Test Document

for

Quick Crave

Version 1.0

Prepared by: Cache Me If You Can (Team 9)

Dnyanesh Hemendra Shinde	230390	dnyanesh23@iitk.ac.in
Nikhilesh Joshi	230700	nikhileshj23@iitk.ac.in
Kamatam Akshay Reddy	230517	areddy23@iitk.ac.in
Arihant Kumar	230189	arihantkmr23@iitk.ac.in
Kshitij Gupta	230581	kshitijg23@iitk.ac.in
Parnika Mittal	230736	mparnika23@iitk.ac.in
Rohit Vijaykumar Somani	230871	rohitvs23@iitk.ac.in
Suvid Goel	231065	gsuvid23@iitk.ac.in
Yashaswi Raj	231191	yashaswir23@iitk.ac.in
Yatharth Dangi	231197	yatharth23@iitk.ac.in

Course: CS253

Mentor TA: *Nij Bharatkumar Padariya*

Submission Date: *April 05, 2025*

Contents

1 Introduction

1.1 Test Strategy

All components of the system were manually tested to verify correctness and functionality. Each feature was individually executed and its output was validated against expected results to ensure reliable performance.

1.2 Testing Timeline

1.3 Testers

The testing process was carried out internally by our development team. However, to maintain objectivity and reduce potential bias, we ensured that the developer who implemented a specific feature was not the one responsible for testing that particular functionality. This approach helped us maintain a level of separation between development and testing, thereby enhancing the reliability and credibility of our test results.

1.4 Coverage Criteria

We used branch coverage as our test coverage criterion to ensure that all possible branches in the code were executed at least once during testing. This helps in identifying untested paths and improves the overall reliability of the system.

1.5 Tools used for testing

Postman was used to extensively test and validate the backend APIs.

2 Unit Testing

2.1 Authentication

2.1.1 Registering a Customer

API Endpoint: /auth/signup
Test Owner: Nikhilesh Joshi
Date: 29/03/2025
Test Description: This test checks if the API correctly accepts the details of the customer and proceeds to verification.
Test Result: The customer is able to proceed to OTP verification page after providing all his details. If his username, phone number or email has already been used earlier, an appropriate message is shown. An OTP is sent to the given email and the user is redirected to the OTP Verification Page.

2.1.2 Email Verification by OTP

API Endpoint: /auth/otp-verification
Test Owner: Nikhilesh Joshi
Date: 30/03/2025
Test Description: This test checks if the API correctly signs up a customer if the OTP entered is correct.
Test Result: The customer is able to successfully signup and his details are entered in the database if the OTP entered is correct. However, after 3 incorrect attempts, the user is redirected to sign-up page.

2.1.3 Registering a Canteen

API Endpoint: `auth/signup`
Test Owner: Kshitij Gupta
Date: 30/3/2025
Test Description: This test checks whether canteen is registered
Test Result: This API works as expected

2.1.4 Logging in

API Endpoint:/auth/login
Test Owner: Yatharth Dangi
Date: 29/03/2025
Test Description: This test checks if the API correctly if the user is able to Login with his Username and password.
Test Result: Allows the user to login and then redirects him to the respective home page. If incorrect password is entered, then an appropriate message is shown.

2.2 Customer Side Functionalities

Accessible by the User

NOTE: In unit testing, the canteen_id is sent in the body of a post request, but in the final web app, it will be sent via an http-only cookie and not inside the body of the request

2.2.1 Search Canteen

API Endpoint: `/customer-search-canteen`
Test Owner: Arihant Kumar
Date: 29/03/2025
Test Description: This test checks if the API correctly returns a list of canteens based on the customer's search query
Test Result: A list of 5 canteens are returned whose names are the closest to the search query made by the customer.

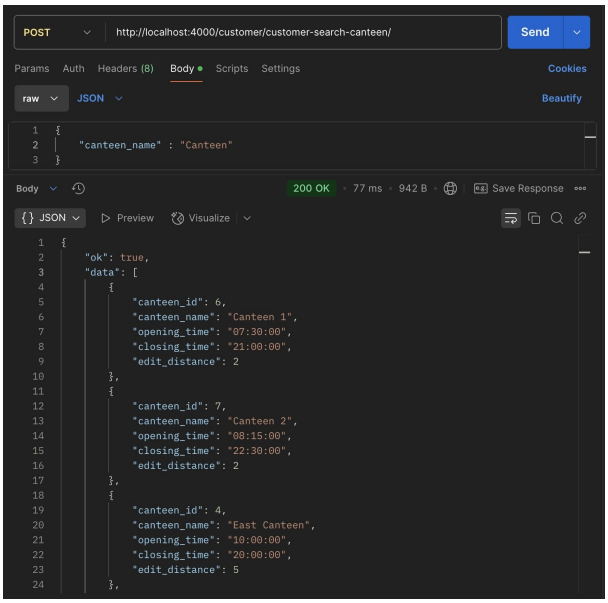

Figure 1: Searching for Canteens

2.2.2 Search Dish

API Endpoint: `/customer-search-dish`
Test Owner: Dnyanesh Hemendra Shinde
Date: 29/03/2025
Test Description: This test checks if the API correctly returns a list of dishes matching the customer's search input
Test Result: A list of 5 dishes are returned whose names are the closest to the search query made by the customer.

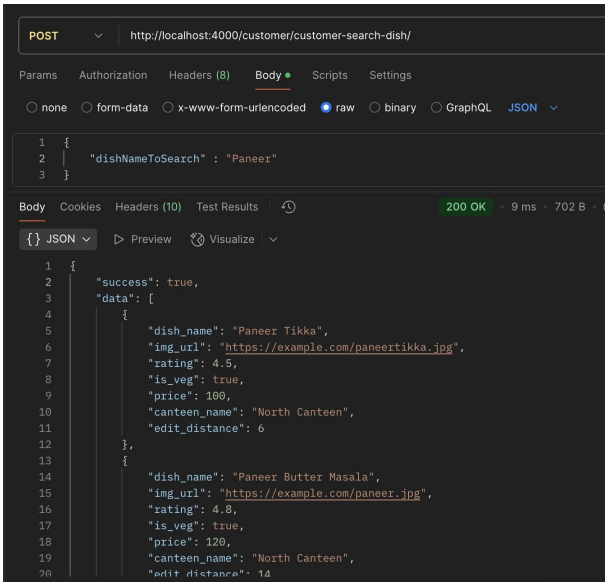

Figure 2: Searching by Dish Name

2.2.3 Fetch Reservations

API Endpoint: `/customer-fetch-reservations`
Test Owner: Parnika Mittal
Date: 29/03/2025
Test Description: This test checks if the API returns all active and past reservation details of the logged-in customer
Test Result: Returns the details of the all the reservations that have been made my the customer, active as well as present.

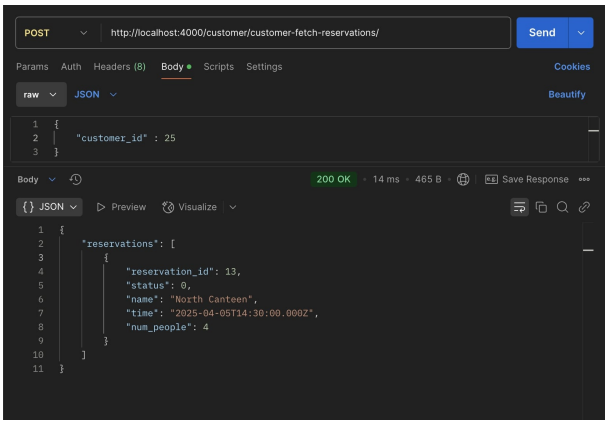

Figure 3: List of Reseravtions

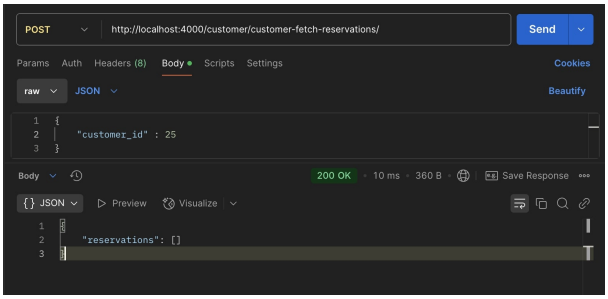

Figure 4: No Reservations

2.2.4 Add Dish To Cart

API Endpoint: `/customer-add-dish`
Test Owner: Kamatam Akshay Reddy
Date: 29/03/2025
Test Description: This test checks if the API correctly adds a selected dish to the customer's cart
Test Result: Adds the dish selected to the cart of the customer and makes appropriate changes in the database as well. The current price of the cart is also updated. It creates a new cart if no previous cart exists, otherwise adds to the existing cart only if the current dish being added is of the same canteen as the dishes in the cart. An appropriate message is shown.

