Phase 4 SWE

Project: Click Store

Teamwork:

Rahma Mourad 202210407

Gehad Mahmoud 202201211

Rahma Ibrahim 202201689

Layla Zayed 202201906

Mariam Mohamed 202201223

Eng. Omar Abdalla

Dr. Yousef Ghates

Project Management

We followed the Scrum methodology by organizing our work into sprints, each lasting one week. Below is a breakdown of our sprint structure and key activities for each meeting:

Sprint 1: Task Division and Initial Planning

- Meeting Time: Immediately after delivering Phase 3 (Offline Meeting).
- objective:
    - Divide tasks for the project into smaller, manageable parts
    - Assign each team member specific tasks for the first sprint
- outcome:
    - Clear task assignments were made for each team member
    - We set a one-week deadline to complete the initial tasks

Sprint 2: Progress Check and Problem Solving

- Meeting Time: One week after Sprint 1 (Online Meeting)
- objective:
    - Review progress made by each team member
    - Discuss any challenges or roadblocks faced during the first sprint
- outcome:
    - Team members shared updates on their tasks
    - Issues faced were addressed collectively, and solutions were implemented
    - Everyone agreed on the next steps and updated their task priorities

Sprint 3: Task Completion and Code Integration

- Meeting Time: One week after Sprint 2 (Online Meeting)
- objective:
    - Ensure that all assigned tasks were completed
    - Prepare for code integration using GitHub
- outcome:
    - Each member finalized their part and created a branch on GitHub for their work
    - we reviewed the code for errors and conflicts, preparing it for merging
    - clear strategy for code integration and testing was agreed upon

Final Sprint: Deliverables Preparation

- meeting Time: One week after Sprint 3 (online meeting)
- objective:
    - Divide and work on final deliverables, including the report, presentation, video, UML diagrams, and deployment
- outcome:
    - Tasks were distributed based on individual strengths and availability
    - Everyone committed to completing their assigned deliverables before the final deadline

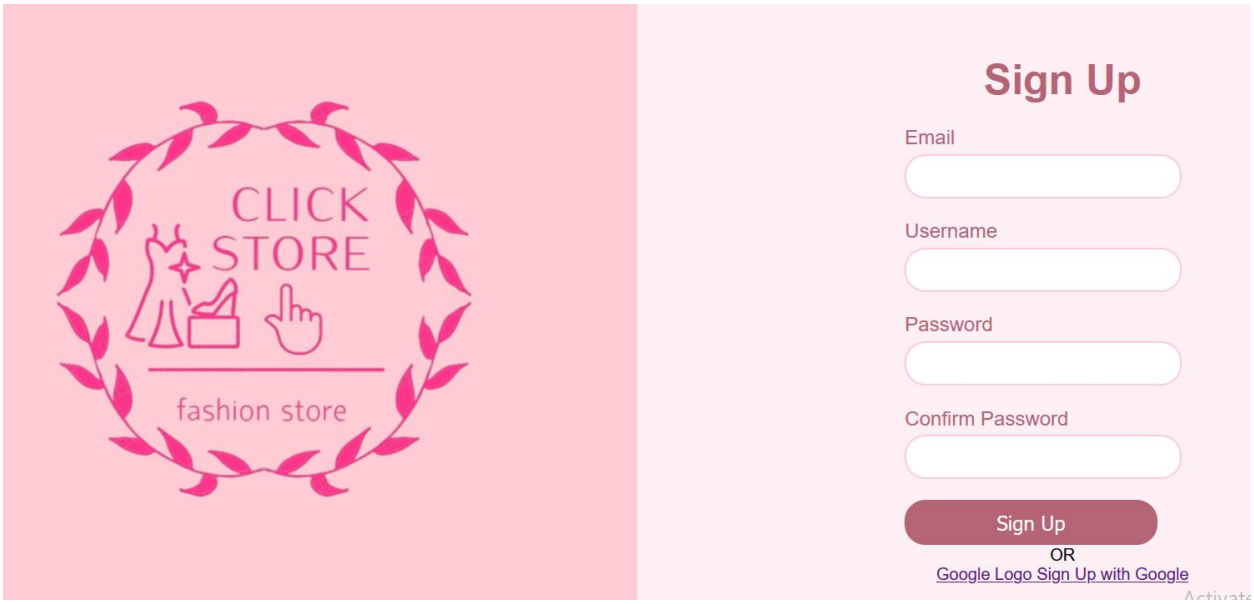
Summary of Sprint Outcomes

- Each sprint helped us stay organized and on track
- GitHub was used effectively for code management, allowing seamless integration and collaboration.
- Regular meetings ensured clear communication, timely progress, and a structured workflow.
- All deliveries were prepared efficiently through focused teamwork

Identify the Product Backlog Items (PBI) and assign them to sprints.

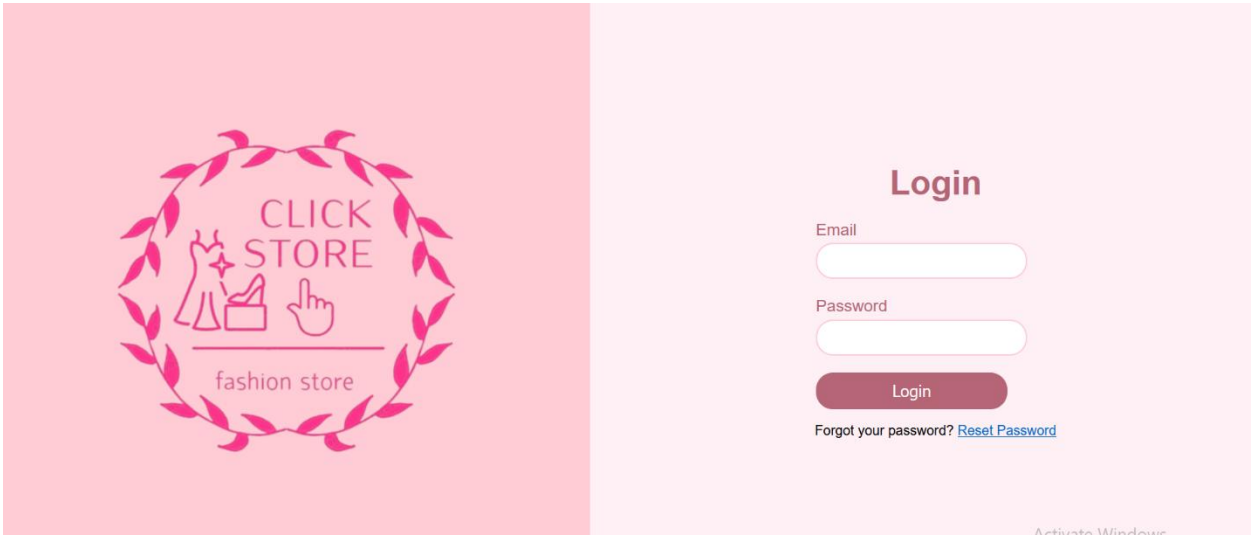
User Registration and Authentication

1-**Sign up:** The user can create an account by entering a username, email, and password to start using the platform

