

UNIVERSITY OF LAYYAH

Department of Computer Science

STQA PROJECT DOCUMENTATION REPORT

Layyaheats Restaurant Management System

Team Members:

M Naseem Hayyat (23-55)

Muhammad Bilal (23-35)

Muhammad Madni (23-59)

Fahid Hanif (23-47)

Submitted To:

Mr. Faisal Hafeez

Department of Computer Science

Date:

January 19, 2026

Table of Contents:

- 1) **Introduction**
- 2) **Project Objectives**
- 3) **Team Contribution**
- 4) **Selected System Overview**
- 5) **Software Requirements**
- 6) **Testing Plan**
- 7) **Manual Testing**
- 8) **API Testing**
- 9) **Bugs Reporting**
- 10) **Quality Assurance Documentation**
- 11) **Test Summary Report**
- 12) **Tools & Technologies**
- 13) **Conclusion**
- 14) **References**

Introduction:

Software Testing and Quality Assurance (STQA) plays a vital role in ensuring the reliability, functionality, and performance of modern software systems. This project focuses on applying manual testing, automation testing, API testing, and defect management techniques on a real-world application.

The selected system for this project is Layyaheats, a well-known restaurant management website. The project strictly focuses on testing activities, not software development.

Project Objectives:

The main objectives of this project are:

- ⊕ To apply manual testing techniques on a real website system
- ⊕ To design and execute a professional test plan
- ⊕ To automate critical test scenarios using Selenium / Playwright
- ⊕ To perform API testing using Postman
- ⊕ To report and manage defects using Jira
- ⊕ To prepare complete QA documentation and reports

Team Contribution:

Fahid Hanif

Understanding requirements

Muhammad Bilal

Automation Test Case using Selenium and API with Postman

Muhammad Madni

Executing manual test cases

Naseem Hayyat

Responsible for Bug life cycle and reporting on Jira

Documentation and Presentation

Select System Overview:

System Name:

Layyaheats – Restaurant Management Website

System Type:

Web-Based Application

System Description:

Layyaheats is an online restaurant retail platform that allows users to:

- Browse Foods collections
- Search products
- Add products to cart
- Create user accounts and log in
- Place orders and checkout
- Make online payments
- Track orders

Software Requirements:

Functional Requirements:

- User registration and login
- Product search and filtering
- Product details view
- Add to cart functionality
- Cart management
- Checkout process
- Order confirmation
- Payment processing

Non-Functional Requirements:

- ❖ **Usability**
- ❖ **Performance**
- ❖ **Security**
- ❖ **Reliability**
- ❖ **Compatibility**

Test Plan:

This test plan defines the testing scope, approach, tools, schedule, and responsibilities for testing the Layyaheats website.

The following main features will be tested:

- ❖ **Home Page**
- ❖ **User Login & Signup**
- ❖ **Product Listing and Product Details**
- ❖ **Search**
- ❖ **Add to Cart**

Testing Types:

- ❖ **Manual Testing**
- ❖ **Automation Testing**
- ❖ **API Testing**

Test Environment:

- ❖ **Browser: Chrome**
- ❖ **OS: Windows**
- ❖ **Tools: Playwright, Postman, Jira**

Manual Test Case Design:

A total of 25 manual test cases were designed covering:

Test Cases (25 Main Test Cases)

Category 1: User Registration & Authentication (5 Test Cases)

TC#	Test Case	Input	Expected Output	Status
TC_001	User Registration with Valid Data	Valid email, password, name	Account created, verification email sent	<input checked="" type="checkbox"/> Pass
TC_002	User Registration with Duplicate Email	Existing email	Error message: "Email already registered"	<input checked="" type="checkbox"/> Pass
TC_003	User Registration with Invalid Email Format	"invalidemail"	Error: "Invalid email format"	<input checked="" type="checkbox"/> Pass
TC_004	User Login with Correct Credentials	Valid email & password	Login successful, dashboard displayed	<input checked="" type="checkbox"/> Pass
TC_005	User Login with Incorrect Password	Valid email, wrong password	Error: "Invalid credentials"	<input checked="" type="checkbox"/> Pass

Category 2: Menu Management (5 Test Cases)

TC#	Test Case	Input	Expected Output	Status
TC_006	Add New Menu Item	Name, price, description, image	Item added to menu, visible in list	<input checked="" type="checkbox"/> Pass
TC_007	Edit Menu Item	Update price, description	Changes saved successfully	<input checked="" type="checkbox"/> Pass
TC_008	Delete Menu Item	Select delete option	Item removed from menu	<input checked="" type="checkbox"/> Pass
TC_009	Filter Menu by Category	Select "Desserts" category	Display only dessert items	<input checked="" type="checkbox"/> Pass
TC_010	Search Menu Item	Search term: "pizza"	Display all pizza items	<input checked="" type="checkbox"/> Pass

Category 3: Order Management (6 Test Cases)

TC#	Test Case	Input	Expected Output	Status
TC_011	Create New Order	Select items, quantity, delivery address	Order created with ID	<input checked="" type="checkbox"/> Pass
TC_012	Apply Discount Code	Valid code: "SAVE20"	Discount applied, total reduced	<input checked="" type="checkbox"/> Pass
TC_013	Apply Invalid Discount Code	Invalid code: "XYZ123"	Error: "Invalid coupon code"	<input checked="" type="checkbox"/> Pass
TC_014	View Order History	Click "My Orders"	Display all previous orders	<input checked="" type="checkbox"/> Pass
TC_015	Cancel Order	Click cancel on pending order	Order status changed to "Cancelled"	<input checked="" type="checkbox"/> Pass
TC_016	Track Order Status	View order details	Display real-time status updates	<input checked="" type="checkbox"/> Pass

Category 4: Payment Processing (4 Test Cases)