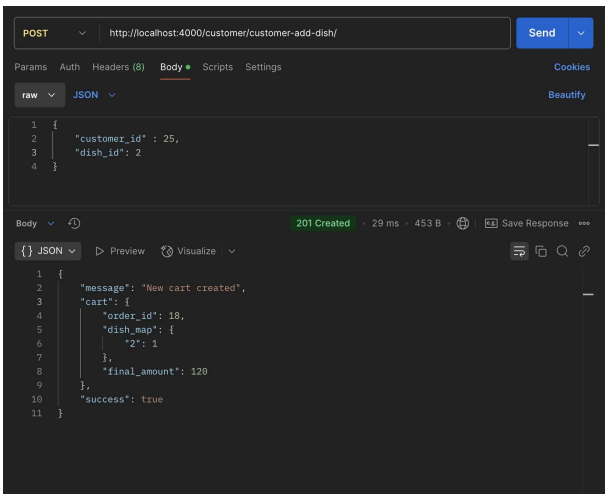

Figure 5: Adding Dish to a New Cart

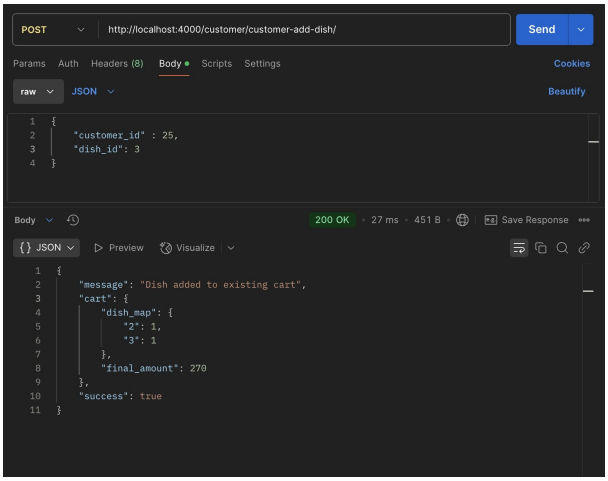

Figure 6: Adding Dish to an Existing Cart

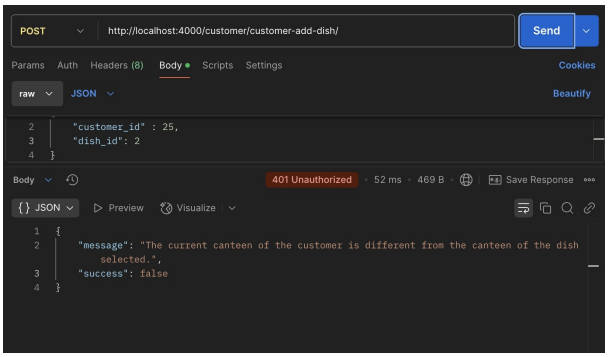

Figure 7: When dishes are from different Canteens

2.2.5 Remove Dish From Cart

API Endpoint: `/customer-remove-dish`
Test Owner: Kshitij Gupta
Date: 29/03/2025
Test Description: This test checks if the API successfully removes a specific dish from the customer's cart
Test Result: Removes the dish selected from the cart of the customer and makes appropriate changes in the database as well. If the cart of the user is now empty, then appropriate changes are made to the database.

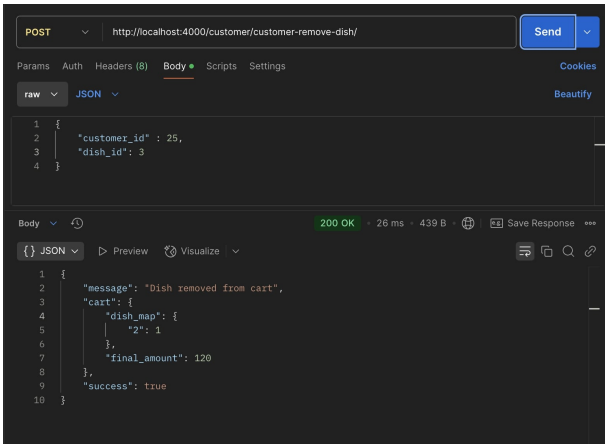

Figure 8: Removing Dish from Cart

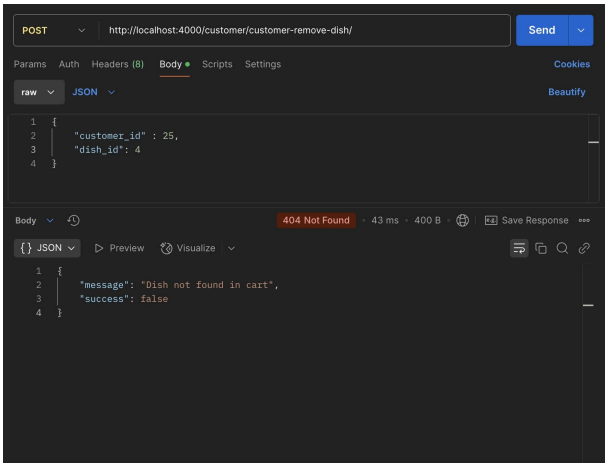

Figure 9: When Dish does not exist

2.2.6 View Order History

API Endpoint: `/customer-view-order-history`
Test Owner: Arihant Kumar
Date: 01/04/2025
Test Description: This test checks if the API correctly returns the customer's complete order history
Test Result: Returns a list of orders made by the customer. The customer can filter the results based on date as well as status of the order.

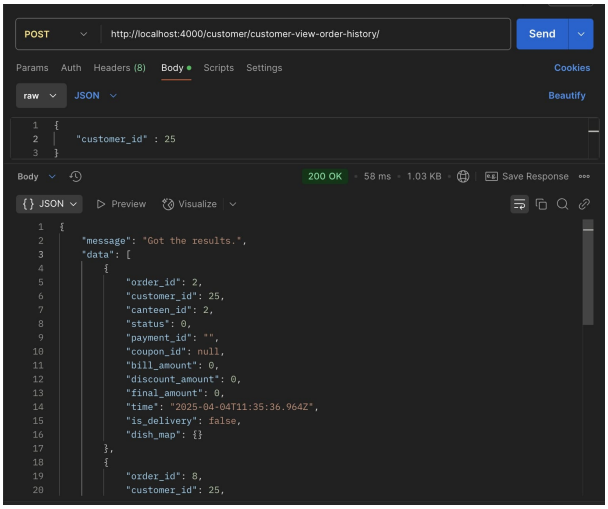

Figure 10: History of the customer

2.2.7 View Favourites

API Endpoint: `/customer-view-favourites`
Test Owner: Suvid Goel
Date: 02/04/2025
Test Description: This test checks if the API returns a list of the customer's favourite dishes
Test Result: Returns a list of the favorite dishes selected by the customer along with all the details.

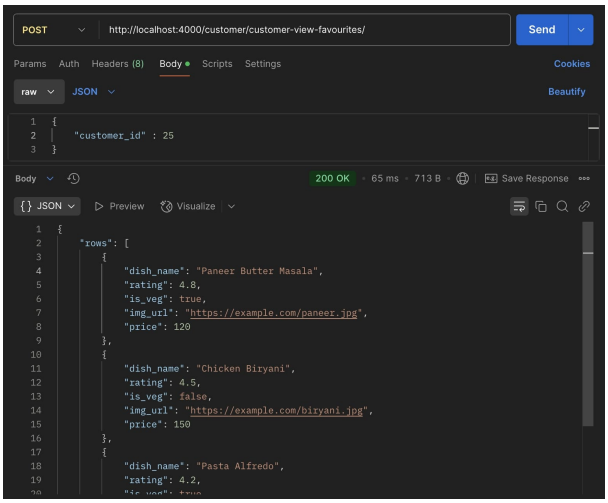

Figure 11: Viewing favourites

2.2.8 Add Favourite

API Endpoint: `/customer-add-favourite`
Test Owner: Yatharth Dangi
Date: 03/04/2025
Test Description: This test checks if the API adds a dish to the customer's favourites successfully
Test Result: Adds the dish selected by the customer to the favorites of the customer and makes appropriate changes to the database.

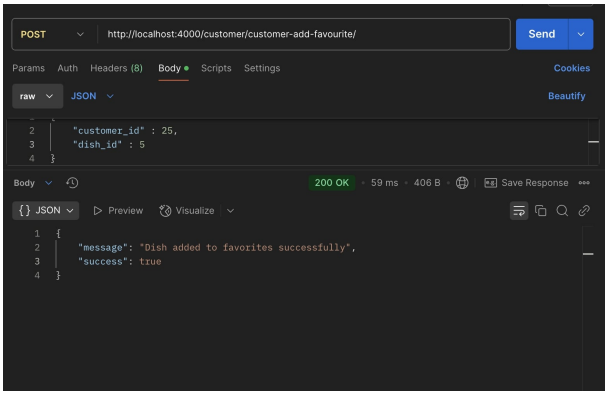

Figure 12: Adding a favourite

2.2.9 View Wallet

API Endpoint: `/customer-wallet`
Test Owner: Rohit Vijaykumar Somani
Date: 04/04/2025
Test Description: This test checks if the API returns the current wallet balance of the customer
Test Result: Returns the current wallet balance of the customer.