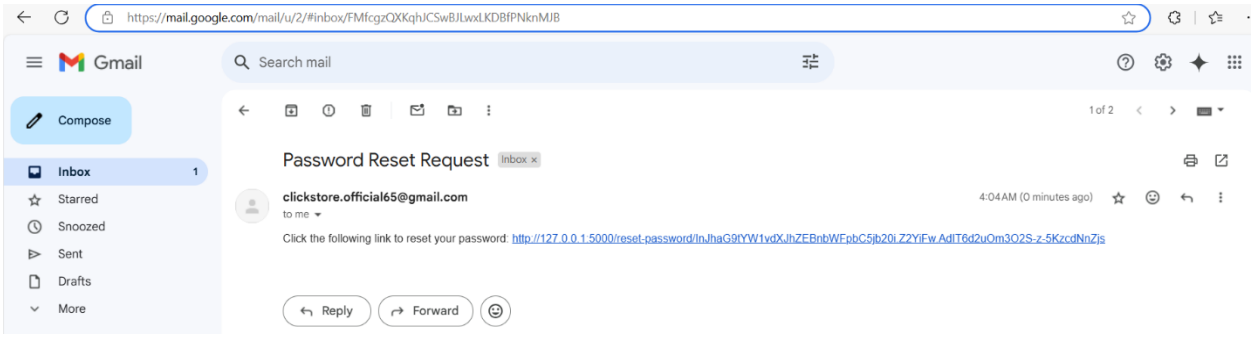


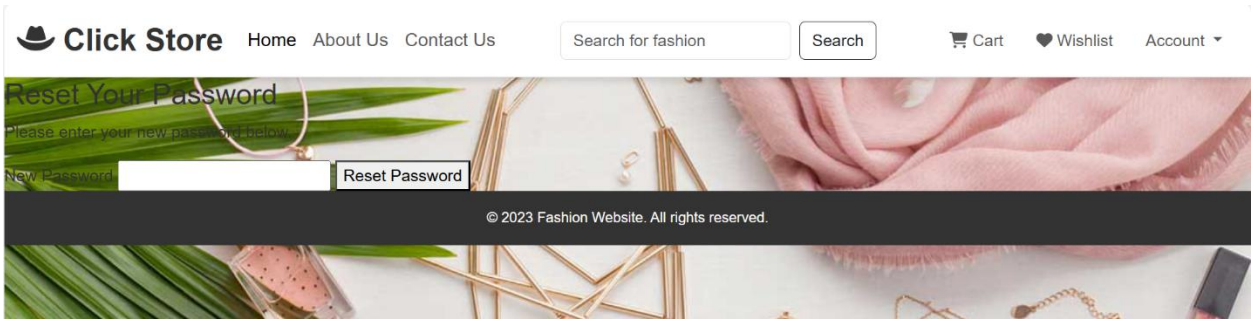
2-**OAuth Authorization:** users can sign in using their Google accounts

3-**Login**: registered user can log in with their username/password to access their account

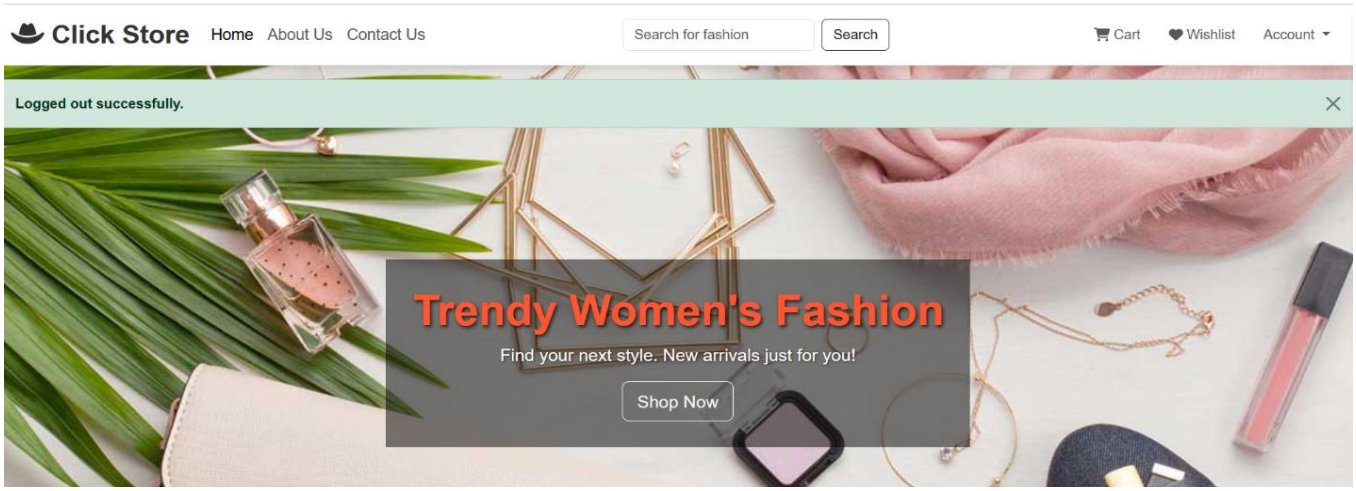


4-**Reset Password**: users can change their password if they want or if they forget their current password. The user receives an email with a verification link to confirm the user's email address from our brand's official email clickstore.official65@gmail.com to make a new password.

