Test Document

for

Campus Craves

Version 0.1

Prepared by

Group 19 **Group Name: Supervised Learners**

Name	Roll no.	E-mail
Aayush Gautam	220020	aayushg22@iitk.ac.in
Abhishek Kumar	210040	abhikumar21@iitk.ac.in
Anand Chutani	220130	anandc22@iitk.ac.in
Anany Dev Choudhary	220135	ananydc22@iitk.ac.in
Chatla Sowmya Sri	200293	srisowmya20@iitk.ac.in
Lokesh Mehra	220591	lokeshm22@iitk.ac.in
Pranshu Mani Tripathi	220800	pranshumt22@iitk.ac.in
Siddharth Pathak	211034	siddharthp21@iitk.ac.in
Sidharth A S	221056	sidharthas22@iitk.ac.in
Sparsh Gupta	221084	gsparsh22@iitk.ac.in
Ujjwal Kumar	211123	ujjwalk21@iitk.ac.in

Course: CS253

Mentor TA: Mr. Ashish Singh

Date: 6 April, 2025

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
0.1	Chatla Sowmya Sri Siddharth Pathak	Initial Commit (First Draft)	06/04/2025

1. Introduction

1.1. Testing Strategy: We adopted a hybrid testing strategy combining both Manual Testing and Test Automation:

- Manual Testing was used during initial development stages to quickly validate UI workflows, API endpoints, and database interactions, especially for features that required user interaction such as cart operations, ordering, and checkout.

- Test Automation was performed using the pytest framework along with Django's built-in testing tools. Automated unit tests were written to validate:

    - Backend API endpoints
    - Model behavior and database constraints
    - Business logic implemented in controller.py
    - Error handling and edge case scenarios
    - Notifications, cart integration, and invoice generation

1.2. Testing Timeline: Testing was carried out **in parallel with the implementation** of each module to enable continuous feedback and faster debugging. Specifically:

- Each app (e.g., users, products, cart, orders, etc.) was tested as soon as its key features were implemented.
- As the integration between apps began (e.g., orders and cart), integration testing was conducted.
- Final system-level testing, including end-to-end flows, was done after implementation was complete, ensuring cross-app compatibility and UI/API consistency.

1.3. Testers: Testing was performed by:

- Primary developers of the project, who wrote unit and integration test cases using pytest.
- Peer developers in the same team cross-tested each other’s modules to ensure fresh perspective testing and avoid tunnel vision.
- Where possible, we also requested feedback from other students in the lab to act as informal user testers for UX-level validations.

1.4. Coverage Criteria: We defined and achieved the following coverage criteria:

1. **Functional Coverage**

    a. All major functionalities from the Software Requirements Specification (SRS) and Design Document were tested.

   b. This included registration/login, store/product management, cart updates, placing orders, payment simulation, reviews, and notifications.

2. **Code Coverage**
   a. Targeted over 80% coverage for all critical files like models.py, views.py, controller.py, and urls.py using pytest-cov.

3. **Edge Case Coverage**
   a. Special test cases were included to cover scenarios such as:
      i. Placing an order with an empty cart
      ii. Database constraint violations
      iii. Invalid product/store IDs
      iv. Canceling already completed orders

4. **Negative Testing**
   a. Verified proper error responses, status codes (400, 403, 404), and exception handling using mock and invalid inputs.

1.5. Testing Tools: The following tools were used:

- **pytest** – for writing and executing unit tests and integration tests.
- **pytest-django** – for Django integration with pytest.
- **pytest-cov** – for measuring code coverage and generating coverage reports.
- **ThunderClient** – for manual API endpoint testing and request simulations.
- **Browser DevTools** – for manually testing UI endpoints and frontend-backend integration.

2. Unit Testing

Structural Coverage for Users App Functionalities

- **Code Coverage:** 89%
- **Branch Coverage:** 74%

1. Signup Functionality

What was tested: The user registration process was tested for various scenarios including successful signup, missing or invalid input fields, and duplicate account creation attempts.
Unit Details: Endpoint: /users/signup/, Method: POST
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Valid inputs (email, username, password, role): User registered successfully (Status: 201)
- Empty fields: Bad Request (Status: 400)
- Invalid role input: Bad Request (Status: 400)
- Duplicate user registration: User with this email already exists (Status: 400)

Structural Coverage: Included in overall users app structural report (see top).
Additional Comments: The signup function was tested with multiple input scenarios, ensuring validation checks work correctly.
GitHub Test Link:

2. Login Functionality

What was tested: Login endpoint functionality was tested using correct and incorrect credentials to verify authentication behavior and error handling.
Unit Details: Endpoint: /users/login/, Method: POST
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Correct Email and Password: Login Successful (Status: 200)
- Wrong Password: Unauthorized - Invalid Credentials (Status: 401)
- Non-existent Email: Unauthorized - Invalid Credentials (Status: 401)

Structural Coverage: Included in overall users app structural report (see top).
Additional Comments: Tests confirm that valid users can log in successfully, and incorrect credentials are properly handled.
GitHub Test Link:

3. Verify Email during Signup

What was tested: Verification email is sent during signup, and the correct OTP must be used to activate the account.
Unit Details: Endpoints: /users/signup-otp/, /users/verify-otp/, Method: POST
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Correct OTP after signup: OTP verified successfully (Status: 200)
- Incorrect OTP: Invalid OTP (Status: 400)

Structural Coverage: Included in overall users app structural report (see top).
Additional Comments: Mocking was used to simulate OTP generation and email delivery. The tests confirm that OTPs are verified correctly and that invalid OTP attempts are handled with appropriate error messages.
GitHub Test Link:

4. Reset Password after OTP Verification

What was tested: Reset password functionality based on verified OTP input was tested for successful reset and rejection of invalid attempts.
Unit Details: Endpoints: /users/send-otp/, /users/reset-password/ , Method: POST
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Valid OTP & Email: Password reset successful (Status: 200)
- Invalid OTP: Invalid OTP (Status: 400)
- Reset attempt without OTP verification: Bad Request - Invalid OTP (Status: 400)

Structural Coverage: Included in overall users app structural report (see top).
Additional Comments: The password reset functionality was tested to ensure that OTP verification is mandatory before resetting the password.
GitHub Test Link:

Structural Coverage for Stores App Functionalities

- **Code Coverage:** 94.5%
- **Branch Coverage:** 78.5%

1. Create Store Functionality

What was tested: Store creation was tested with role-based access and to ensure sellers can only create one store.
Unit Details: Endpoint: /stores/create/, Method: POST
Test Owner: Chatla Sowmya Sri

Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Authenticated seller creates store: Store created successfully (Status: 201)
- Seller attempts to create a second store: Only one store allowed per seller (Status: 400)
- Buyer attempts to create a store: Forbidden - Only sellers can create a store (Status: 403)

Structural Coverage: Included in overall stores app structural report (see top).
Additional Comments: The test cases ensure that only authenticated sellers can create stores and that each seller can own only one store.
GitHub Test Link:

2. Update Store Details Functionality
What was tested: Ability of store owners to update their store details. Prevents unauthorized access and modifications.
Unit Details: Endpoint: /stores/{store.id}/, Method: GET, PUT, DELETE
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Store owner updates store details: Store updated successfully (Status: 200)
- Unauthorized user attempts update: Forbidden - Unauthorized or not found (Status: 403)
- Buyer attempts to create a store: Forbidden - Only sellers can create a store (Status: 403)

Structural Coverage: Included in overall stores app structural report (see top).
Additional Comments: The test cases verify that only the store owner can update store details, ensuring security and data integrity.
GitHub Test Link:

Structural Coverage for Cart App Functionalities

- Code Coverage: 89%
- Branch Coverage: 60%

1. Add to Cart Functionality
What was tested: Adding products to the cart by buyers and ensuring sellers cannot perform this action. Verifies proper creation of cart items in the database.
Unit Details: Endpoint: /cart/add/, Method: POST
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Buyer adds a product: Item added (Status: 200)
- Seller attempts to add a product: Forbidden (Status: 403)
- Buyer attempts to create a store: Forbidden - Only sellers can create a store (Status: 403)

Structural Coverage: Included in overall cart app structural report (see top).
Additional Comments: Buyers can add multiple products to their cart. Sellers are restricted from accessing cart endpoints.
GitHub Test Link:

2. Clear Cart Functionality
What was tested: Complete removal of all items from the buyer's cart. Confirms proper deletion and role-based access control.
Unit Details: Endpoint: /cart/clear/{store_id}/, Method: DELETE
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Buyer clears entire cart: Cart cleared successfully (Status: 200)
- Seller attempts to clear cart: Forbidden (Status: 403)

Structural Coverage: Included in overall cart app structural report (see top).
Additional Comments: Clearing functionality works as expected, ensuring all cart items are removed for the buyer.
GitHub Test Link:

Structural Coverage for Products App Functionalities

- Code Coverage: 81.4%
- Branch Coverage: 75%

1. Public Category List View
What was tested: Retrieval of all public categories for a specific store.
Unit Details: Endpoint: `/products/public/categories/{store_id}/`, Method: GET
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Public categories retrieved successfully (Status: 200).

