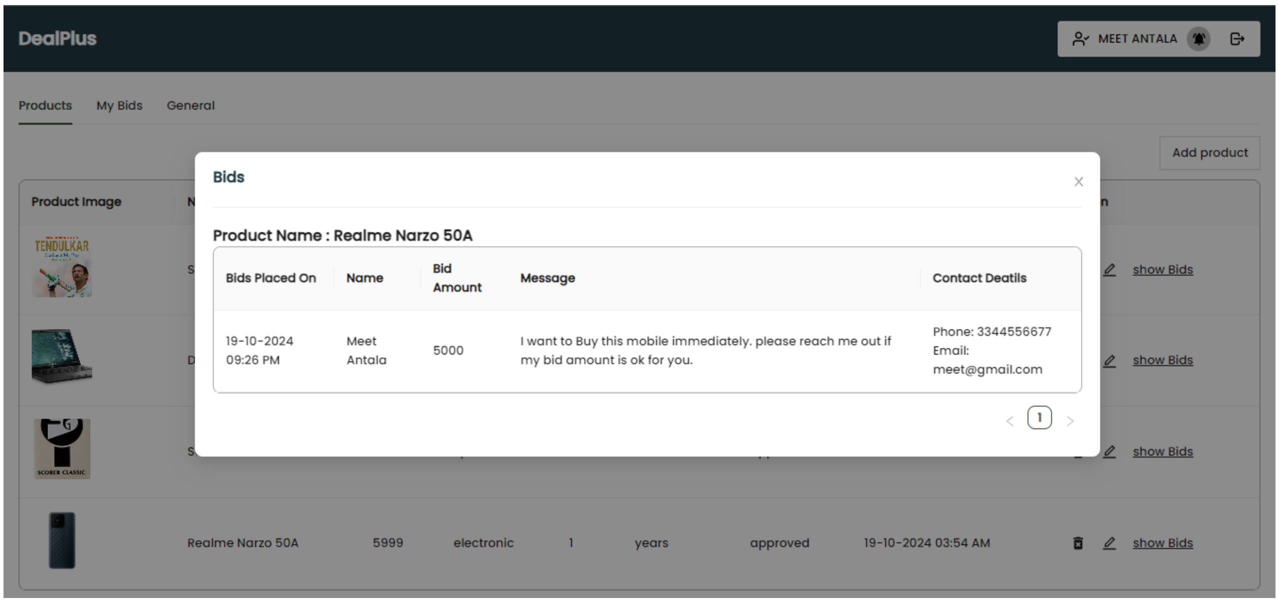
Price	$1000
Category	sport
Bill Available	Yes
Warranty Available	Yes
Accessories Available	Yes
Box Available	Yes
Product Damage	No
First Owner	Yes
Scratches on product	No
Product Age	6 Months
Purchased Month	Apr-2024

Seller Details

Owner Name	Meet Antala
Email	meet36@gmail.com

Bids

Figure 8.6 (Product Details)



Bids

Product Name : Realme Narzo 50A

Bids Placed On	Name	Bid Amount	Message	Contact Deatils
19-10-2024 09:26 PM	Meet Antala	5000	I want to Buy this mobile immediately. please reach me out if my bid amount is ok for you.	Phone: 3344556677 Email: meet@gmail.com

Realme Narzo 50A	5999	electronic	1	years	approved	19-10-2024 03:54 AM	show Bids

Figure 8.7 (Bids page)

9. Conclusion

- The **DealPlus** platform provides a robust solution for users to buy and sell products in a secure and efficient manner.
- **JWT-based authentication** ensures that users' actions are secure, with users needing to log in to access functionalities such as bidding, posting products, and chatting with sellers.
- Authenticated users can sell items by adding product details and uploading images, all stored securely in cloudinary Storage.
- The platform supports **bidding** on products, allowing buyers to place bids with messages, and sellers can view all bids on their products through their profile.
- **Sellers** have full control over their listings, with the ability to add, edit, and manage products they are selling.
- **Notification** helps by notifying users about important activities such as bids on their products.
- The system has been carefully tested to ensure smooth operation of essential features like product management, bidding, user registration, and profile updates, providing a user-friendly experience.

10. Limitations and future extension

10.1 Limitations

- Lack of real-time features like live chat for seamless communication between buyers and sellers.
- Limited filtering options for users to narrow down product searches based on advanced parameters.
- Absence of detailed analytics for sellers to track the performance of their products, such as views or bid trends.

10.2 Future Extensions

- Add real-time chat functionality, allowing buyers and sellers to communicate instantly within the platform.
- Introduce advanced filtering options, such as price range, location, and product condition, for improved search capabilities.
- Enhance seller analytics, providing detailed insights into product views, bid history, and sales performance.
- Implement a mobile-friendly UI/UX to make the platform more responsive and user-friendly across all devices.

11. Bibliography

MERN Stack Documentation

- MongoDB: MongoDB Official Documentation
- Express.js: Express.js Official Documentation
- React.js: React.js Official Documentation
- Node.js: Node.js Official Documentation