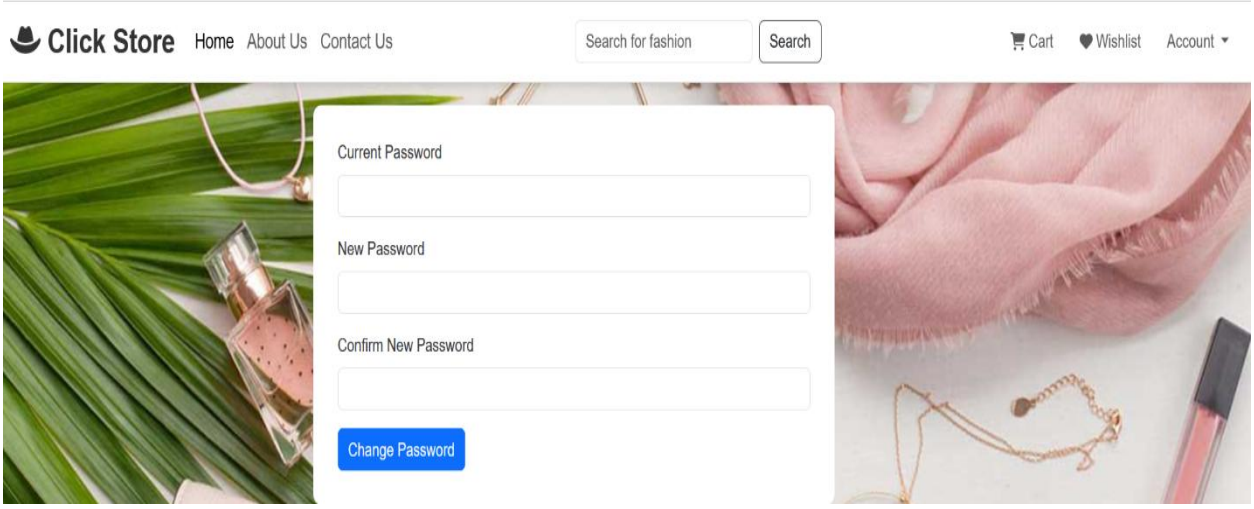




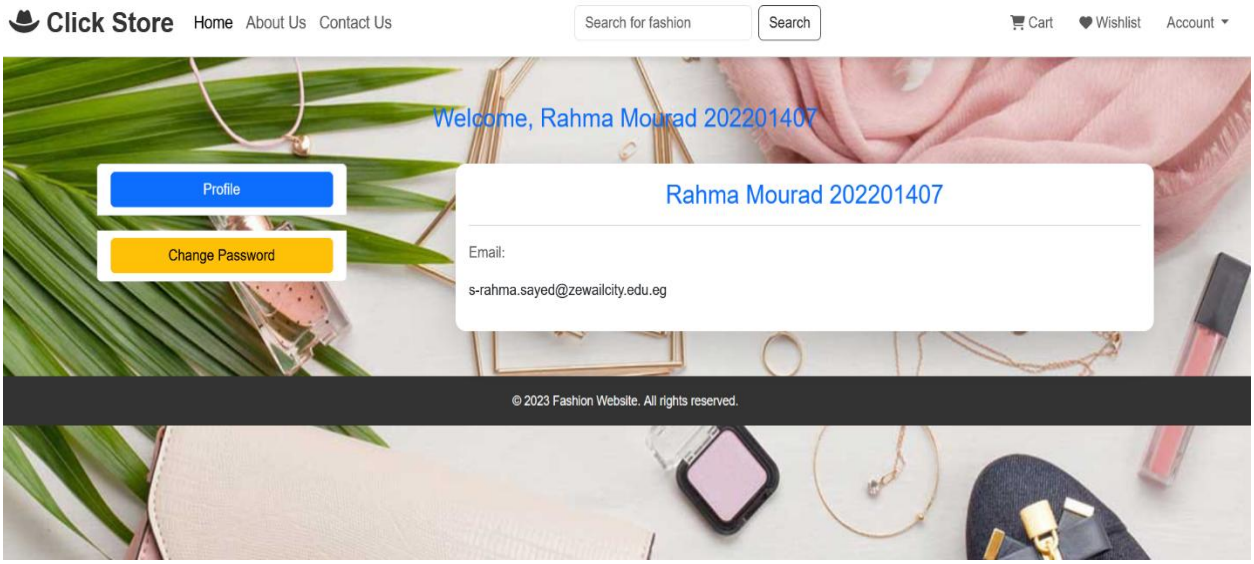
5-**Logout:** The user can to logout from his account



6-**Change password:** After registration, the user can change the password to a new password if he remembers the current password.

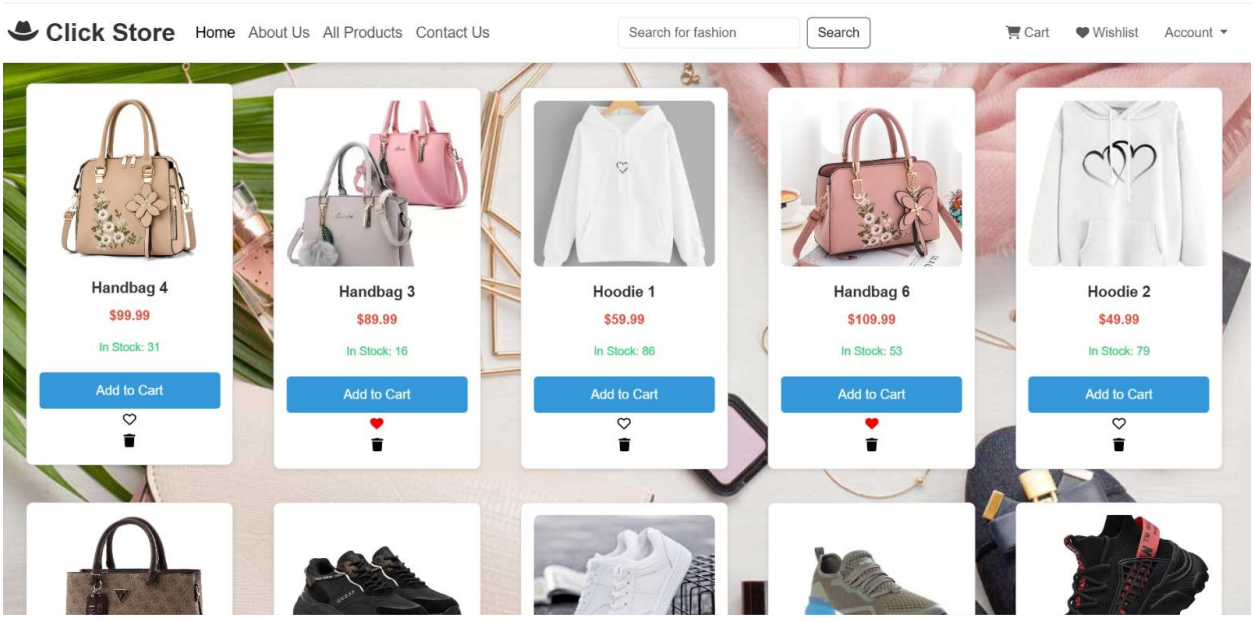


7-**Use profile:** represents all information about the user ‘s data



Order management

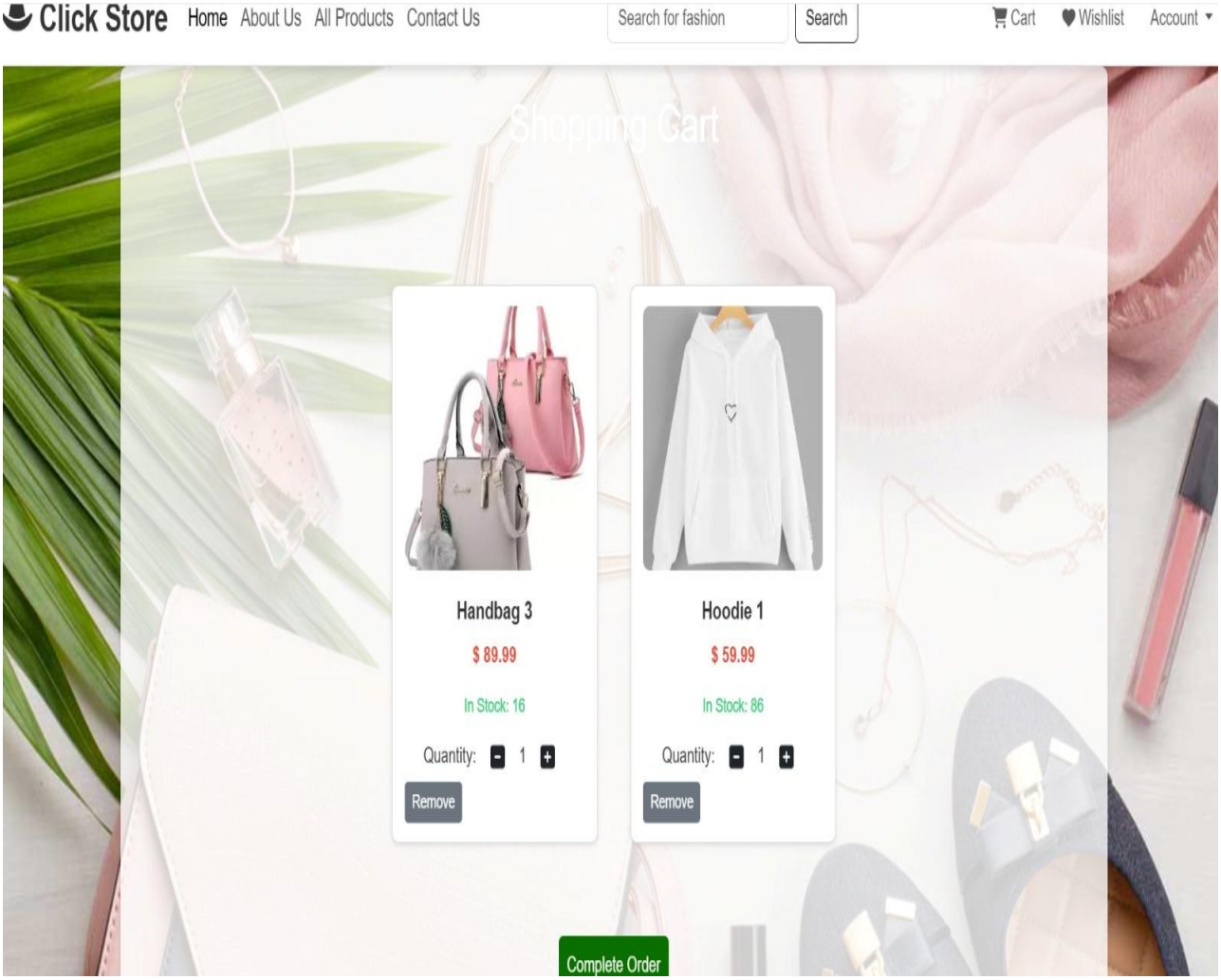
8-**Add Item to Cart:** The user can add products to the cart while browsing.