Structural Coverage: Included in overall products app structural report (see top).
Additional Comments: Ensures public access to category listing.

2. Public Product List View
What was tested: Retrieval of all public products in a specific category for a store.

Unit Details: Endpoint: `/products/public/products/{store_id}/{category_id}/`, Method: GET
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Public products retrieved successfully (Status: 200).

Structural Coverage: Included in overall products app structural report (see top).
Additional Comments: Ensures public access to product listing.

3. Authenticated Product List and Create View
What was tested: Retrieval of all products and creation of a new product by an authenticated seller.
Unit Details: Endpoint (List): `/products/products/`, Method: GET
Endpoint (Create): `/products/products/`, Method: POST
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Products retrieved successfully (Status: 200).
- Product created successfully (Status: 201).

Structural Coverage: Included in overall products app structural report (see top).
Additional Comments: Confirms role-based access control and product creation functionality.

4. Authenticated Product Detail View
What was tested: Retrieval, update, and deletion of a specific product by an authenticated seller.
Unit Details: Endpoint (Retrieve): `/products/products/{product_id}/`, Method: GET
Endpoint (Update): `/products/products/{product_id}/`, Method: PUT
Endpoint (Delete): `/products/products/{product_id}/`, Method: DELETE
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Product retrieved successfully (Status: 200).
- Product updated successfully (Status: 200).
- Product deleted successfully (Status: 204).

Structural Coverage: Included in overall products app structural report (see top).
Additional Comments: Confirms CRUD operations for products.

5. Authenticated Category List and Create View

What was tested: Retrieval of all categories and creation of a new category by an authenticated seller.

Unit Details: Endpoint (List): `/products/categories/`, Method: GET
Endpoint (Create): `/products/categories/`, Method: POST

Test Owner: Siddharth Pathak

Test Date: [03/04/2025] - [04/04/2025]

Test Results:

- Categories retrieved successfully (Status: 200).
- Category created successfully (Status: 201).

Structural Coverage: Included in overall products app structural report (see top).

Additional Comments: Confirms role-based access control and category creation functionality.

6. Authenticated Category Detail View

What was tested: Retrieval, update, and deletion of a specific category by an authenticated seller.

Unit Details: Endpoint (Retrieve): `/products/categories/{category_id}/`, Method: GET
Endpoint (Update): `/products/categories/{category_id}/`, Method: PUT
Endpoint (Delete): `/products/categories/{category_id}/`, Method: DELETE

Test Owner: Siddharth Pathak

Test Date: [03/04/2025] - [04/04/2025]

Test Results:

- Category retrieved successfully (Status: 200).
- Category updated successfully (Status: 200).
- Category deleted successfully (Status: 204).

Structural Coverage: Included in overall products app structural report (see top).

Additional Comments: Confirms CRUD operations for categories.

Structural Coverage for Orders App Functionalities

- Code Coverage: 92.4%
- Branch Coverage: 77%

1. Checkout Cart Success

What was tested: Successful checkout of a cart with items, Ensures proper order creation and validation.

Unit Details: Endpoint: `/orders/checkout`, Method: POST

Test Owner: Siddharth Pathak

Test Date: [03/04/2025] - [04/04/2025]

Test Results:

- Cart items checked out successfully (Status: 200).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Confirms that the checkout process creates an order with valid items, payment method, and delivery address.

2. Checkout Cart Empty
What was tested: Attempted checkout with an empty cart. Ensures proper error handling.
Unit Details: Endpoint: `/orders/checkout`, Method: POST
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Checkout failed with an appropriate error message (Status: 400).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Confirms that the API prevents checkout when the cart is empty.

3. Checkout Without Phone Number
What was tested: Attempted checkout without a phone number in the buyer's profile. Ensures proper validation.
Unit Details: Endpoint: `/orders/checkout`, Method: POST
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Checkout failed due to missing phone number (Status: 400).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Confirms that a valid phone number is required for placing an order.