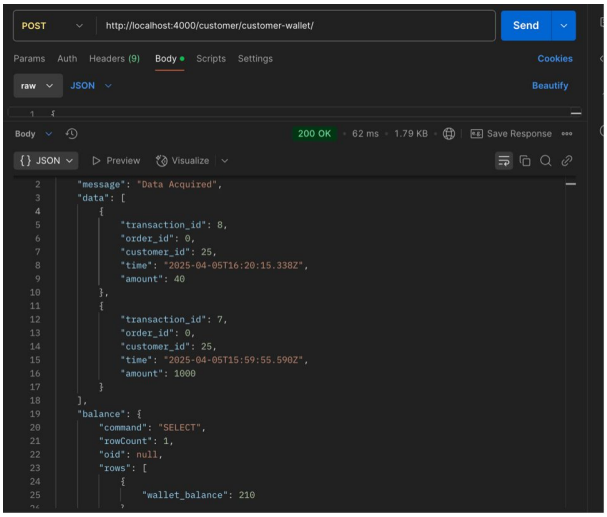

Figure 13: Wallet Balance and Transactions

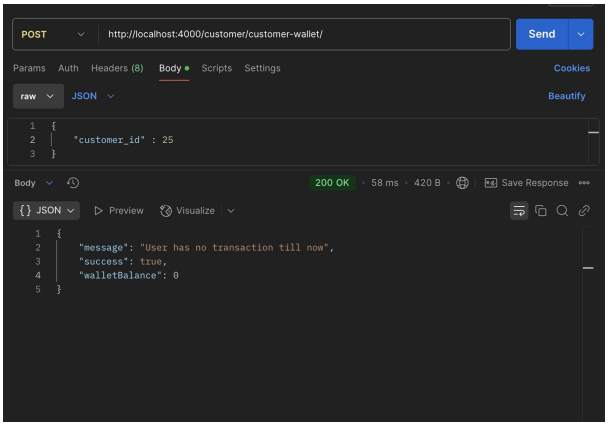

Figure 14: Empty Wallet

2.2.10 Update Wallet

API Endpoint: `/customer-update-wallet`
Test Owner: Yashaswi Raj
Date: 04/03/2025
Test Description: This test checks if the API correctly updates the customer's wallet balance after a transaction
Test Result: After a transaction is made by the customer, the wallet of the customer is updated appropriately and database is also updated.

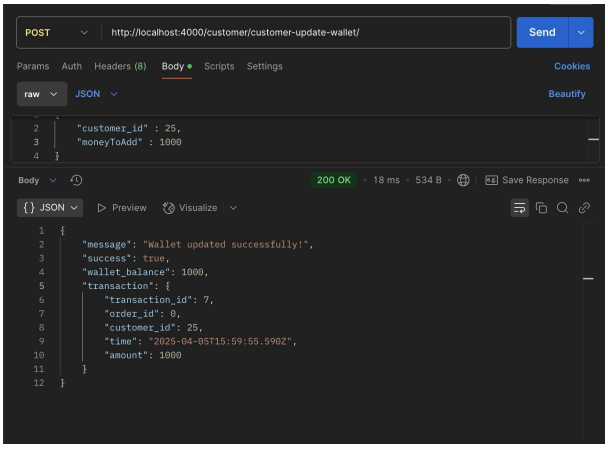

Figure 15: Updating Wallet

2.2.11 Add Reservation

API Endpoint: `/customer-add-reservations`
Test Owner: Arihant Kumar
Date: 04/04/2025
Test Description: This test checks if the API allows the customer to successfully place a reservation request
Test Result: Takes the necessary details from the customer and sends the reservation request to the canteen manager.

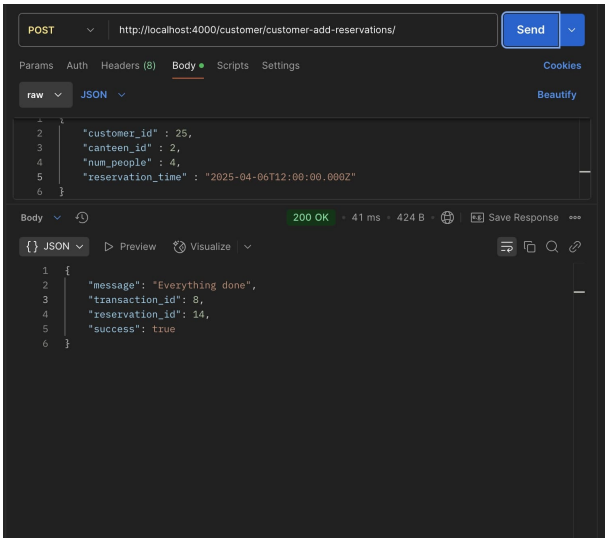

Figure 16: Making a Reservation

2.2.12 View Profile

API Endpoint: `/customer-view-profile`
Test Owner: Dnyanesh Hemendra Shinde
Date: 03/04/2025
Test Description: This test checks if the API returns the complete profile details of the logged-in customer
Test Result: Returns the details of the customer, along with order and reservation data. The details are successfully displayed on the customer profile page.

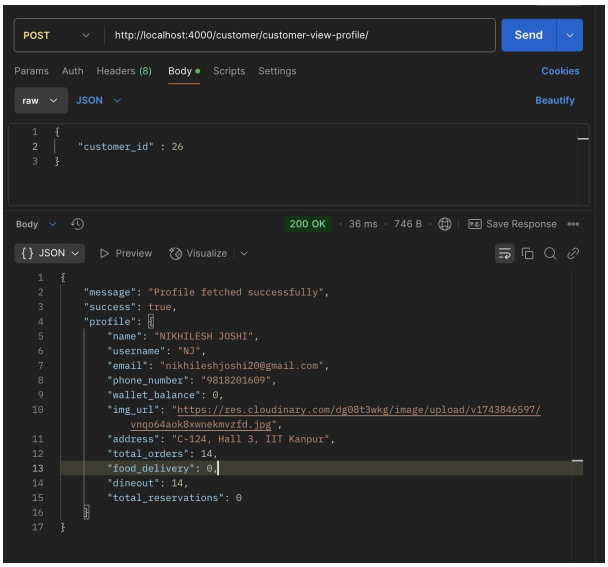

Figure 17: Viewing Profile

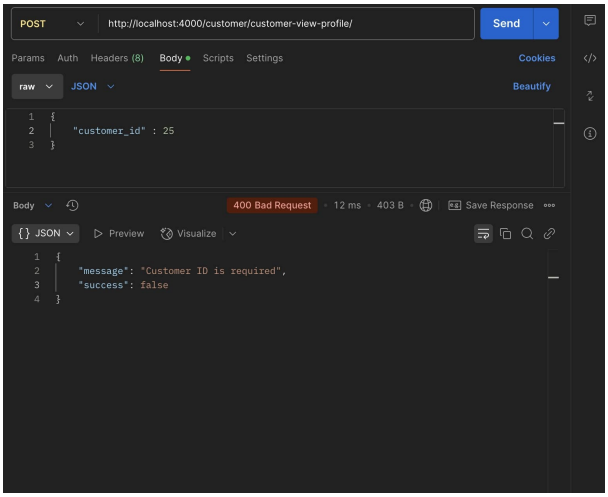

Figure 18: Viewing Profile Edge Case

2.2.13 Fetch Top 5 Dishes

API Endpoint: `/customer-fetch-top5`
Test Owner: Parnika Mittal
Date: 02/04/2025
Test Description: This test checks if the API returns the top 5 most frequently ordered dishes
Test Result: Fetches the top 5 dishes according to the order count. The picture of these dishes, on clicking will redirect to the corresponding canteen's details page.

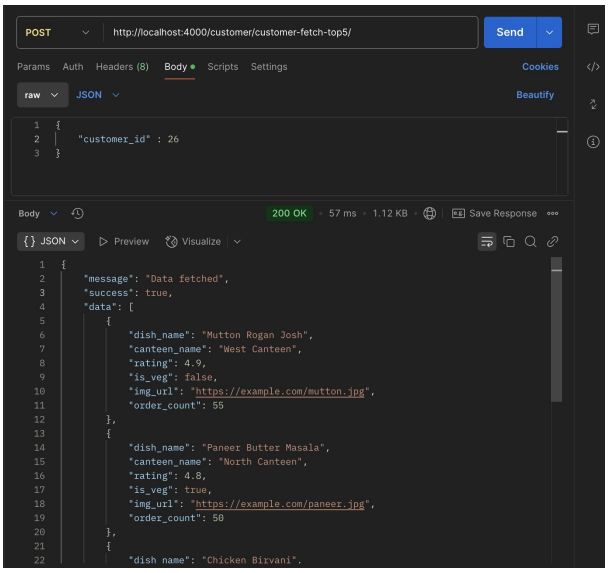

Figure 19: Fetching the Recommended Dishes

2.2.14 Reset Password

API Endpoint: `/customer-reset-password`
Test Owner: Kamatam Akshay
Date: 02/04/2025
Test Description: This test checks if the API allows the customer to reset their password successfully
Test Result: Allows the customer to rest their password, in case they have forgotten their

password. They will provide their username or email address and using the OTP the new password will get updated. The new password must follow the set security standards.

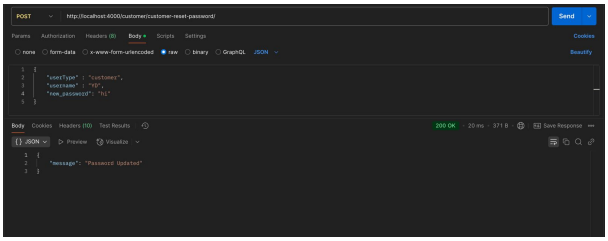

Figure 20: Resetting Password

2.2.15 Update Profile Photo

API Endpoint: `/customer-update-profilephoto`
Test Owner: Kshitij Gupta
Date: 02/04/2025
Test Description: This test checks if the API successfully updates the profile picture of the customer
Test Result: Allows the customer to change their profile photo and make the changes in the database. The image is uploaded and a unique url is fetched and is updated in the backend.