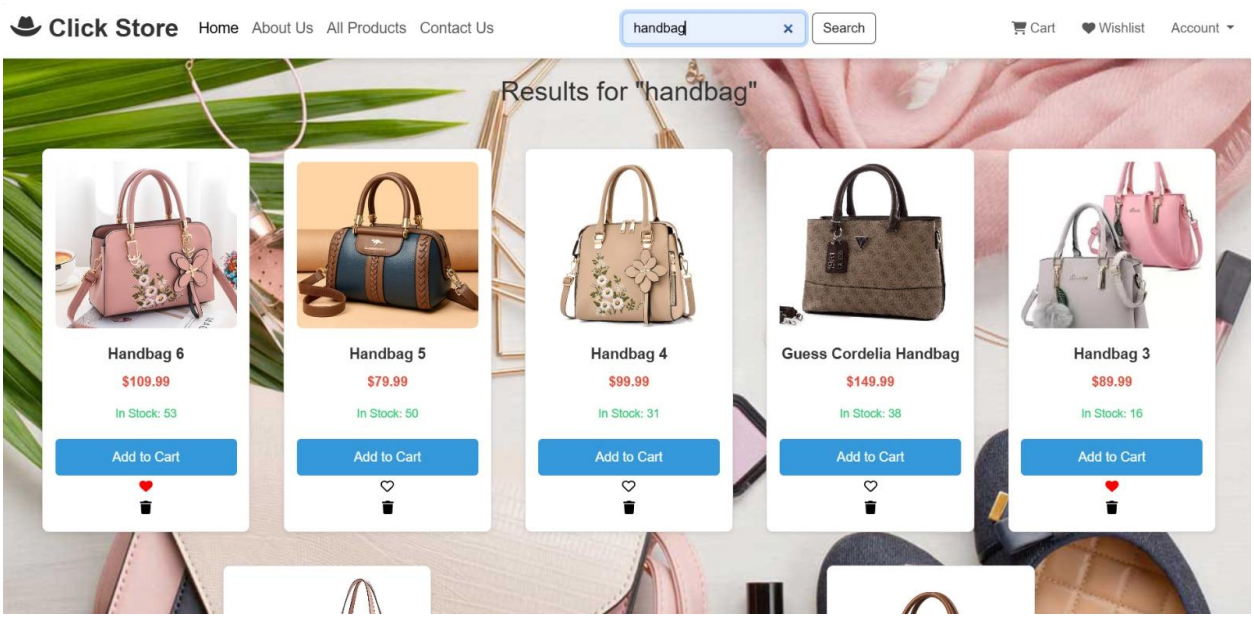
9-**Remove Item from Cart**: users can remove a product from the cart if they decide not to buy.

10-**minus quantities**: users can minus quantity by clicking on the minus icon

11-**plus quantities**: users can add quantity by clicking on the plus icon

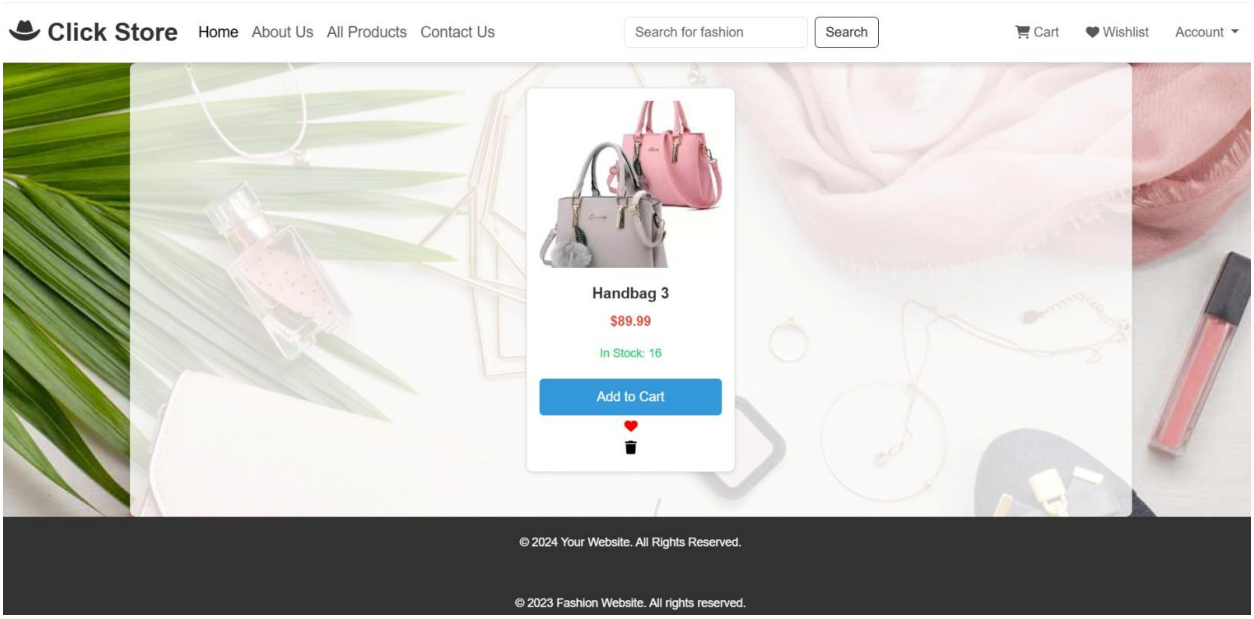


12-**product Search**: The user can search for products using a search bar

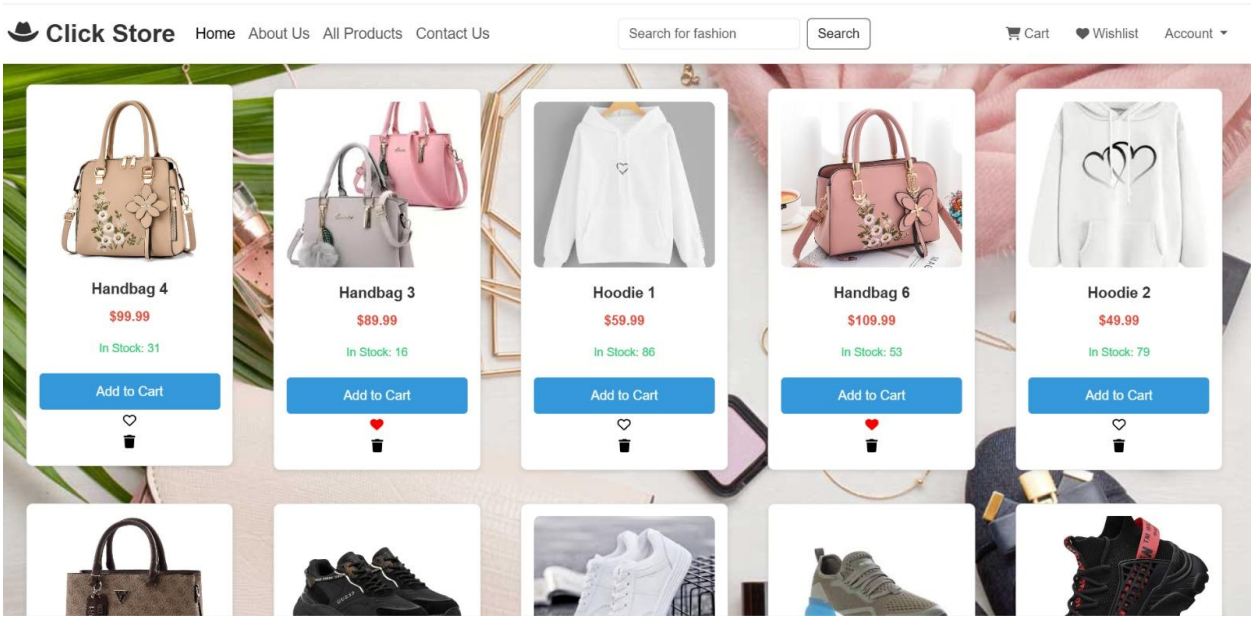


13-**Add to Wishlist**: The user can add products to the Wishlist to come back to them later