4. Cancel Order Success
What was tested: Successful cancellation of an order by the buyer.
Unit Details: Endpoint (Cancel Order) `/cancel/{order_id}/`, Method: POST
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Order canceled successfully (Status: 200).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Confirms that buyers can cancel their orders before delivery.

5. Cancel Delivered Order
What was tested: Attempted cancellation of an already delivered order. Ensures proper restrictions.
Unit Details: Endpoint (Cancel Order): `/orders/cancel/{order_id}/`, Method: POST
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Cancellation failed with an appropriate error message (Status: 403).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Confirms that delivered orders cannot be canceled.

6. Confirm Delivery (Success & Reconfirmation Attempt)
What was tested: Successful confirmation of delivery by the buyer. Attempted confirmation of an already delivered order to ensure proper validation.
Unit Details: Endpoint (Confirm Delivery): `/orders/confirm/{order_id}/`, Method: POST
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Delivery confirmed successfully (Status: 200).
- Re-confirmation of an already delivered order failed with an appropriate error message (Status: 400).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Ensures that buyers can confirm the delivery of their orders, and proper validation prevents reconfirmation of already delivered orders.

7. User Order List
What was tested: Retrieval of all orders placed by a user. Ensures proper filtering by user ID.
Unit Details: Endpoint: `/orders/myorders/`, Method: GET
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Orders retrieved successfully (Status: 200).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Confirms that users can retrieve their order history.

8. Seller Order view

What was tested: Retrieval of all orders for a seller's store. Ensures proper filtering by store and seller ID.
Unit Details: Endpoint: `/orders/sellerorders/`, Method: GET
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Seller's orders retrieved successfully (Status: 200).

Structural Coverage: Included in overall orders app structural report (see top).
Additional Comments: Confirms that sellers can view all orders for their store.

3. Integration Testing

1. OTP Verification During Sign Up

Module Details: OTP Workflow (Send and Verify)
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- OTP Sending Test: OTP was successfully sent to the user's email during signup.
- OTP Verification Test: Correct OTP was verified successfully, allowing signup to proceed.
- Invalid OTP Test: Incorrect or expired OTP resulted in an appropriate error message ("Invalid OTP").

Additional Comments: Integration between OTP verification and signup ensures secure registration.

2. Signup and Login Flow

Module Details: Signup and Login Redirection
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Signup Redirection Test: After new user signs up with valid credentials, the app is redirected to the login page as expected.
- Login Success Test: Login with new credentials succeeded; user was authenticated and allowed into the system.

Additional Comments: The signup and login flow is functioning as intended. New user accounts are created successfully, and the application redirects to the login page immediately after signup. Login with the newly registered credentials works without any issues, confirming that account creation and authentication processes are correctly integrated.

3. Login Role-Based Redirection

Module Details: Login
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Upon entering valid login credentials, users are redirected based on their role.
- Buyer Redirection Test: Buyers are redirected to the homepage.
- Seller Redirection Test: Sellers are redirected to the ordersview page.

Additional Comments: Test confirms correct post-login redirection flow based on user roles, ensuring a smooth and contextual user experience.

4. Forgot Password Flow

Module Details: Forgot Password, OTP Verification, Password Reset
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- OTP Generation Test: Entering a registered email in the “Forgot Password” page triggered OTP generation and successfully sent it to the user’s email.
- Password Reset Test: Correct OTP with a new password, reset the account successfully and redirected the user to the login page.
- Post-Reset Login Test: Login with the newly set password worked without issues.

Additional Comments: The entire password recovery flow is functioning as expected — from email validation and OTP delivery to password reset and login verification.

5. Navigation Bar Integration

Module Details: Navigation Bar
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Navigation Links Test: All navigation links were functional and correctly directed users to the intended pages.
- Dynamic Content Test: Navigation bar updated dynamically based on user login status. For example, buyers see the "Orders" link when logged in.

Additional Comments: Integration with the navigation bar enhances user experience by providing easy access to key sections of the platform.

6. Search Functionality Integration

Module Details: Store-Specific Product Search
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Search Results Test (Products): Verified that relevant products were displayed based on search queries within a selected store.

Additional Comments: Search functionality is limited to individual store pages, allowing users to search for products after selecting a store.

7. Profile Page Integration

Module Details: Profile Page, Login
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]

Test Results:

- User Info Fetch Test: Successfully fetched and displayed user information such as phone number and address.
- Buyer Profile Edit Test: Verified that buyers can view and edit their personal details.
- Seller Profile Update Test: Confirmed that for sellers, the profile page is linked with store information, and both user and store details can be updated.