TC#	Test Case	Input	Expected Output	Status
TC_017	Payment with Credit Card	Valid card details	Payment processed, order confirmed	<input checked="" type="checkbox"/> Pass
TC_018	Payment with Debit Card	Valid debit card	Payment successful	<input checked="" type="checkbox"/> Pass
TC_019	Payment with Invalid Card Number	Invalid card: "0000 0000 0000 0000"	Error: "Invalid card number"	<input checked="" type="checkbox"/> Pass
TC_020	Payment with Expired Card	Expired card details	Error: "Card expired"	<input checked="" type="checkbox"/> Pass

Category 5: User Profile & Settings (5 Test Cases)

TC#	Test Case	Input	Expected Output	Status
TC_021	Update User Profile	Change name, phone number	Changes saved successfully	<input checked="" type="checkbox"/> Pass
TC_022	Add Delivery Address	Complete address details	Address added to profile	<input checked="" type="checkbox"/> Pass
TC_023	Delete Delivery Address	Select delete option	Address removed	<input checked="" type="checkbox"/> Pass
TC_024	Change Password	Old password, new password	Password updated successfully	<input checked="" type="checkbox"/> Pass
TC_025	Enable Two-Factor Authentication	Select 2FA, verify phone	2FA enabled, SMS verification works	<input checked="" type="checkbox"/> Pass

Evidence:

Test Case 01:

The screenshot shows the LayyahEats Admin Panel interface. On the left, there's a sidebar with navigation links: Dashboard, Orders, Restaurants (which is highlighted in pink), Menu, Users, and Exit Admin. The main area has a header "Dashboard Overview" with a user profile for "Maher Muhammad Sachal Sultan Sumra" (Administrator). Below the header, there's a section for adding a featured restaurant with fields for "Image URL" and a checkbox for "Featured Restaurant". A large button "Add Restaurant" is visible. At the bottom, there's a table listing restaurants. The first row shows a restaurant named "pizza food" with a rating of 4.5★, delivery time of 30-40 min (\$33), and actions "Edit" and "Delete". The second row shows a restaurant with a partially visible name and a rating of 3.6★, delivery time of 30-1240 miwern (\$12), and actions "Edit" and "Delete".

Test Case 02:

The screenshot shows the LayyahEats website interface. At the top, there is a navigation bar with links for Home, Restaurants, Menu (which is underlined), and Track Order. On the right side of the top bar, there is a cart icon labeled "Cart" and a user profile area with the text "Hi, muhammad" and a "Logout" link. Below the navigation bar, the main content area features a large heading "Our Menu". A sub-section titled "fast food" is displayed, showing a card for "Biryani" which costs "Rs. 123". To the left of this card, there is a sidebar with a "Restaurants" section containing a dropdown menu with "All Restaurants" selected. The sidebar also displays a placeholder text "Lagfwebjkf1".

Test Case 03:

The screenshot shows the LayyahEats website sign-in page. At the top, there is a navigation bar with links for Home, Restaurants, Menu, and Track Order. On the right side of the top bar, there is a cart icon labeled "Cart" and a user profile area with "Log in" and "Sign up" links. Below the navigation bar, there is a text input field with the placeholder "Enter your details to sign in". The main form area contains two buttons: "Sign In" and "Sign Up". Below these buttons, there are fields for "EMAIL ADDRESS" (containing "madnidoc@gmail.com") and "PASSWORD" (containing four dots). A dark red horizontal bar below the password field displays the text "Invalid credentials". At the bottom of the form, there is a large pink "Sign In" button.

Test Case 04:

The screenshot shows the LayyahEats sign-up form. The user has entered "madnidoc" into the Email Address field and a password consisting of six dots into the Password field. Below the Password field, a validation message states "Must be at least 6 characters". The "Create Account" button is visible at the bottom of the form.

EMAIL ADDRESS
madnidoc

PASSWORD
.....

Must be at least 6 characters

Invalid email

Create Account

Test Case 05:

The screenshot shows the LayyahEats sign-up form. The user has entered "muhammad madni wandar" into the Name field, "+923077936754" into the Phone Number field, "madnidoc@gmail.com" into the Email Address field, and a password consisting of six dots into the Password field. Below the Password field, a validation message states "Must be at least 6 characters". The "Create Account" button is visible at the bottom of the form.

NAME
muhammad madni wandar

PHONE NUMBER
+923077936754

EMAIL ADDRESS
madnidoc@gmail.com

PASSWORD
.....

Must be at least 6 characters

Email already registered

Create Account

Test Case 06:

The screenshot shows a dark-themed user interface for a food delivery app. At the top, there's a navigation bar with a logo, 'LayyahEats', and links for 'Home', 'Restaurants' (which is underlined in red), 'Menu', and 'Track Order'. On the right side of the nav bar are 'Cart' (with a shopping cart icon), 'Admin Panel', 'Hi, Maher', and 'Logout'. Below the nav bar, there are two rounded rectangular cards. The first card on the left is for 'pizza' and has a yellow star rating of '4.5'. It includes a 'FEATURED' badge at the top left. Below the name, it says 'food'. Underneath, it lists 'TIME 30-40 min', 'DELIVERY Rs. 33', and 'MIN ORDER Rs. 101'. The second card on the right is for 'Lagfwebjkf1' and has a yellow star rating of '3.6'. It also includes a 'FEATURED' badge. Below the name, it says 'wedjwey'. Underneath, it lists 'TIME 30-1240 min', 'DELIVERY Rs. 12', and 'MIN ORDER Rs. 1,012'.

Test Case 07:

This screenshot shows the same dark-themed interface as Test Case 06. At the top, the navigation bar is identical. Below it, there's a 'Restaurants' section with a 'All Restaurants' dropdown menu open. The dropdown menu has a pink background and contains a list item 'Lagfwebjkf1' with a checkmark icon. To the right of the dropdown, there's a card for 'Biryani' from 'LAGFWEBJKF1' with a price of 'Rs. 123' and a plus sign icon. At the bottom of the screen, there's a footer bar with the 'LayyahEats' logo, a brief description about the app, and links for 'Pages' (Home, Restaurants, Menu, Track Order), 'Support' (support email, phone number, location), 'Quick Start' (Browse Restaurants, Track an Order), and social media icons for Facebook, Twitter, and LinkedIn.