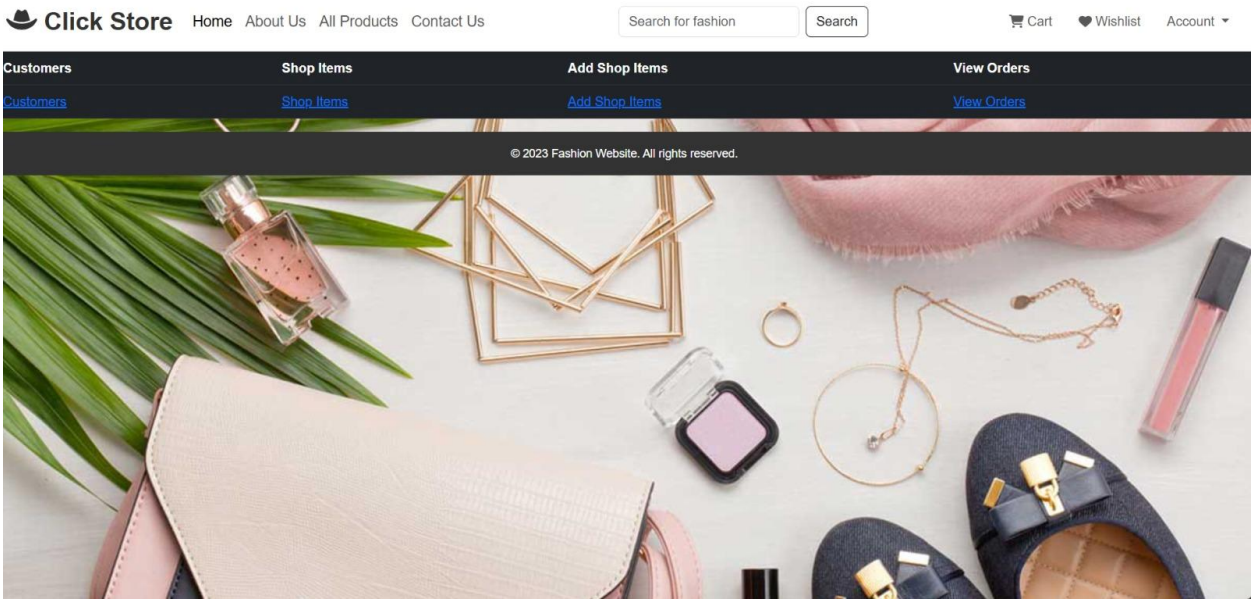
14-**Remove from Wishlist**: The user can remove the product from wishlist



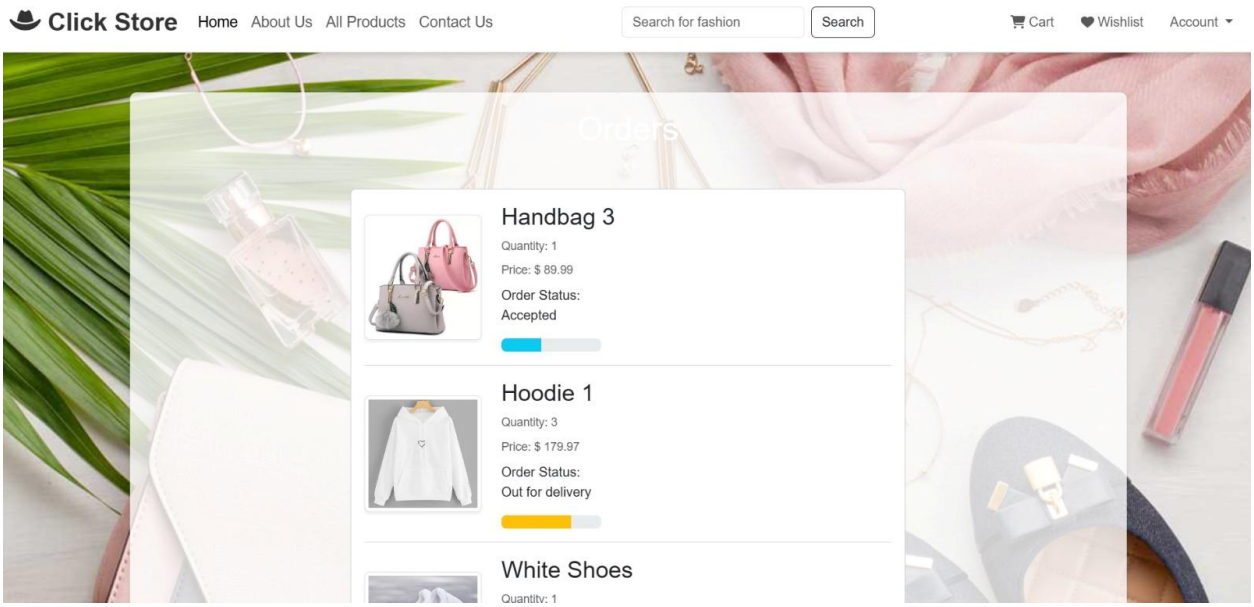
15-**View product:** users after login to the click store website can view all products available on the website