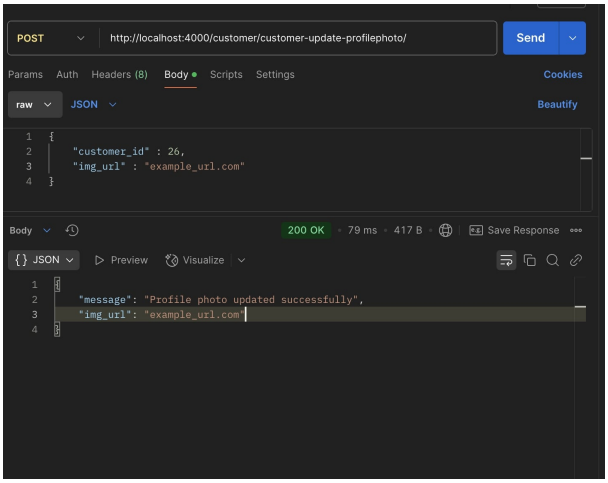

Figure 21: Updating Profile Photo

2.2.16 Update Profile

API Endpoint: `/customer-update-profile`
Test Owner: Nikhilesh Joshi
Date: 31/03/2025
Test Description: This test checks if the API correctly updates the customer's profile information
Test Result: Allows the user to alter some attributes of their profile and make the appropriate changes in the database. They may not change their username.

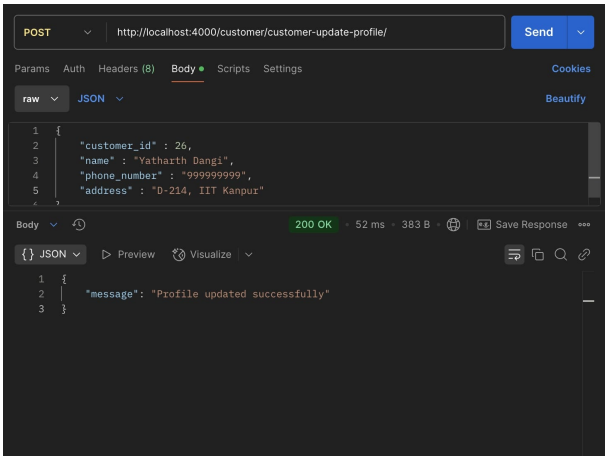

Figure 22: Updating Profile Details

2.2.17 Change Password

API Endpoint: `/customer-change-password`
Test Owner: Suvid Goel
Date: 04/04/2025
Test Description: This test checks if the API allows the customer to successfully change their current password
Test Result: Allows the user to change his password as long as the new password follows the set security standards and they remember their old password.

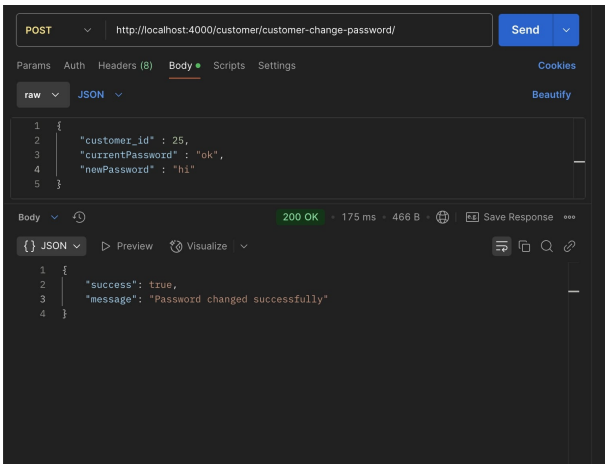

Figure 23: Changing the Password

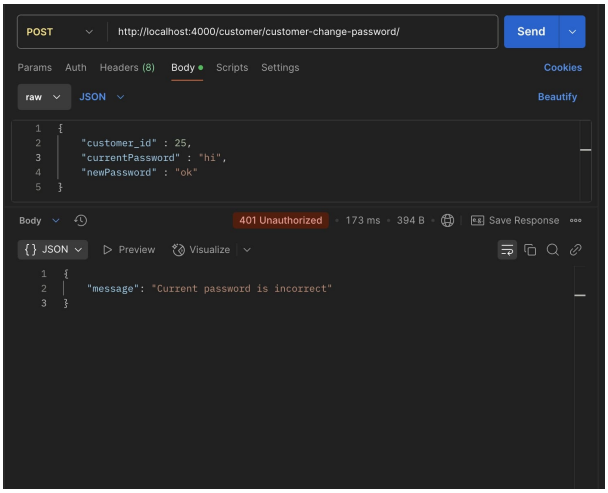

Figure 24: When entered password is wrong

2.2.18 Fetch Dishes

API Endpoint: `/customer-fetch-dishes`
Test Owner: Yatharth Dangi
Date: 03/04/2025
Test Description: This test checks if the API returns a list of dishes available across canteens
Test Result: Returns a list of dishes across all canteens successfully allowing the user to add them to cart or remove them directly without entering a canteen's menu.

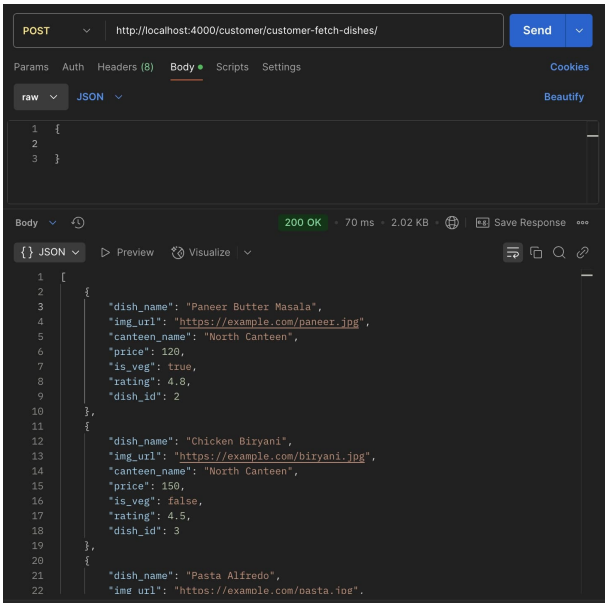

Figure 25: Fetching available dishes

2.2.19 Fetch Canteens

API Endpoint: `/customer-fetch-canteens`
Test Owner: Rohit Vijaykumar Somani
Date: 04/04/2025
Test Description: This test checks if the API returns a list of registered canteens with their

details
Test Result: Returns a list of all registered canteens with their details successfully.

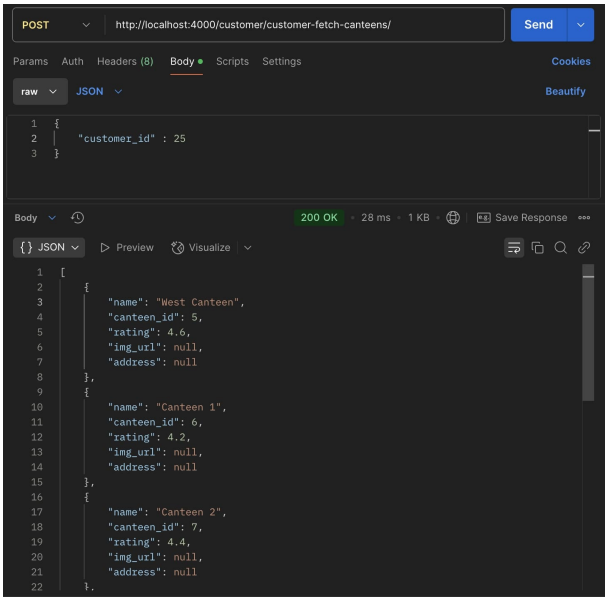

Figure 26: Fetching all registered canteens

2.2.20 Fetch Menu

API Endpoint: `/customer-fetch-menu`
Test Owner: Yashaswi Raj
Date: 02/04/2025
Test Description: This test checks if the API returns the complete menu of a selected canteen
Test Result: Returns a list of food items available in the selected canteen successfully along with their count if they are already added in a cart.

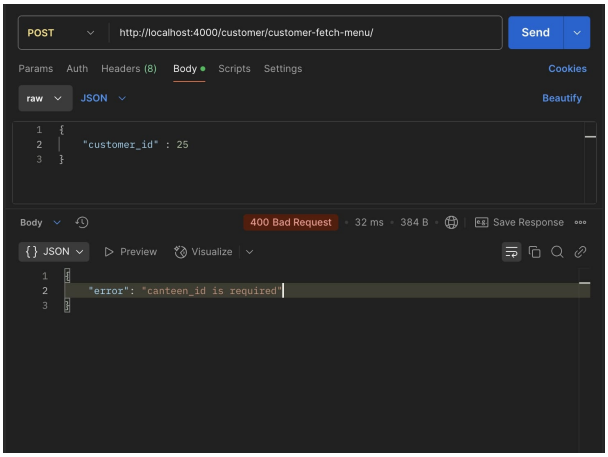

Figure 27: Fetching Menu of a Canteen

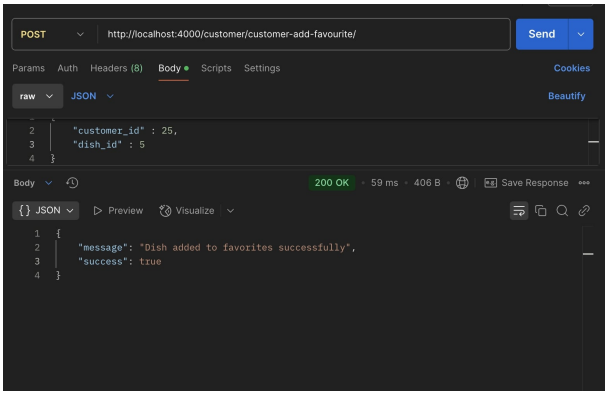

Figure 28: Edge Case

2.2.21 View Cart

API Endpoint: `/customer-view-cart`
Test Owner: Arihant Kumar
Date: 02/04/2025
Test Description: This test checks if the API returns the current cart contents of the customer
Test Result: Returns the details of all dishes currently in the cart of the customer successfully. If the cart is empty then appropriate page is rendered on the site.