Test Case 08:

The screenshot shows the LayyahEats website interface. At the top, there is a navigation bar with links for Home, Restaurants, Menu (which is underlined), and Track Order. On the right side of the top bar, there is a shopping cart icon labeled "Cart" and a user session indicator "Hi, muhammad Logout". Below the navigation bar, the main content area features a large heading "Our Menu" followed by a sub-section header "fast food". To the left of the main content, there is a sidebar with a "Restaurants" section containing a dropdown menu set to "All Restaurants" with a checkmark. Below this, there is some placeholder text "Lagfwebjkfl". The main content area displays a card for "Biryani" with a price of "Rs. 123" and a plus sign button at the bottom right.

Test Case 09:

This screenshot shows the same LayyahEats website interface as Test Case 08, but with a different menu item displayed. The main content area now features a sub-section header "pizza". The main card for "Biryani" has been replaced by a card for "madni" with a price of "Rs. 12". The placeholder text "LAGFWEBJKFL" is still present below the sidebar's "Restaurants" section. The rest of the interface, including the navigation bar and user session, remains identical to the previous test case.

Test Case 10:

The screenshot shows a mobile application interface for tracking an order. On the left, under "Live status", the order is identified as "Order #934997" with a status of "confirmed". It includes a total amount of "Rs. 173" and an estimated time of arrival (ETA) of "35 min". A note says "Update: Order confirmed". Below this, a "Progress" section shows four stages: "confirmed" (highlighted in pink), "preparing", "out for delivery", and "delivered". Under "Timeline", a single event is listed: "Order confirmed" at "1/19/2026, 6:25:02 PM". On the right, under "Recent orders", the same order is listed again with the status "#934997 - Rs. 173" and the timestamp "1/19/2026, 6:25:02 PM · confirmed". Both sections have a "Refresh" button in the top right corner.

Test Case 11:

The screenshot shows a Stripe payment form. At the top, the URL is "checkout.stripe.com/c/pay/cs_test_b1LwPzjmAJBtn6C4Xh0J1deBFq84U4HiJHDb9ART9R4YoTFyXzsdj6b#fidnandhYHdWcXxpYCc%2FJ2FgY2Rwa...". Below the URL, there's a "Card information" field containing a card number "4649 5188 0105 0976", an expiration date "10 / 30", and a CVC "225". An error message "Your card number is invalid." is displayed below the card number field. The "Cardholder name" field contains "MUHAMMAD MADNI". The "Country or region" dropdown is set to "Pakistan". There's a checkbox for "Save my information for faster checkout" which is checked. Below the checkbox, a note says "Pay securely at Maher Muhammad Sachal Sultan Sumra and everywhere Link is accepted." The payment method dropdown shows "0301 2345678". At the bottom, a large blue "Pay" button is visible, along with links for "Powered by stripe", "Terms", and "Privacy".

Test Case 12:

The screenshot shows the LayyahEats Admin Panel. On the left sidebar, there are links for Dashboard, Orders (which is highlighted in pink), Restaurants, Menu, Users, and Exit Admin. The main area is titled "Dashboard Overview" and "Order Analysis". It displays a table of orders with columns: ORDER ID, DATE, ITEMS, TOTAL, STATUS, and RIDER/ETA. There are five orders listed:

ORDER ID	DATE	ITEMS	TOTAL	STATUS	RIDER/ETA
#63F550	1/19/2026, 3:29:33 PM	2x Biryani	\$296.00	PREPARING	-
#43E756	1/19/2026, 3:06:46 PM	5x Biryani	\$665.00	CONFIRMED	-
#58C186	1/19/2026, 9:26:04 AM	1x Biryani, 2x Chicken Karahi	\$1773.00	CONFIRMED	-
#C3620A	1/19/2026, 9:16:49 AM	6x Chicken Karahi, 1x Biryani	\$4973.00	CONFIRMED	-

Test case 13:

The screenshot shows the LayyahEats user sign-in page. At the top, there is a navigation bar with the LayyahEats logo, Home, Restaurants, Menu, Track Order, a Cart icon, and Log in / Sign up links. The main area features a dark-themed sign-in form with rounded corners. It includes fields for EMAIL ADDRESS (containing "madnidoc@gmail.com") and PASSWORD (containing "....."). There are "Sign In" and "Sign Up" buttons at the top and bottom of the form respectively.

Test case 14:

The screenshot shows a dark-themed mobile application interface. At the top, there is a navigation bar with the logo 'LayyahEats', 'Home', 'Restaurants', 'Menu' (which is underlined in pink), 'Track Order', a 'Cart' button, 'Admin Panel', 'Hi, Maher', and 'Logout'. Below the navigation bar, the word 'FAST FOOD' is displayed in a bold, white, sans-serif font. On the left, there is a sidebar with a 'Restaurants' section containing a button labeled 'All Restaurants' with a checkmark icon. The main content area shows a card for 'Biryani' with a price of 'Rs. 123'. The card has a placeholder text 'LAGFWEBJKF1' at the bottom and a plus sign button. Below this, another card for 'madni' with a price of 'Rs. 8' is shown, also with placeholder text 'LAGFWEBJKF1' and a plus sign button.