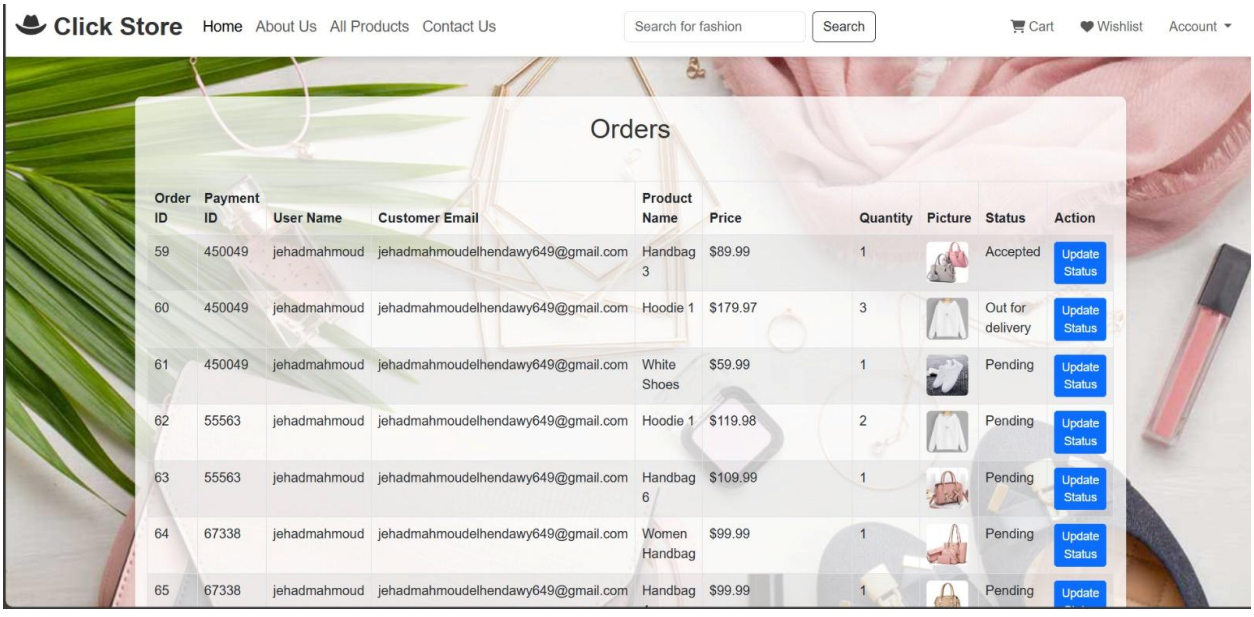
16-Admin dashboard



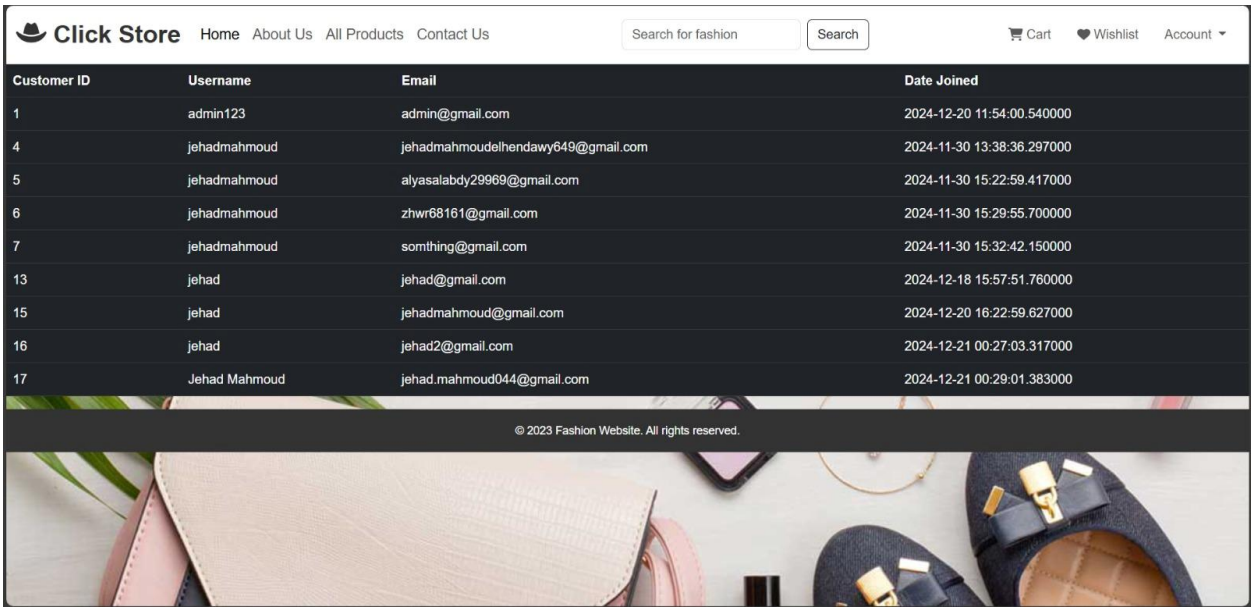
17-**View order status for admin**: The admin shows the order status (Pending - Accepted -Out for Delivery - Delivered-Canceled) with the option to edit it if needed.



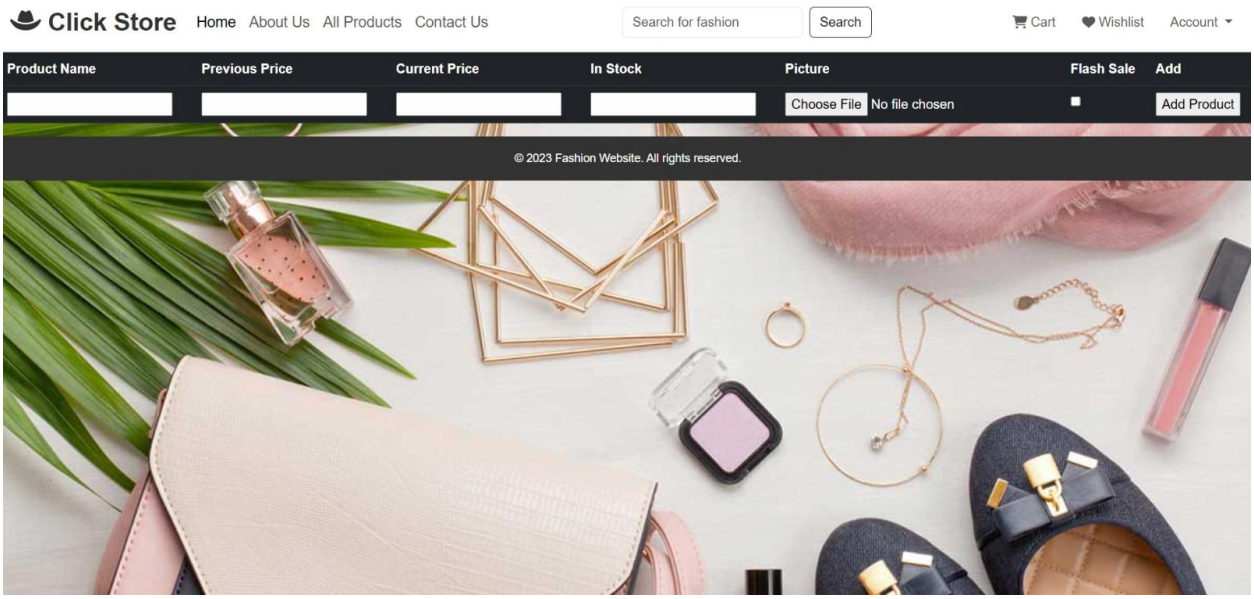
18-**Display order:** The admin can see the details of each order, like the products bought, quantities, and prices.



19- **Admin view**: admin has a page containing all customers who login the website with their data