Additional Comments: The profile component provides a role-specific interface—buyers manage their personal info, while sellers have integrated access to both profile and store-related data.

8. Logout

Module Details: Logout
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Logout functionality works across all pages.
- Post-Logout Redirection Test: Upon logging out from any page, the user is successfully redirected to the homepage.

Additional Comments: Session is properly cleared upon logout, ensuring secure exit and preventing unauthorized access through browser navigation or refresh.

9. Product Viewing & Management

Module Details: Categories and Products
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Seller Prdoucts Management Test:Sellers can add, update, delete categories and products via categoriesview.
- Buyer Menu Access Test: Buyers access product listings through store-specific menu view.

Additional Comments: Navigation and product/category display are functioning as intended for both sellers and buyers, with smooth transitions and proper data rendering across views.

10. Cart to Orders Integration

Module Details: Cart and Orders
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Cart Checkout Test: Items added to the cart were successfully converted into an order upon checkout.
- Empty Cart Test: Attempting to checkout an empty cart resulted in an appropriate error message ("No cart found for this user at this store").

Additional Comments: Integration between the cart and orders modules ensures seamless order creation upon checkout, with proper validation for empty carts.

11. Orders View Based on Role
Module Details: Orders Integration for Buyers and Sellers
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Buyer Order List Test: Verified that buyers can view all their placed orders via the dedicated orders page.
- Seller Order List Test: Confirmed that sellers can access all store-related orders through the ordersview page.

Additional Comments: The integration ensures accurate routing and filtering of orders based on user roles—buyers and sellers are directed to their respective order views (/orders and /ordersview).

12. Products to Orders Integration
Module Details: Product Selection, Category-Based Filtering
Test Owner: Chatla Sowmya Sri
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Category Filtering Test: When a buyer selects a category on a store's menu page, only the products corresponding to that category are displayed.
- Cart Addition Test: Buyers can add products to the cart from the displayed product list.

Additional Comments: The flow from product selection to cart addition works smoothly, with category-based filtering and cart integration functioning as expected.

13. Cart to Orders Integration
Module Details: Cart and Orders
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Cart Checkout Test: Items added to the cart were successfully converted into an order upon checkout.
- Empty Cart Test: Attempting to checkout an empty cart resulted in an appropriate error message ("No cart found for this user at this store").

Additional Comments: Integration between the cart and orders modules ensures seamless order creation upon checkout, with proper validation for empty carts.

14. Order-Payment Integration Flow
Module Details: Payment and Orders
Test Owner: Siddharth Pathak
Test Date: [03/04/2025] - [04/04/2025]
Test Results:

- Payment Creation Test: Payments were successfully initiated for valid orders.
- Payment Verification Test: Payments were verified successfully using Razorpay API integration.
- Refund Processing Test: Refunds were processed correctly for canceled orders.

Additional Comments: Integration between payment and orders ensures smooth payment initiation, verification, and refund processing.

4. System Testing

1. Performance Testing

Requirement: The system should handle a certain number of concurrent users without performance degradation.
Test Owner: Siddharth Pathak
Test Date: [04/04/2025] - [05/04/2025]
Test Results:

- Conducted performance testing using Apache Benchmark to assess system response times and resource utilization under normal and peak load conditions.
- Monitored server response times, database query performance, and system stability under load.
- Analyzed performance metrics to ensure the system meets performance requirements.

Additional Comments: The system performed well under normal and peak load conditions, meeting performance requirements and providing a responsive user experience.

2. Data Integrity Testing

Requirement: The system should ensure the integrity of user data by accurately storing and retrieving information from the database.
Test Owner: Siddharth Pathak
Test Date: [04/04/2025] - [05/04/2025]
Test Results:

- Conducted data integrity testing manually to verify that data entered by users is correctly stored in the database and retrieved accurately when requested.
- Tested various data types, including text, numbers, dates, and images.
- Checked for data consistency and accuracy across different modules and transactions.

Additional Comments: Data integrity testing confirms that the system maintains the integrity and consistency of user data, providing reliable information for users and administrators.