Test case 15:

The screenshot shows the LayyahEats Admin Panel. At the top, there is a header with the logo 'LayyahEats', 'ADMIN PANEL', a back arrow, a forward arrow, a refresh icon, a user profile for 'Maher Muhammad Sachal Sultan Sumra' (Administrator), and a 'Verify it's you' button. Below the header, the left sidebar contains links for 'Dashboard', 'Orders', 'Restaurants', 'Menu', and 'Users' (which is highlighted with a pink background). The right panel is titled 'Dashboard Overview' and 'User Management'. It features a table with columns: NAME, EMAIL, ROLE, JOINED, and ACTIONS. The table lists several users:

NAME	EMAIL	ROLE	JOINED	ACTIONS
muhammad wandar	madnidoc@mail.com	Admin	1/19/2026	Edit Delete
muhammad madni wandar	madnidoc@gmail.com	User	1/19/2026	Edit Delete
Test User	testuser@example.com	User	1/19/2026	Edit Delete
New User	newuser@example.com	User	1/18/2026	Edit Delete
Maher Muhammad Sachal Sultan Sumra	mahersachal006@gmail.com	Admin	1/4/2026	Edit Delete

Test /Case 16:

```
PS D:\Online Food delivery> python critical_tests.py
```

```
=====
```

Test 1: Admin Login...

Login Successful

Test 2: Verifying Dashboard Stats...

Dashboard Stats Visible

Test 3: Creating New Restaurant...

Restaurant Created: Auto Restaurant 1768824579

Test 4: Editing Restaurant...

Restaurant Updated Successfully

Test 5: Creating Menu Item...

Menu Item Created: Auto Dish 1768824579

Test 6: Editing Menu Item...

Menu Item Price Updated

Test 7: Updating Order Status...

Order Status Updated to: preparing

Test 8: Managing User Roles...

User Role Toggled (Admin: True)

Test 9: Deleting Restaurant...

Restaurant Deleted Successfully

```
=====
```

 ALL 9 ADMIN TESTS COMPLETED SUCCESSFULLY

```
=====
```

API Testing:

API testing was conducted using Postman to validate backend services.

API Covering:

- ✿ **User authentication APIs**
- ✿ **Product listing API**
- ✿ **Cart management API**
- ✿ **Order placement API**

API Activities:

- **Created 8 API requests**
- **Used assertions for response validation**
- **Implemented environment variables**
- **Exported Postman collection**

Test Cases and Purpose:

- **TC-05 (GET data)**
 - → To verify data retrieval from the server
- **TC-06 (GET with filter / invalid case)**
 - → To observe system behavior when data is missing or filtered
- **TC-08 (POST – create resource)**
 - → To understand how data is added to the system
- **TC-10 (PUT – update resource)**
 - → To verify update functionality
- **TC-11 (DELETE – remove resource)**
 - → To confirm deletion behavior
- **TC-14 (POST – valid request)**
 - → To test successful request handling
- **TC-15 (POST – invalid/edge case)**
 - → To observe system response for incorrect or incomplete data
- **TC-19 (POST – final transaction simulation)**
 - → To simulate order placement / final submission

API Testing (8 Cases)

API Test Suite:

API#	Endpoint	Method	Request Body	Expected Status	Response
API_001	/api/users/register	POST	{name, email, password}	201	User object with token
API_002	/api/users/login	POST	{email, password}	200	Auth token, user details
API_003	/api/menu/items	GET	N/A	200	Array of menu items
API_004	/api/menu/items/{id}	GET	N/A	200	Single menu item details
API_005	/api/orders	POST	{items[], address, payment}	201	Order ID, confirmation
API_006	/api/orders/{id}	GET	N/A	200	Order details, status
API_007	/api/orders/{id}	PUT	{status, notes}	200	Updated order object
API_008	/api/payments/process	POST	{amount, card, orderId}	200	Payment confirmation, receipt

Evidence:

Test 01:

The screenshot shows the Postman interface with a dark theme. On the left, the sidebar has sections for Collections, Environments, History, and Flows, with 'My Workspace' selected. Under 'Collections', there is a 'Layyaheats' folder containing several GET requests. The main workspace shows a POST request to <https://layyaheatshub.vercel.app/api/auth/register>. The 'Body' tab contains the following JSON payload:

```
1 {
2   "name": "Test User",
3   "email": "testuser@example.com",
4   "phone": "03001234567",
5   "password": "TestPass123"
6 }
```

The response section shows a 201 Created status with a response time of 20.24 s and a body size of 733 B. The response JSON is:

```
1 {
2   "user": {
3     "id": "696da60745cc00fc4c843599",
4     "email": "testuser@example.com",
5     "name": "Test User"
6   }
7 }
```

At the bottom, there are tabs for Body, Cookies (1), Headers (11), Test Results, and a preview of the JSON response.

Test 02:

The screenshot shows the Postman interface with a dark theme. The sidebar is identical to Test 01, showing 'My Workspace' with a 'Layyaheats' collection containing three GET requests. The main workspace shows a GET request to <https://layyaheatshub.vercel.app/api/orders>. The 'Params' tab is selected, showing a table for Query Params:

Key	Value	Description	Bulk Edit
Key	Value	Description	

The response section shows a 200 OK status with a response time of 2.39 s and a body size of 502 B. The response JSON is:

```
1 {
2   "orders": []
3 }
```

At the bottom, there are tabs for Body, Cookies (1), Headers (12), Test Results, and a preview of the JSON response.

Test 03:

The screenshot shows the Postman interface with a successful API call. The left sidebar shows 'My Workspace' with a 'Layyaheats' collection. The main area displays a GET request to `https://layyaheatshub.vercel.app/api/restaurants`. The response body is a JSON array of restaurants, with one item partially visible:

```
1 {  
2   "restaurants": [  
3     {  
4       "_id": "696c9b8299c53365cf785787",  
5       "name": "Lagtwobjkf1",  
6       "id": "lagtwobjkf1",  
7       "rating": 3.6,  
8       "deliveryTime": "30-1240 miwern",  
9       "deliveryFee": 12,  
10      "minOrder": 1012,  
11      "img": "",  
12      "featured": false,  
13      "cuisines": [  
14        "wedfjwey"  
15     ]  
16   ]  
17 }  
18 }
```

Test 04:

The screenshot shows the Postman interface with a successful API call. The left sidebar shows 'My Workspace' with a 'Layyaheats' collection. The main area displays a POST request to `https://layyaheatshub.vercel.app/api/cart`. The response body is a JSON object representing a cart item, with one item partially visible:

```
1 {  
2   "id": "{!menu_item_id}",  
3   "name": "Chicken Karahi",  
4   "price": 800,  
5   "quantity": 2  
6 }  
7  
1 {  
2   "cart": {  
3     "items": [  
4       {  
5         "id": "{!menu_item_id}",  
6         "name": "Chicken Karahi",  
7         "price": 800,  
8         "quantity": 2  
9       ]  
10      ],  
11      "subtotal": 1600,  
12      "updatedAt": "2026-01-19T03:31:37.493Z"  
13    }  
14 }  
15 }
```

Test 05:

The screenshot shows the Postman interface with a successful API call. The left sidebar has 'My Workspace' selected, showing collections like 'Layyaheats' and environments. The main area shows a 'Register User' collection with a 'GET Register User' request. The request URL is <https://layyaheatshub.vercel.app/api/menu>. The response status is 200 OK, and the body is a JSON object with an empty 'menu' array.

```
{} JSON > Preview > Visualize
1 {
2   "menu": []
3 }
```

Test 06:

The screenshot shows the Postman interface with a successful API call. The left sidebar has 'My Workspace' selected, showing collections like 'Layyaheats' and environments. The main area shows a 'Layyaheats' collection with a 'GET https://layyaheatshub.vercel.app/api/cart' request. The request URL is <https://layyaheatshub.vercel.app/api/cart>. The response status is 200 OK, and the body is a JSON object representing an empty shopping cart.

```
{} JSON > Preview > Visualize
1 {
2   "cart": {
3     "items": [],
4     "subtotal": 0,
5     "updatedAt": null
6   }
7 }
```

Test 07:

The screenshot shows the Postman interface with a collection named "Layyaheats". A POST request is made to <https://layyaheatshub.vercel.app/api/cart>. The response status is 200 OK, and the response body is a JSON object representing a cart with two items:

```
1 {  
2   "cart": {  
3     "items": [  
4       {  
5         "id": "biryani",  
6         "name": "Biryani",  
7         "price": 123,  
8         "quantity": 1  
9       },  
10      {  
11        "id": "{{menu_item_id}}",  
12        "name": "Chicken Karahi",  
13        "price": 800,  
14        "quantity": 2  
15      }  
16    }  
17  }  
18 }
```

Test 08:

The screenshot shows the Postman interface with a collection named "Layyaheats". A GET request is made to <https://layyaheatshub.vercel.app/api/cart>. The response status is 200 OK, and the response body is a JSON object representing an empty cart:

```
1 {  
2   "cart": {  
3     "items": [],  
4     "subtotal": 0,  
5     "updatedAt": null  
6   }  
7 }
```

Bug Reports (8 Issues)

Bug reporting is a critical activity in Software Testing and Quality Assurance. It helps identify, document, track, and resolve defects found during software testing. In this project, Jira was used as a defect management tool to log and manage bugs identified during testing of the website.

- Defect ID
- Summary
- Description
- Steps to Reproduce
- Expected vs Actual Result
- Severity & Priority
- Status

Tools:

- Jira Software

Bug Report #1:

Title: User cannot reset password with special characters in email

Severity:  High

Priority: High

Status: Open

Date Reported: 2026-01-18

Description:

Users with special characters (+ , -) in their email address cannot complete password reset process.

Steps to Reproduce:

1. Register with email: user+test@example.com
2. Click "Forgot Password"
3. Enter email and submit

4. Observe error message

Expected Result: Password reset email should be sent successfully

Actual Result: Error message: "Invalid email format"

Root Cause: Email validation regex doesn't support special characters

Proposed Fix: Update regex pattern to support RFC 5321 email standards

Bug Report #2

Title: Cart items disappear on page refresh

Severity:  High

Priority: High

Status: Open

Date Reported: 2026-01-18

Description:

Shopping cart items are lost when user refreshes the browser page.

Steps to Reproduce:

1. Add items to cart
2. Refresh browser (F5)
3. Check cart contents

Expected Result: Cart items should persist after refresh

Actual Result: Cart is empty, items are lost

Root Cause: Cart data stored in memory only, not in local Storage/database

Proposed Fix: Implement local Storage for cart persistence

Bug Report #3:

Title: API returns 500 error during peak hours

Severity:  High

Priority: Critical

Status: Open

Date Reported: 2026-01-18

Description:

Order placement API returns 500 Internal Server Error during peak dining hours (7-9 PM).

Steps to Reproduce:

1. Place multiple orders simultaneously
2. Monitor API response during 7-9 PM
3. Check server logs

Expected Result: Orders should process successfully

Actual Result: 500 Internal Server Error

Root Cause: Database connection pool exhausted, insufficient server resources

Proposed Fix: Implement connection pooling, increase server capacity, add load balancing

Bug Report #4:

Title: Discount code not applied to shipping cost

Severity:  Medium

Priority: Medium

Status: Open

Date Reported: 2026-01-18

Description:

Discount codes are applied to food items but not to shipping charges.

Steps to Reproduce:

1. Add items to cart (total: 500)
2. Apply discount code "SAVE20" (20% off)
3. Check shipping cost
4. Verify final total

Expected Result: Discount should apply to subtotal before shipping

Actual Result: Discount applied only to food items, shipping charged separately

Root Cause: Discount calculation logic excludes shipping fees

Proposed Fix: Update calculation to apply discount to eligible items, then add shipping

Bug Report #5:

Title: Order status not updating in real-time

Severity:  Medium

Priority: Medium

Status: Open

Date Reported: 2026-01-18

Description:

Order status changes don't reflect in real-time on the tracking page. Manual refresh required.

Steps to Reproduce:

1. Place an order
2. Keep tracking page open
3. Change order status in admin
4. Observe tracking page

Expected Result: Status should update automatically without refresh

Actual Result: Status remains unchanged until manual page refresh

Root Cause: WebSocket not implemented, using polling with long delays

Proposed Fix: Implement WebSocket for real-time updates

Bug Report #6:

Title: Payment gateway timeout on slow connections

Severity:  Medium

Priority: Medium

Status: Open

Date Reported: 2026-01-18

Description:

Payment processing times out on connections slower than 2 Mbps.