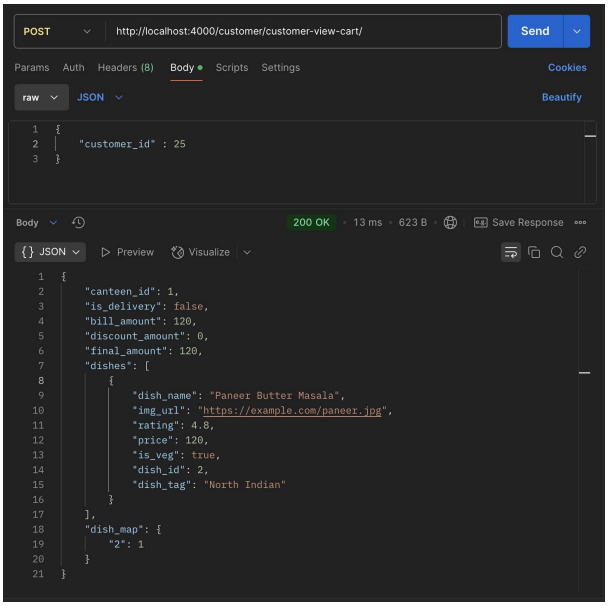

Figure 29: Viewing Cart

2.2.22 Requesting Order

API Endpoint: `/request-order`
Test Owner: Yatharth Dangi
Date: 30/03/2025
Test Description: This test checks if the API places the order and makes necessary changes
Test Result: Places the order at the requested canteen. The amount is deducted from the wallet of the customer, order details have been added to the database and the canteen manager is also informed.

2.2.23 Placing Order

API Endpoint: `/place-order`
Test Owner: Rohit Vijaykumar Somani
Date: 29/03/2025
Test Description: This test checks if the API requests the order from the canteen manager of the corresponding canteen
Test Result: The request is conveyed to the canteen manager of the canteen from which the order is requested. If the auto accept of the canteen manager is on, the order is automatically accepted. Otherwise, if the order is not accepted within 5 minutes of the request, it is automatically canceled.

2.3 Canteen Side Functionalities

Accessible by the Canteen Manager

NOTE: In unit testing, the canteen_id is sent in the body of a post request, but in the final web app, it will be sent via an http-only cookie and not inside the body of the request

2.3.1 Active Orders / Pending Order Requests

API Endpoint: `/canteen/home-order-queue`
Test Owner: Arihant Kumar
Date: 01/04/2025
Test Description: This test checks if the API returns the correct number of active orders and pending order requests of the past 24 hours
Test Result: The API is working as expected

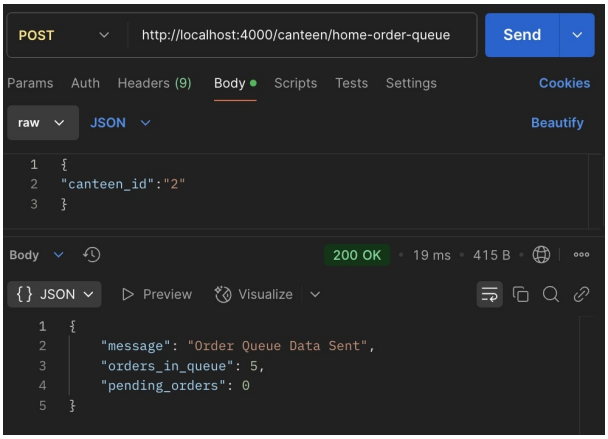

Figure 30: home-order-queue request

2.3.2 Confirmed / Pending Reservation Requests

API Endpoint: `/canteen/home-reservation-queue`
Test Owner: Dnyanesh Hemendra Shinde
Date: 02/04/2025
Test Description: This test checks if the API returns the correct number of confirmed and pending reservation requests of the past week
Test Result: The API is working as expected

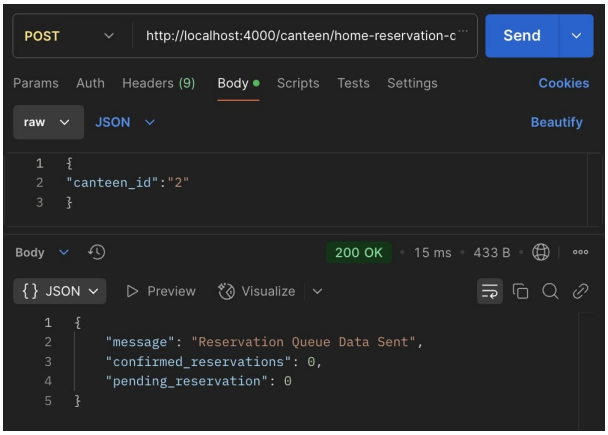

Figure 31: home-reservation-queue request

2.3.3 Active Coupons / Discounts

API Endpoint: `/canteen/home-discount-n-coupons`
Test Owner: Nikhilesh Joshi
Date: 03/04/2025
Test Description: This test checks if the API returns the correct number of currently active discounts and coupons
Test Result: The API is working as expected

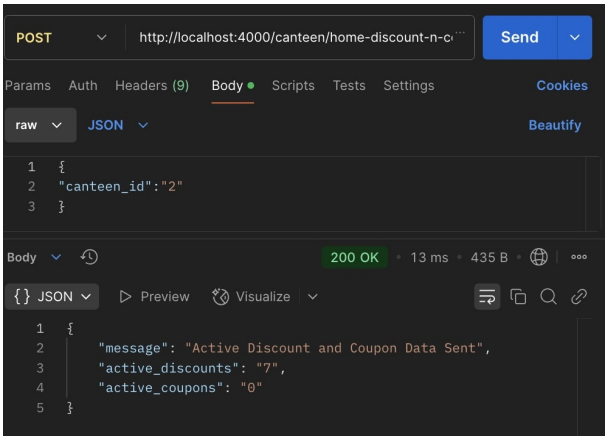

Figure 32: home-discount-n-coupons request

2.3.4 Trending Picks

API Endpoint: `/canteen/trending-picks`
Test Owner: Kamatam Akshay Reddy
Date: 01/04/2025
Test Description: This test checks if the API correctly returns the data of the most ordered dishes from the canteen in the past week
Test Result: The API is working as expected

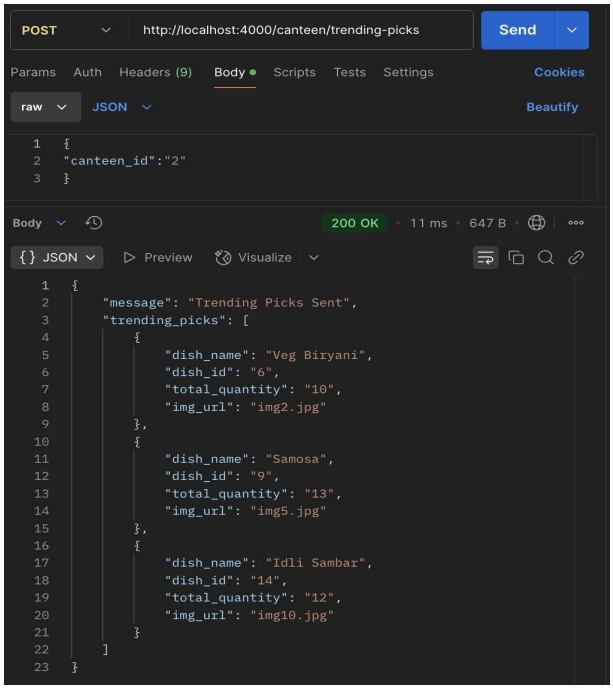

Figure 33: trending-picks request

2.3.5 Order Queue

API Endpoint: `/canteen/order-queue`
Test Owner: Kshitij Gupta
Date: 31/03/2025
Test Description: This test checks if the API correctly returns the data orders received by the canteen in the past 24 hours
Test Result: The API is working as expected