3. Error Handling Testing

Requirement: The system should effectively handle errors and exceptions to provide users with informative error messages and recover gracefully from unexpected situations.
Test Owner: Siddharth Pathak

Test Date: [04/04/2025] - [05/04/2025]
Test Results:

- Conducted error handling testing to identify and validate the system's response to various error conditions, such as invalid inputs, network errors, and system failures.
- Verified that appropriate error messages are displayed, and the system gracefully recovers from errors without data loss or instability.

Additional Comments: Error handling testing ensures that the system provides a robust user experience by guiding users through error situations and maintaining system integrity.

4. Concurrency Testing

Requirement: The system should support multiple users accessing and interacting with the system simultaneously without data corruption or performance degradation.
Test Owner: Siddharth Pathak
Test Date: [04/04/2025] - [05/04/2025]
Test Results:

- Conducted concurrency testing using Apache Benchmark to evaluate the system's behavior under concurrent user loads.
- Simulated multiple users accessing the system concurrently and performing typical interactions like logging in, browsing, and submitting data.
- Monitored performance, resource utilization, and data consistency under concurrent load conditions.

Additional Comments: Concurrency testing confirms that the system can handle multiple users simultaneously without experiencing data corruption or significant performance degradation.

5. Session Management Testing

Requirement: The system should manage user sessions securely and efficiently to maintain authentication and session state.
Test Owner: Siddharth Pathak
Test Date: [04/04/2025] - [05/04/2025]
Test Results:

- Verified that user sessions are created, maintained, and terminated correctly.
- Tested session expiration, persistence across pages, and session hijacking prevention measures.
- Confirmed that users are logged out automatically after inactivity and that session tokens are securely transmitted.

Additional Comments: Session management testing ensures secure authentication throughout user sessions.

6. **Email Notification Testing**

Requirement: The system should send email notifications to users for events like account registration, password reset, order confirmation, etc.

Test Owner: Chatla Sowmya Sri

Test Date: [04/04/2025] - [05/04/2025]

Test Results:

- Validated email notifications for various events (e.g., registration confirmation).
- Verified content formatting, delivery success across providers, and spam handling.
- Checked error handling for failed email deliveries.

Additional Comments: Email notification testing ensures timely communication with users for important events.

5. Conclusion

How Effective and exhaustive was the testing?

The testing process was highly effective and exhaustive, covering a wide range of functional and non-functional requirements. Through meticulous test planning, execution, and analysis, we ensured thorough coverage of all use cases and scenarios. As a result, the system's robustness and reliability were significantly enhanced, and the risk of critical defects in production was minimized.

Which components have not been tested adequately?

While our testing efforts were comprehensive, certain components, such as those involving complex integrations with external systems or third-party APIs, may have received less extensive coverage. Additionally, non-functional aspects like performance, scalability, and security could benefit from further testing to ensure optimal system behavior under varying conditions

What difficulties have you faced during testing?

Large interconnection among different modules and views made it difficult to perform the integration testing. Additionally, time constraints left limited room for thorough testing and debugging. On top of this, adapting to changing requirements during the testing phase and managing those changes effectively was a constant struggle, as it required revisiting test cases and workflows to ensure everything aligned with the updated scope.

How could the testing process be improved?

To make our testing process better, we should start by introducing automated tools and frameworks to handle repetitive tasks and increase test coverage. Automating integration testing with something like Cypress could really help us consistently check workflows across different components without manual effort. It’s also important to dedicate specific resources to testing and improve communication between the development and QA teams so we can work more smoothly and resolve issues faster. By doing risk assessments early in the project, we can figure out which areas need the most attention and focus our efforts where they’ll have the biggest impact. On top of that, regularly taking time to reflect on what’s working and what’s not—through retrospectives—will help us keep improving our testing strategies over time.

Appendix A - Group Log

During testing, the group maintained constant communication via our WhatsApp group to coordinate efforts and address issues promptly. We also have collaborated offline in the KD Lab of the Dept. of Computer Science and Engineering to ensure smoother and more efficient testing.

Meeting Time	**Agenda/Discussions**
29 March 2025 (21:00 - 22:00 IST)	Discussed the software testing plan, and work distribution.
31 March 2025 (21:00 - 22:00 IST)	Discussed the automated and manual testing part with the TA
2 April 2025 (16:30 - 17:00 IST)	Manual Testing start
3 April 2025 (14:30 - 15:30 IST)	Manual testing ends, start writing code for automated tests.
4 April 2025 (10:00 - 11:00 IST)	Automated testing (Unit testing) done
5 April 2025 (20:00 - 3:00 IST)	Document writing