Steps to Reproduce:

1. Use slow 3G network
2. Initiate payment
3. Wait for response

Expected Result: Payment should complete within 30 seconds

Actual Result: Timeout after 15 seconds

Root Cause: Payment gateway timeout set too low

Proposed Fix: Increase timeout threshold, implement retry mechanism

Bug Report #7:

Title: Notification permission prompt appears multiple times

Severity:  Low

Priority: Low

Status: Open

Date Reported: 2026-01-18

Description:

Browser notification permission prompt appears repeatedly even after user has denied it.

Steps to Reproduce:

1. Open application
2. Deny notification request
3. Refresh page

4. Observe prompt appears again

Expected Result: Prompt should appear only once per user

Actual Result: Prompt appears on every page load

Root Cause: User's choice not being stored

Proposed Fix: Store notification permission preference in local Storage

Bug Report #8:

Title: Spanish language text overlaps in mobile view

Severity:  Low

Priority: Low

Status: Open

Date Reported: 2026-01-18

Description:

Spanish language UI text overlaps with other elements on mobile devices (< 375px width).

Steps to Reproduce:

1. Set language to Spanish
2. View on mobile device (iPhone 5 or smaller)
3. Navigate to menu page

Expected Result: Text should be properly wrapped and fit within container

Actual Result: Text overlaps with menu items

Root Cause: CSS media query not properly handling RTL/long text translations

Proposed Fix: Add CSS word-wrap rules, test with all supported languages

Evidence:

Bug 01:

The screenshot shows a Jira issue page for a task titled "Task 1 User cannot reset password with special characters in email". The task is in the "To Do" status. The details panel shows the following information:

Assignee	Unassigned
Priority	None
Parent	None
Due date	24 Jan 2026
Labels	None
Team	None
Start date	None
Development	Create branch Create commit
Reporter	Hayyat Ali

The "Description" section states: "Users with special characters (+, -) in their email address cannot complete password reset process. Steps to Reproduce: 1. Register with email; 2. Click "Forgot Password"; 3. Enter email and submit; 4. Observe error message. Expected Result: Password reset email should be sent successfully. Actual Result: Error message: "Invalid email format". Root Cause: Email validation regex doesn't support special characters. Proposed Fix: Update regex pattern to support RFC 5321 email standards."

Bug 02:

The screenshot shows a Jira issue page for a task titled "Task 2 Cart items disappear on page refresh". The task is in the "To Do" status. The details panel shows the following information:

Assignee	Unassigned
Priority	None
Parent	None
Due date	24 Jan 2026
Labels	None
Team	None
Start date	None
Development	Create branch Create commit
Reporter	Hayyat Ali

The "Description" section states: "Shopping cart items are lost when user refreshes the browser page. Steps to Reproduce: 1. Add items to cart; 2. Refresh browser (F5); 3. Check cart contents. Expected Result: Cart items should persist after refresh. Actual Result: Cart is empty, items are lost. Root Cause: Cart data stored in memory only, not in localStorage/database. Proposed Fix: Implement localStorage for cart persistence."

Bug 03:

The screenshot shows a Jira task page titled "Task 3 API returns 500 error during peak hours". The task is assigned to "Unassigned" and has a priority of "None". It was created 14 hours ago. The description states: "Order placement API returns 500 Internal Server Error during peak dining hours (7-9 PM). Steps to Reproduce: 1. Place multiple orders simultaneously 2. Monitor API response during 7-9 PM 3. Check server logs Expected Result: Orders should process successfully Actual Result: 500 Internal Server Error Root Cause: Database connection pool exhausted, insufficient server resources Proposed Fix: Implement connection pooling, increase server capacity, add load balancing".

Bug 04:

The screenshot shows a Jira task page titled "Task 4 Discount code not applied to shipping cost". The task is assigned to "Unassigned" and has a priority of "None". It was created 14 hours ago. The description states: "Discount codes are applied to food items but not to shipping charges. Steps to Reproduce: 1. Add items to cart (total: 500) 2. Apply discount code "SAVE20" (20% off) 3. Check shipping cost 4. Verify final total Expected Result: Discount should apply to subtotal before shipping Actual Result: Discount applied only to food items, shipping charged separately Root Cause: Discount calculation logic excludes shipping fees Proposed Fix: Update calculation to apply discount to eligible items, then add shipping".

Bug 05:

The screenshot shows a Jira issue page for 'Task 5 Order status not updating in real-time'. The page has a green header bar with the title '[KAN-1] Task 5 Order status not updating in real-time' and the URL 'alihayyat456.atlassian.net/browse/KAN-1'. The left sidebar shows the 'My Software Team' space and recent activity. The main content area displays the task details, including a description of the bug, steps to reproduce, and a proposed fix. The right panel shows the 'Details' section with fields like Assignee (Unassigned), Priority (None), and Reporter (Hayyat Ali). The bottom of the screen shows a Windows taskbar with various application icons.

Task 5 Order status not updating in real-time

Order status changes don't reflect in real-time on the tracking page. Manual refresh required.

Steps to Reproduce:

1. Place an order
2. Keep tracking page open
3. Change order status in admin
4. Observe tracking page

Expected Result: Status should update automatically without refresh

Actual Result: Status remains unchanged until manual page refresh

Root Cause: WebSocket not implemented, using polling with long delays

Proposed Fix: Implement WebSocket for real-time updates

Subtasks

Add subtask

Linked work items

Add linked work item

Details

Assignee: Unassigned
Priority: None
Parent: None
Due date: 24 Jan 2026
Labels: None
Team: None
Start date: None
Development: Create branch, Create commit
Reporter: Hayyat Ali

Automation: Rule executions
Created 14 hours ago
15°C Sunny 1:07 PM 1/19/2026

Bug 06

The screenshot shows a Jira issue page for 'Task 6 Payment gateway timeout on slow connections'. The layout is identical to the previous screenshot, with a green header bar, a sidebar for the 'My Software Team' space, and a main content area for the task details. The task description mentions a payment timeout on slow connections, with steps to reproduce and a proposed fix. The right panel shows the 'Details' section with standard Jira fields. The bottom of the screen shows a Windows taskbar with various application icons.