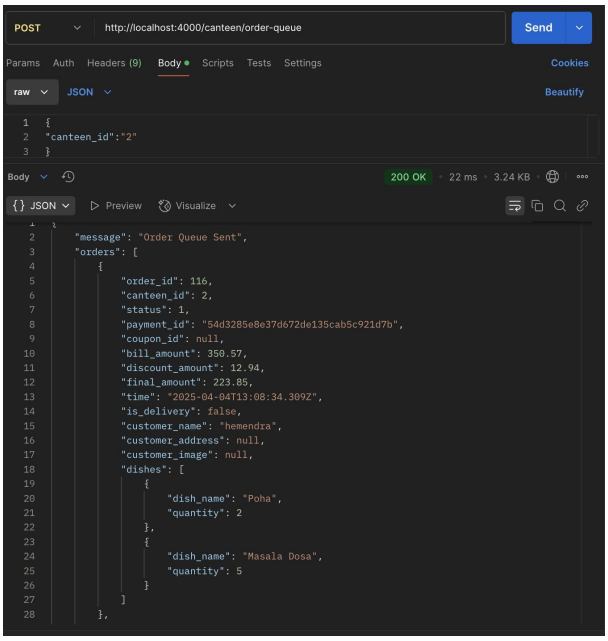

Figure 34: order-queue request

2.3.6 Accept Order

API Endpoint: `/canteen/accept-order`
Test Owner: Rohit Somani
Date: 31/03/2025
Test Description: This test checks if the API successfully updates the status of a given order to 'accepted' and returns a confirmation response
Test Result: The API is working as expected

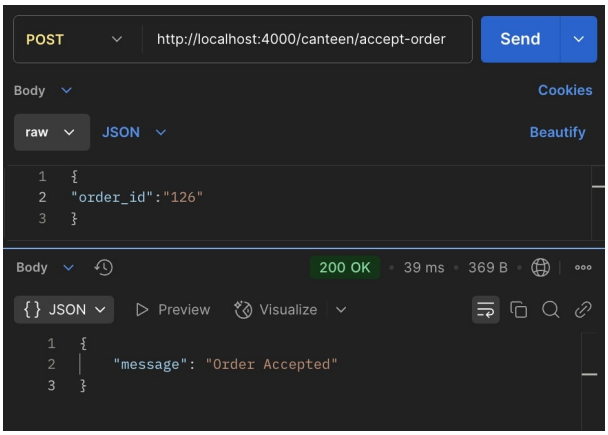

Figure 35: accept-order request

2.3.7 Reject Order

API Endpoint: `/canteen/reject-order`
Test Owner: Yashaswi Raj
Date: 01/04/2025
Test Description: This test checks if the API correctly marks an order as 'rejected' and returns the expected status update
Test Result: The API is working as expected

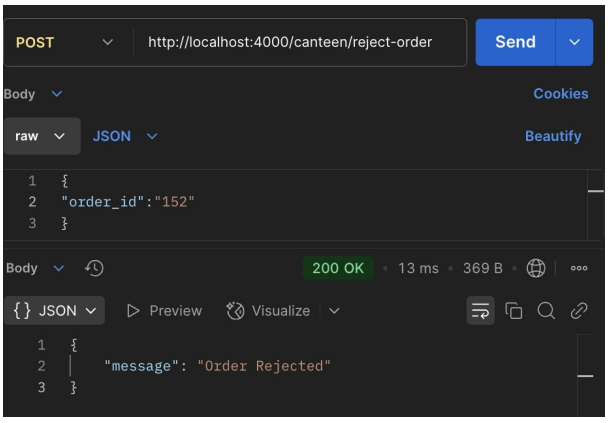

Figure 36: reject-order request

2.3.8 Mark Order As Ready

API Endpoint: `/canteen/ready-order`
Test Owner: Parnika Mittal
Date: 03/04/2025

Test Description: This test checks whether the API marks a previously accepted order as ready and sends a valid response
Test Result: The API is working as expected

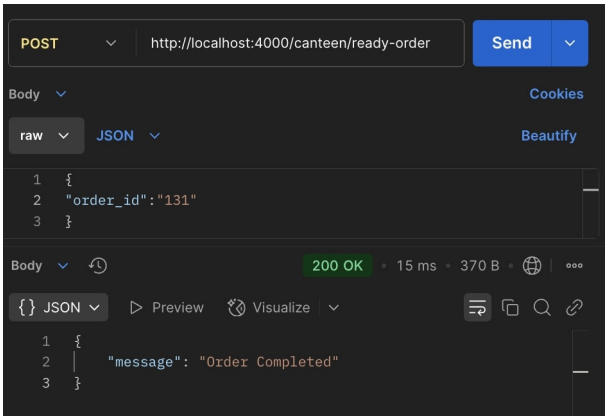

Figure 37: ready-order request

2.3.9 Reservation Queue

API Endpoint: `/canteen/reservation-queue`
Test Owner: Suvid Goel
Date: 30/03/2025
Test Description: This test checks if the API returns all reservation requests that are either pending or confirmed in the last 7 days
Test Result: The API is working as expected

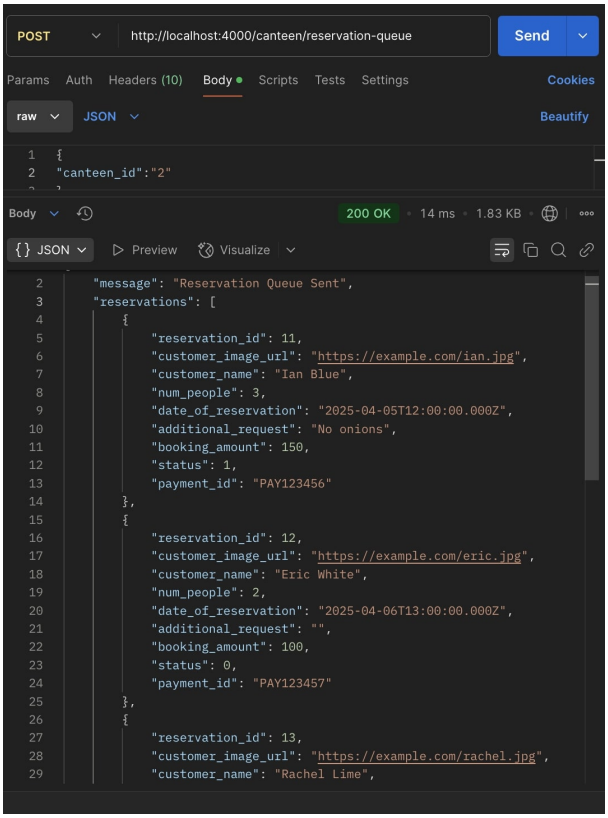

Figure 38: reservation-queue request

2.3.10 Accept Reservation

API Endpoint: `/canteen/accept-reservation`
Test Owner: Yatharth Dangi
Date: 04/04/2025
Test Description: This test checks if the API correctly updates the status of a reservation to 'accepted' and returns a success message
Test Result: The API is working as expected

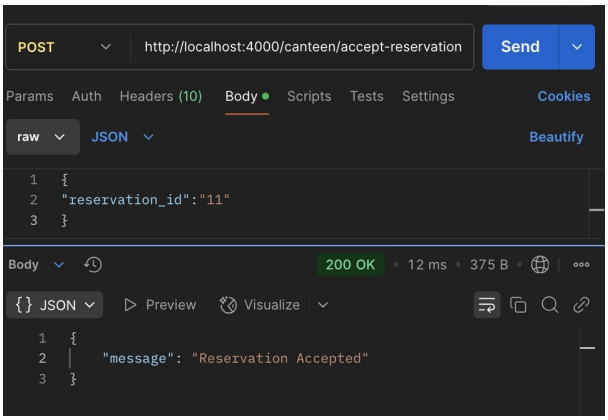

Figure 39: accept-reservation request

2.3.11 Reject Reservation

API Endpoint: `/canteen/reject-reservation`
Test Owner: Kamatam Akshay Reddy
Date: 01/04/2025
Test Description: This test checks whether the API properly sets the reservation status to 'rejected' and returns the appropriate response
Test Result: The API is working as expected

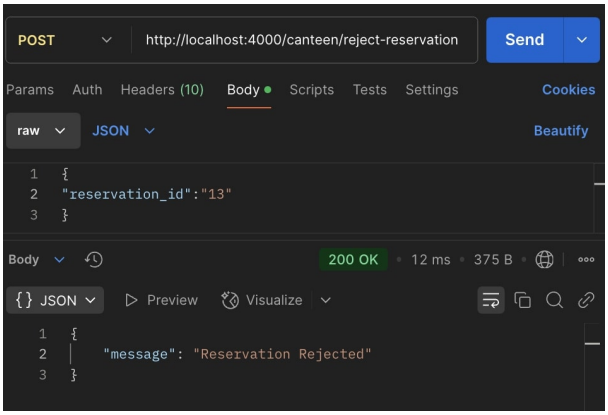

Figure 40: reject-reservation request

2.3.12 See Active Discounts

API Endpoint: `/canteen/active-discounts`
Test Owner: Dnyanesh Hemendra Shinde
Date: 01/04/2025

Test Description: This test checks if the API returns a list of all currently active discounts for the canteen
Test Result: The API is working as expected