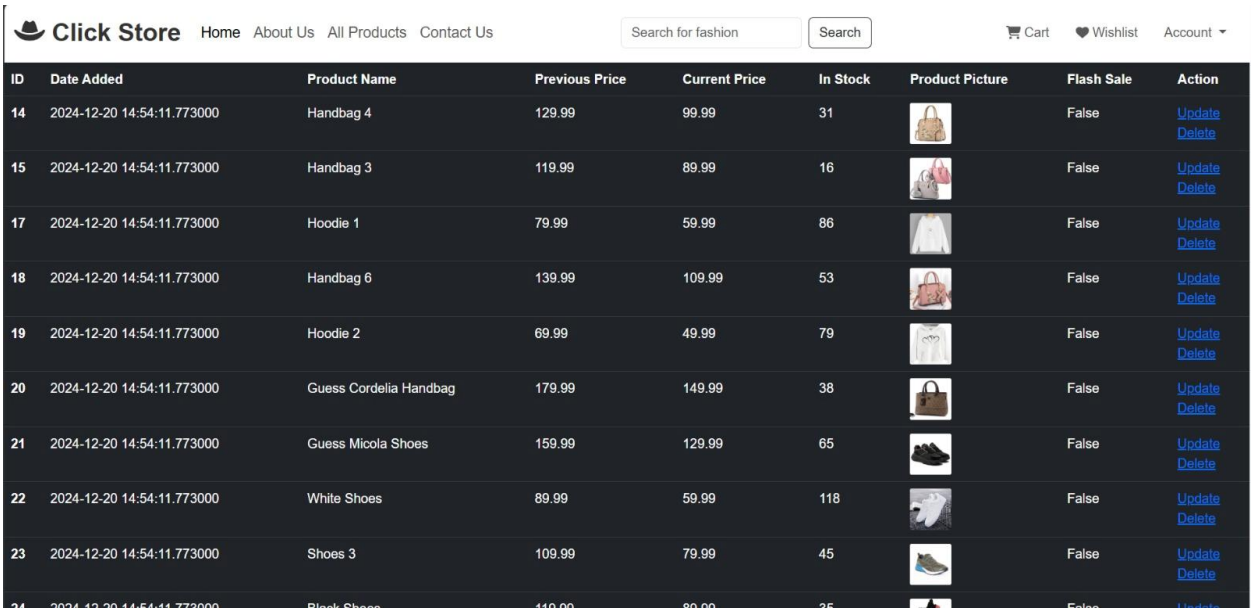


20-**Add products**: The admin will add products to the platform



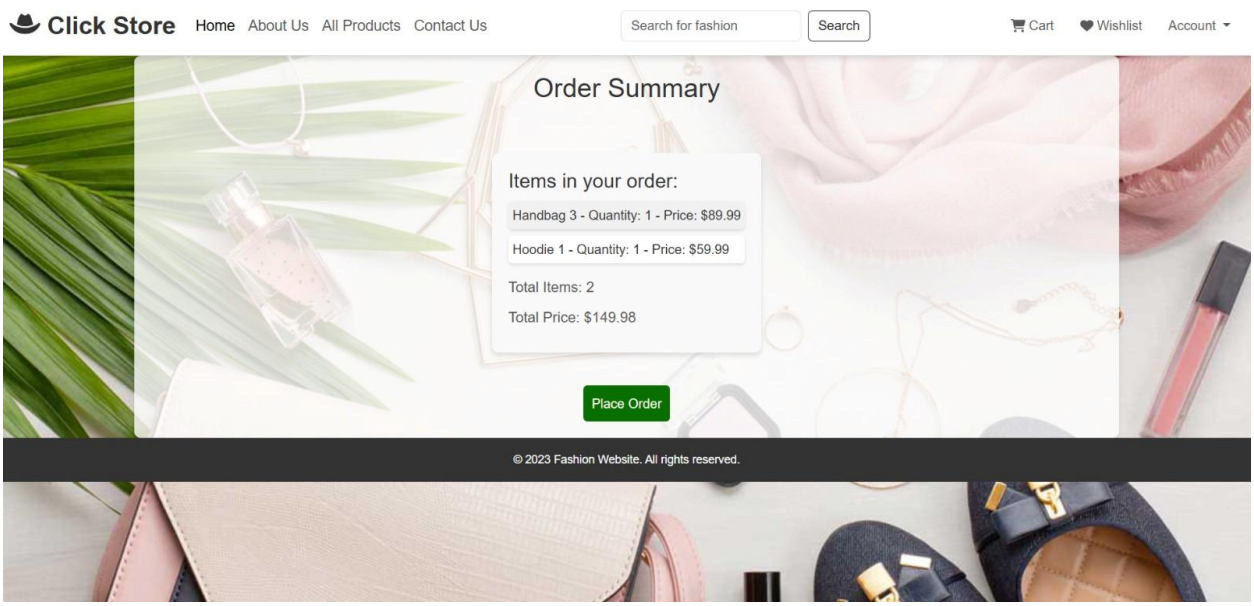
21-**Delete products:** the admin will remove products from the platform.

22-**Update product:** the admin will update products from the platform.

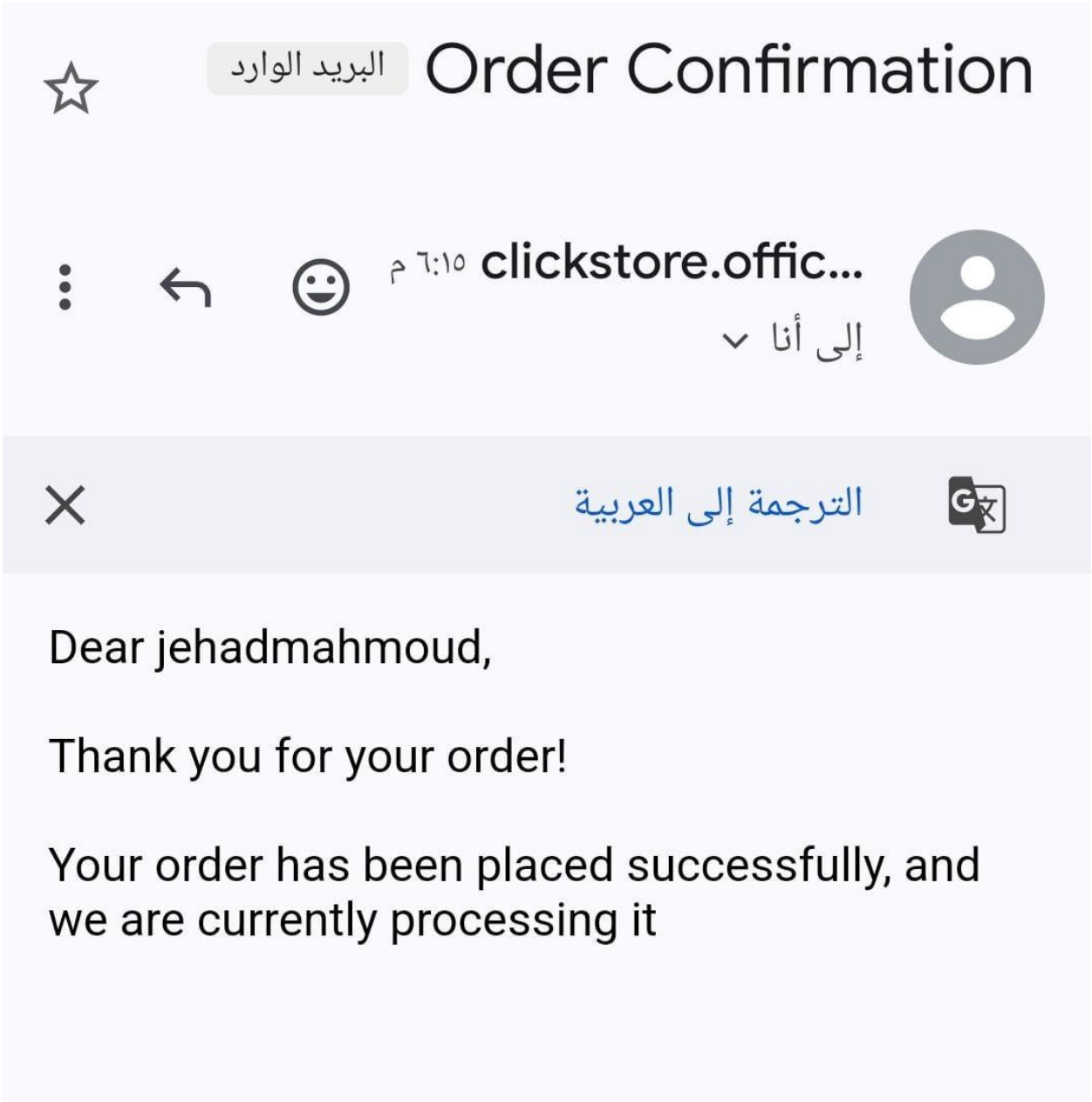


ID	Date Added	Product Name	Previous Price	Current Price	In Stock	Product Picture	Flash Sale	Action
14	2024-12-20 14:54:11.773000	Handbag 4	129.99	99.99	31		False	Update Delete
15	2024-12-20 14:54:11.773000	Handbag 3	119.99	89.99	16		False	Update Delete
17	2024-12-20 14:54:11.773000	Hoodie 1	79.99	59.99	86		False	Update Delete
18	2024-12-20 14:54:11.773000	Handbag 6	139.99	109.99	53		False	Update Delete
19	2024-12-20 14:54:11.773000	Hoodie 2	69.99	49.99	79		False	Update Delete
20	2024-12-20 14:54:11.773000	Guess Cordelia Handbag	179.99	149.99	38		False	Update Delete
21	2024-12-20 14:54:11.773000	Guess Micola Shoes	159.99	129.99	65		False	Update Delete
22	2024-12-20 14:54:11.773000	White Shoes	89.99	59.99	118		False	Update Delete
23	2024-12-20 14:54:11.773000	Shoes 3	109.99	79.99	45		False	Update Delete