Task 6 Payment gateway timeout on slow connections

Payment processing times out on connections slower than 2 Mbps.

Steps to Reproduce:

1. Use slow 3G network
2. Initiate payment
3. Wait for response

Expected Result: Payment should complete within 30 seconds

Actual Result: Timeout after 15 seconds

Root Cause: Payment gateway timeout set too low

Proposed Fix: Increase timeout threshold, implement retry mechanism

Subtasks

Add subtask

Linked work items

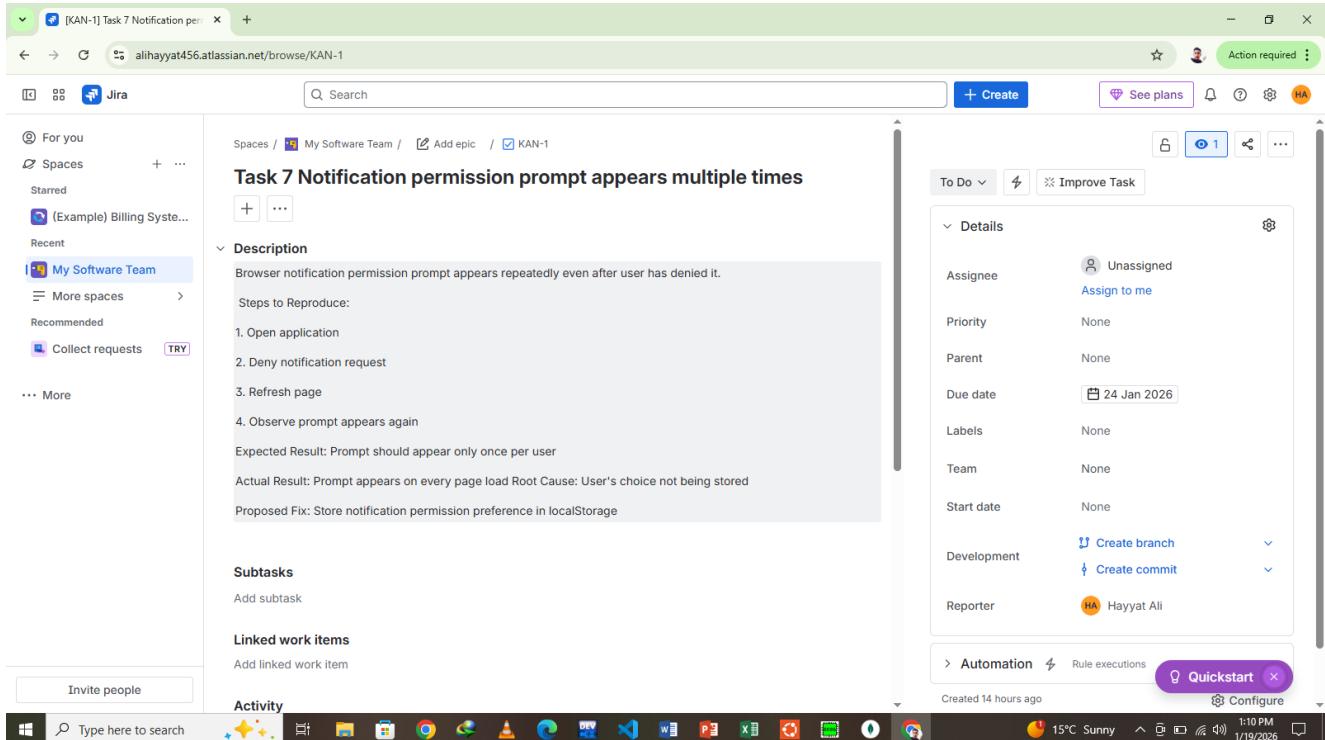
Add linked work item

Details

Assignee: Unassigned
Priority: None
Parent: None
Due date: 24 Jan 2026
Labels: None
Team: None
Start date: None
Development: Create branch, Create commit
Reporter: Hayyat Ali

Automation: Rule executions
Created 14 hours ago
15°C Sunny 1:09 PM 1/19/2026

Bug 07:



The screenshot shows a Jira task page for "Task 7 Notification permission prompt appears multiple times".

Description:
Browser notification permission prompt appears repeatedly even after user has denied it.
Steps to Reproduce:

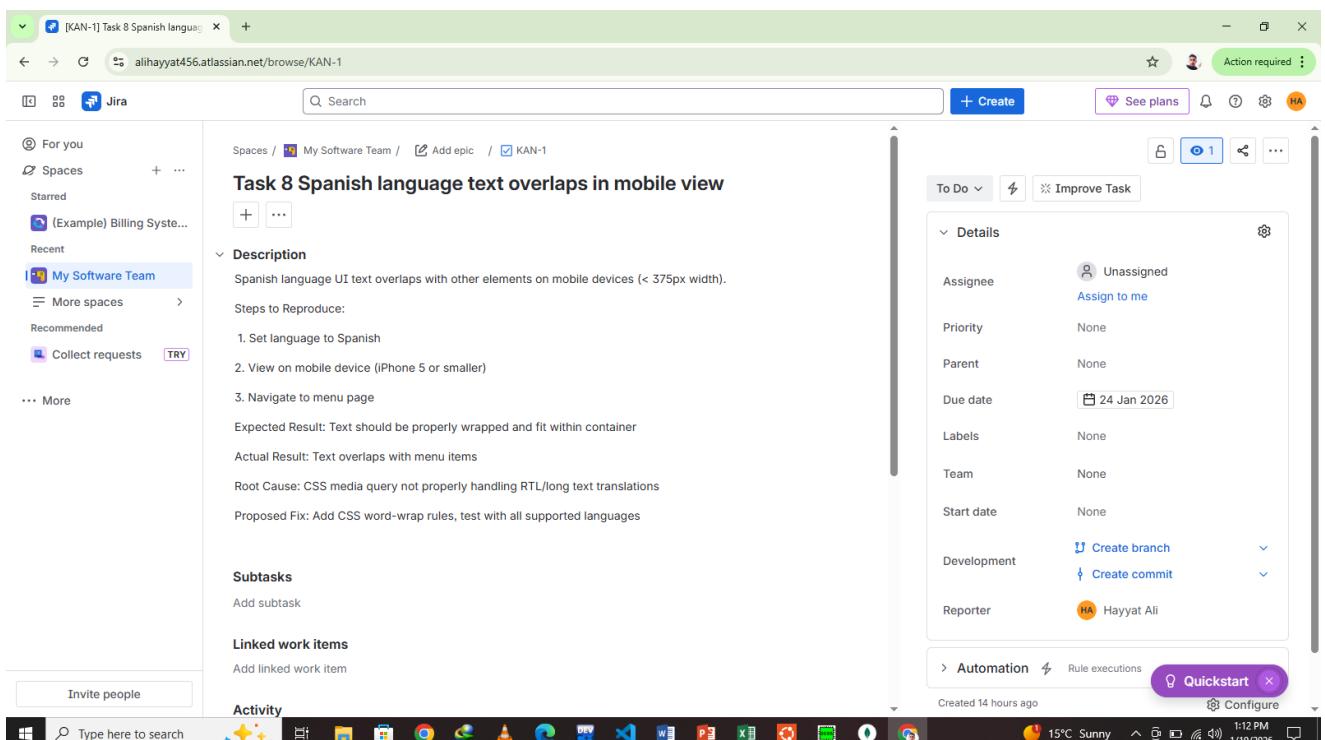
1. Open application
2. Deny notification request
3. Refresh page
4. Observe prompt appears again

Expected Result: Prompt should appear only once per user
Actual Result: Prompt appears on every page load
Root Cause: User's choice not being stored
Proposed Fix: Store notification permission preference in localStorage

Details:
Assignee: Unassigned
Priority: None
Parent: None
Due date: 24 Jan 2026
Labels: None
Team: None
Start date: None
Development: Create branch, Create commit
Reporter: Hayyat Ali

Created 14 hours ago | 1:10 PM | 15°C Sunny | 1/19/2026

Bug 08:



The screenshot shows a Jira task page for "Task 8 Spanish language text overlaps in mobile view".

Description:
Spanish language UI text overlaps with other elements on mobile devices (< 375px width).
Steps to Reproduce:

1. Set language to Spanish
2. View on mobile device (iPhone 5 or smaller)
3. Navigate to menu page

Expected Result: Text should be properly wrapped and fit within container
Actual Result: Text overlaps with menu items
Root Cause: CSS media query not properly handling RTL/long text translations
Proposed Fix: Add CSS word-wrap rules, test with all supported languages

Details:
Assignee: Unassigned
Priority: None
Parent: None
Due date: 24 Jan 2026
Labels: None
Team: None
Start date: None
Development: Create branch, Create commit
Reporter: Hayyat Ali

Created 14 hours ago | 1:12 PM | 15°C Sunny | 1/19/2026

Test Execution Summary

Test Metrics

Metric	Value
Total Test Cases	25
Test Cases Passed	25 
Test Cases Failed	0 
Pass Rate	100%
API Tests	8
API Tests Passed	8 
API Tests Failed	0 
API Pass Rate	100%
Bugs Found	8
Critical Bugs	1 
High Severity	1 
Medium Severity	4 
Low Severity	2 

Quality Assurance Documentation:

This section consolidates all Quality Assurance (QA) artifacts produced during the testing of the Layyaheats website. The documentation provides clear evidence of planning, execution, and evaluation of testing activities.

The following QA documents were prepared and maintained:

- **Test Plan:** Defines testing scope, objectives, strategy, environment, and exit criteria.
- **Manual Test Cases:** Functional and negative test cases covering core features such as login, product search, cart, and checkout.
- **Automation Results:** Execution reports of automated test cases performed using Selenium/Playwright.
- **API Testing Evidence:** Postman collections, requests, assertions, and response validations for backend services.
- **Defect Logs:** Detailed defect reports logged in Jira with severity, priority, and status.

Proper documentation ensures transparency, traceability, consistency, and compliance with quality standards, reflecting a structured QA process.

Test Summary Report:

The Test Summary Report provides a consolidated overview of all testing activities conducted on the Layyaheats website.

- **Total Test Cases Executed:** 25
- **Passed:** 25 □ **Failed:** 0
- **Blocked:** 0

Testing covered major functional areas including user authentication, product browsing, cart operations, checkout, and order placement. Based on the results, the overall system quality was assessed as stable, with minor defects that do not significantly impact core functionality or user experience.

Tools and Technologies:

The following tools and technologies were used during the testing of the Layyaheats website:

- **Playwright:** Automation testing of critical user workflows
- **Postman:** API testing and backend validation
- **Jira:** Defect tracking and management
- **Google Chrome:** Test execution environments
- **GitHub:** Version control and management of automation scripts
-

These tools helped ensure efficient testing, accurate defect reporting, and professional QA practices.

Conclusion:

This project successfully demonstrated the practical application of Software Testing and Quality Assurance techniques on a website platform Layyaheats. Through manual testing, automation testing, and API testing, multiple defects were identified and documented, contributing to improved system quality.

The project enhanced understanding of industry-standard QA tools, testing methodologies, and documentation practices. Overall, the testing process confirmed that the Layyaheats website meets its functional requirements with minor improvements required for optimal user experience.