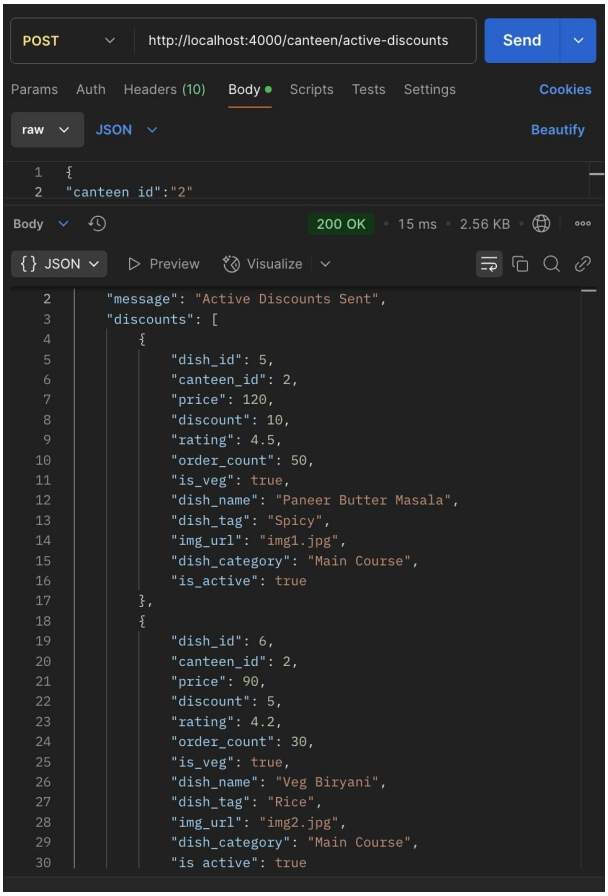

Figure 41: active-discounts request

2.3.13 Create Discounts

API Endpoint: `/canteen/create-discounts`
Test Owner: Yatharth Dangi
Date: 04/04/2025
Test Description: This test checks if the API successfully adds a new discount and validates the input fields
Test Result: The API is working as expected

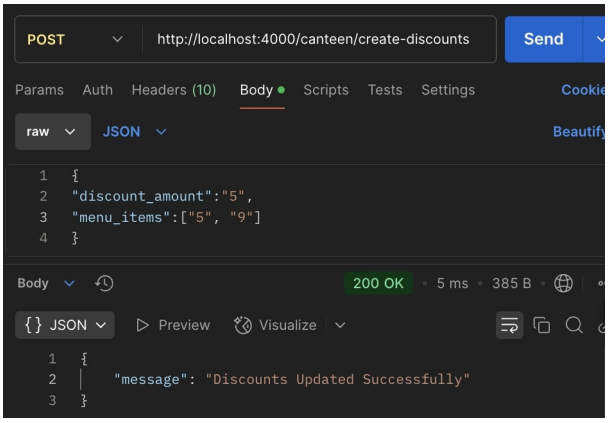

Figure 42: create-discounts request

2.3.14 Delete Discount

API Endpoint: `/canteen/delete-discount`
Test Owner: Kamatam Akshay Reddy
Date: 31/03/2025
Test Description: This test checks whether the API removes the specified discount and confirms deletion in the response
Test Result: The API is working as expected

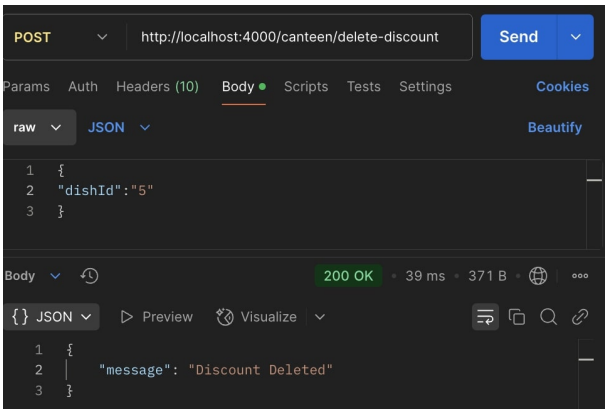

Figure 43: delete-discount request

2.3.15 See Active Coupons

API Endpoint: `/canteen/active-coupons`
Test Owner: Kshitij Gupta
Date: 03/04/2025
Test Description: This test checks if the API returns all the currently available and valid coupons for the canteen
Test Result: The API is working as expected

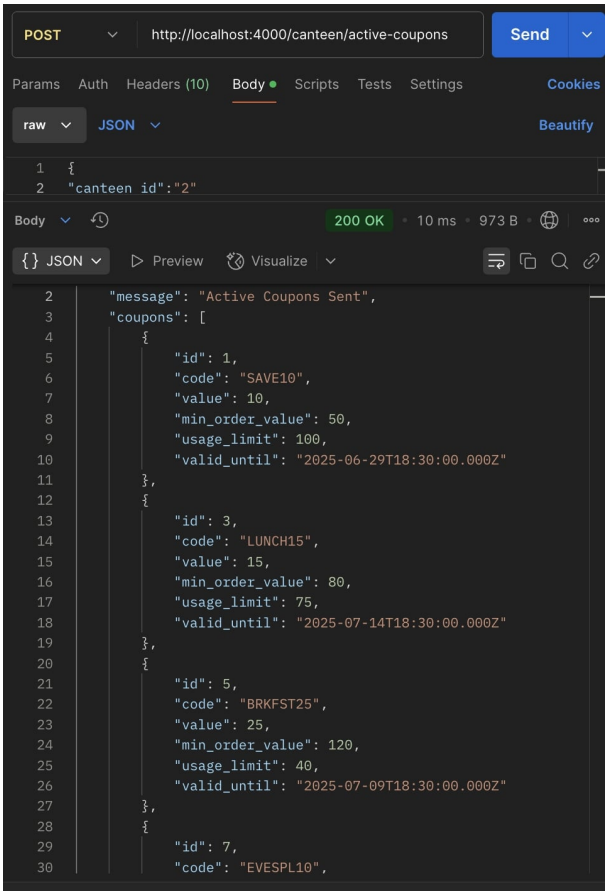

Figure 44: active-coupons request

2.3.16 Create Coupon

API Endpoint: `/canteen/create-coupon`
Test Owner: Parnika Mittal
Date: 02/04/2025
Test Description: This test checks if the API creates a new coupon with the specified parameters and returns success
Test Result: The API is working as expected

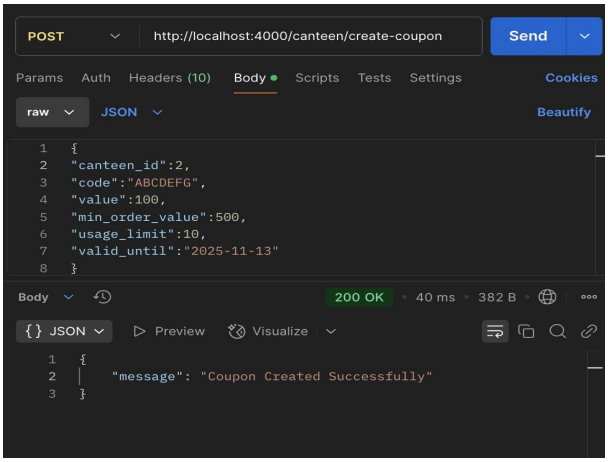

Figure 45: create-coupon request

2.3.17 Delete Coupon

API Endpoint: `/canteen/delete-coupon`
Test Owner: Suvid Goel
Date: 29/03/2025
Test Description: This test verifies whether the API deletes the given coupon entry and returns a confirmation
Test Result: The API is working as expected

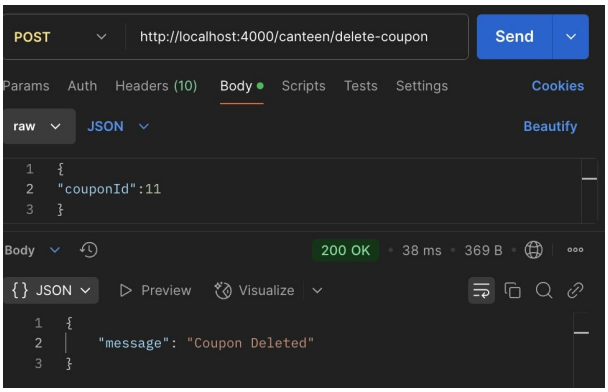

Figure 46: delete-coupon request

2.3.18 Get Profile Details

API Endpoint: `/canteen/profile`
Test Owner: Rohit Somani
Date: 31/03/2025
Test Description: This test checks whether the API fetches the profile details of the currently logged-in canteen
Test Result: The API is working as expected

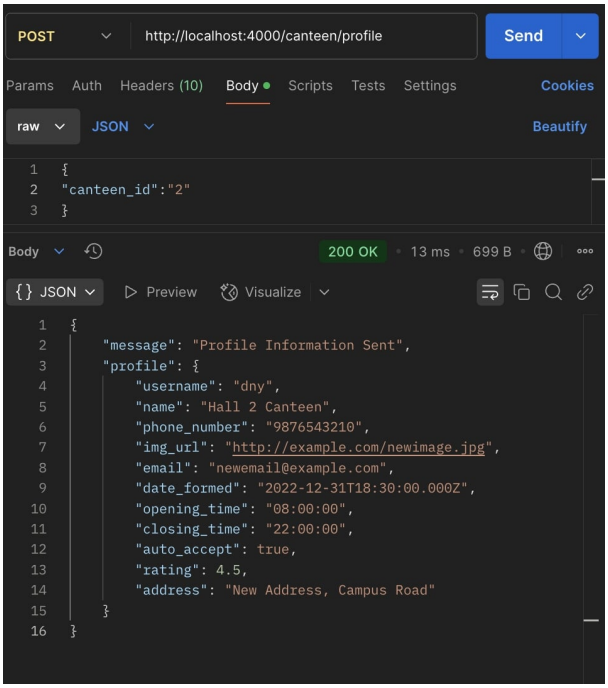

Figure 47: profile request

2.3.19 Edit Profile

API Endpoint: `/canteen/edit-profile`
Test Owner: Yatharth Dangi
Date: 02/04/2025
Test Description: This test checks if the API updates the canteen's profile information and confirms the update
Test Result: The API is working as expected