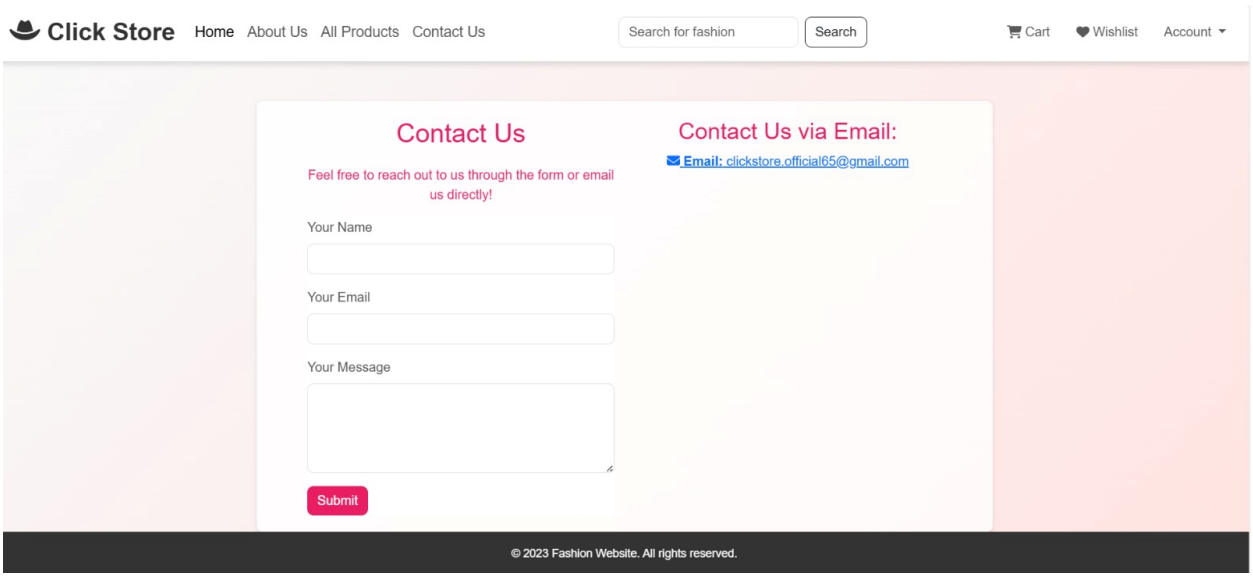
23-**Order Summary**: The customer Shows a summary of the order that includes the product quantities and prices before completing the payment.

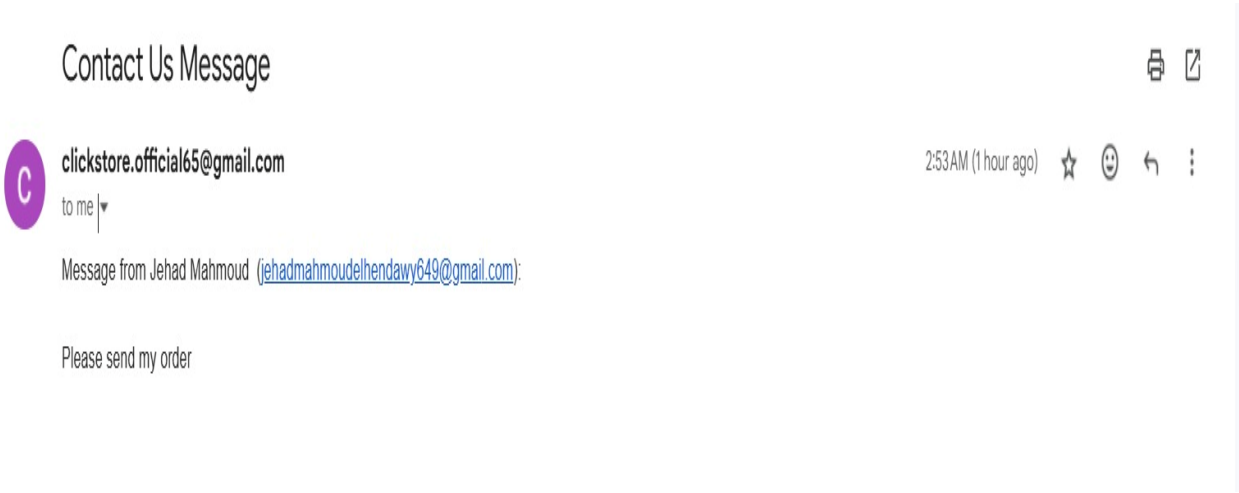


24-**Order confirmation:** user confirms order summary state



25-**contact us**: The user can write a message or feedback for us and the backend with added by form will update in database and with email verification.





Challenges Faced

1. **Database Issues:**
    - We faced challenges ensuring that data was properly saved and synchronized in the database. Sometimes the back end failed to

handle the data as expected, which caused delays in functionality. Debugging these issues took additional time and effort.

2. **Code Integration:**
    - Each team member worked on different features, and merging these into a single project was complex. We encountered errors and compatibility issues when trying to make all features work together seamlessly.
3. **Time Management and Deadlines:**
    - Meeting deadlines while working on multiple project submissions was a significant challenge. We struggled with allocating enough time to complete each feature properly within the set timeframe.
4. **Feature Assignment:**
    - Ensuring that each team member worked on a distinct and complete feature was difficult. There were times when the assigned features were too different or disconnected, which caused inefficiencies.

Solutions to Challenges

1. **Debugging and Team Collaboration:**
    - For database issues, we focused on debugging individually. When problems persisted, we held team meetings to analyze the errors together. Collaborative efforts helped us find effective solutions more quickly.

2. **Using GitHub Effectively:**
    - To manage code integration, we used GitHub to create separate branches for each feature. Once a feature was completed, it was

merged into the main project. This approach minimized errors and made merging smoother.

3. **Time Allocation:**
    - To tackle time management challenges, we set specific deadlines for completing each feature. By breaking down tasks into smaller parts and assigning fixed timeframes, we stayed on track with our project goals.
4. **Feature Grouping:**
    - To improve feature assignment, we reorganized tasks so that team members worked on features that were similar in functionality or implementation. This made it easier to understand the work and reduced inconsistencies.

Improvements for the Future

1. **Stronger Time Management:**
    - Continue to set clear deadlines for each feature and monitor progress regularly to ensure timely completion.
2. **Improved Task Assignment:**
    - Group related features together when assigning tasks to team members, making the work more logical and manageable.
3. **Enhanced Testing Protocols:**
    - Test each feature thoroughly before merging it with the main project to minimize errors during integration.
4. **Better Use of Tools:**
    - Utilize project management tools like Trello to organize tasks and track deadlines more effectively

Explain two design patterns

1. Factory Pattern

Why we used it:
The Factory Pattern helps in creating different types of objects without needing to know their exact class. For example, during user registration, we used a factory to create different types of users, like regular users, admins, or users signing in with login

How it works:
Instead of writing separate code for each user type every time, we used a factory class that takes input (like user type) and returns the correct user object. This makes the system easier to update. For example, if we need to add a new user type, we can extend the factory without changing the rest of the code

2. Observer Pattern

Why we used it:
The observer Pattern allows one part of the system to notify others automatically when something changes. We used this for Order Management, especially for updating the order status.

How it works:
When an admin updates the order status (e.g., from “Pending” to “Delivered”), all connected components, like the user interface, notifications, and admin dashboard, are automatically updated. This makes the system easier to manage and extend. We can do it without changing the existing code if we need to add a new feature (like a service to send analytics about orders).

Why These Patterns?

- **Factory Pattern** ensures easy scalability for user-related features.
- **Observer Pattern** keeps the system synchronized and allows us to add new components effortlessly.