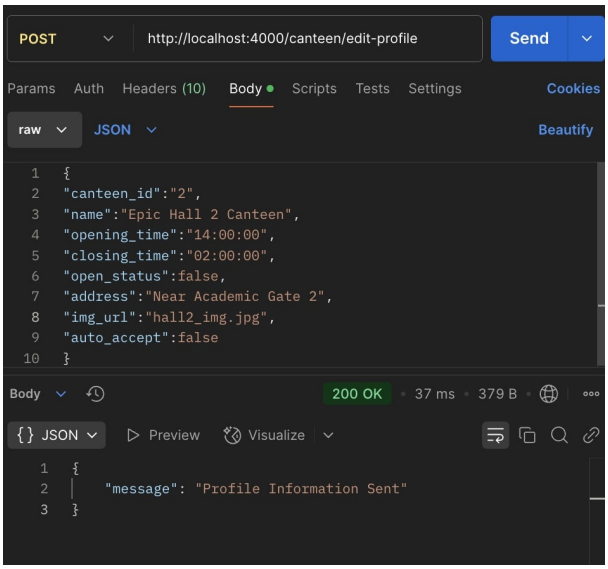

Figure 48: edit-profile request

2.3.20 Edit Password

API Endpoint: `/canteen/edit-password`
Test Owner: Yashaswi Raj
Date: 31/03/2025
Test Description: This test verifies if the API allows the canteen user to securely change their password
Test Result: The API is working as expected

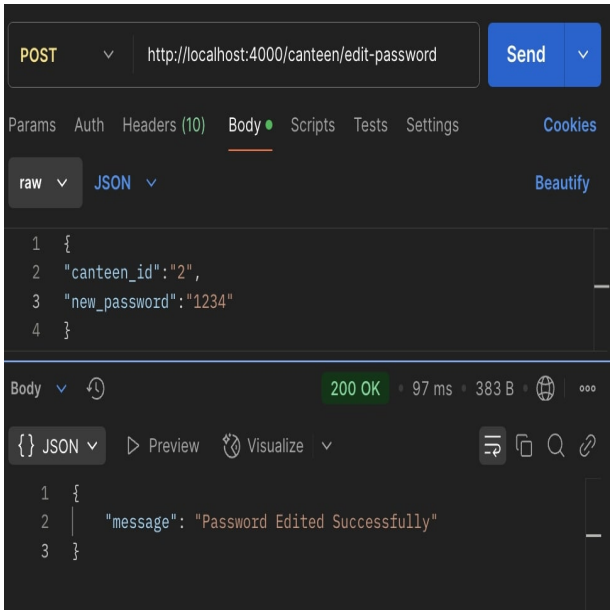

Figure 49: edit-password request

2.3.21 Show Menu Items

API Endpoint: `/canteen/menu-items`
Test Owner: Dnyanesh Hemendra Shinde
Date: 29/03/2025
Test Description: This test checks if the API returns all current dishes available in the canteen menu with its details
Test Result: The API is working as expected

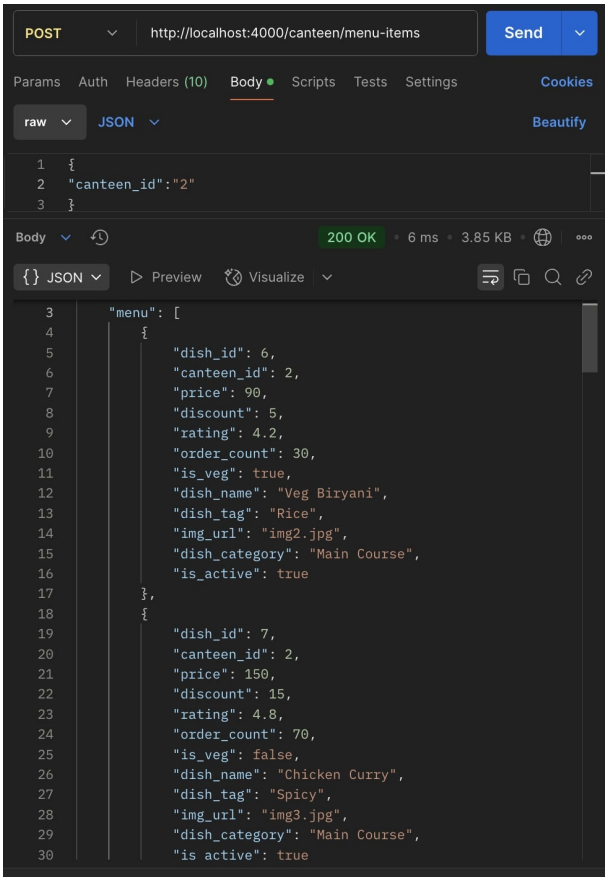

Figure 50: menu-items request

2.3.22 Edit Dish Details

API Endpoint: `/canteen/edit-dish`
Test Owner: Nikhilesh Joshi
Date: 30/03/2025
Test Description: This test checks if the API successfully updates the details of an existing dish and confirms the modification
Test Result: The API is working as expected

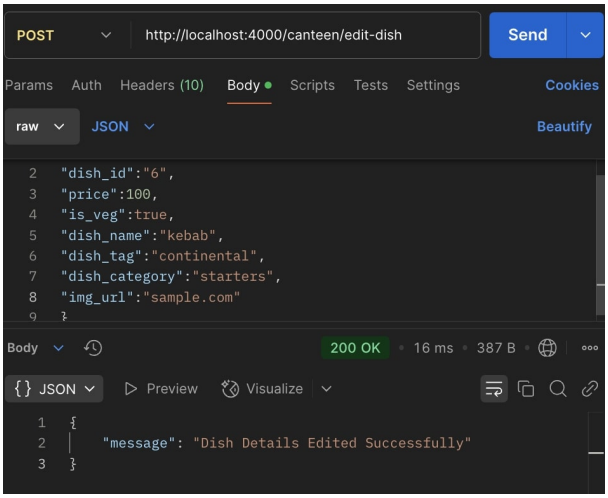

Figure 51: edit-dish request

2.3.23 Add New Dish

API Endpoint: `/canteen/add-dish`
Test Owner: Nikhilesh Joshi
Date: 21/03/2025
Test Description: This test verifies that the API correctly adds a new dish to the menu and returns a success response
Test Result: The API is working as expected

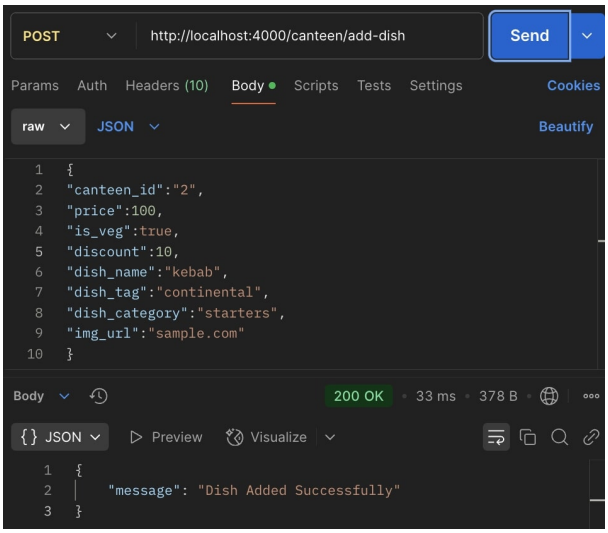

Figure 52: add-dish request

2.3.24 Get Statistics

API Endpoint: `/canteen/get-statistics`
Test Owner: Rohit Somani
Date: 02/04/2025
Test Description: This test checks if the API returns accurate statistical data about sales, orders, average order value and total customers along with their growth rate
Test Result: The API is working as expected

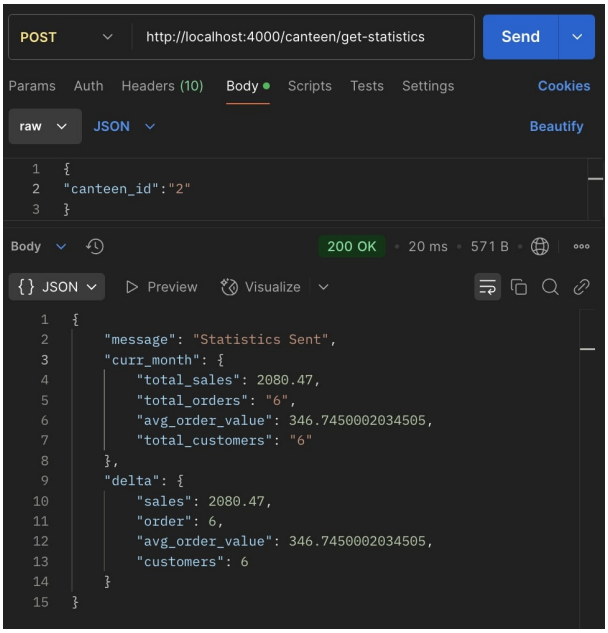

Figure 53: get-statistics request

2.3.25 Get Graph Details

API Endpoint: `/canteen/get-graphs`
Test Owner: Arihant Kumar
Date: 30/03/2025
Test Description: This test checks whether the API provides the required graph data related to orders and sales trends
Test Result: The API is working as expected

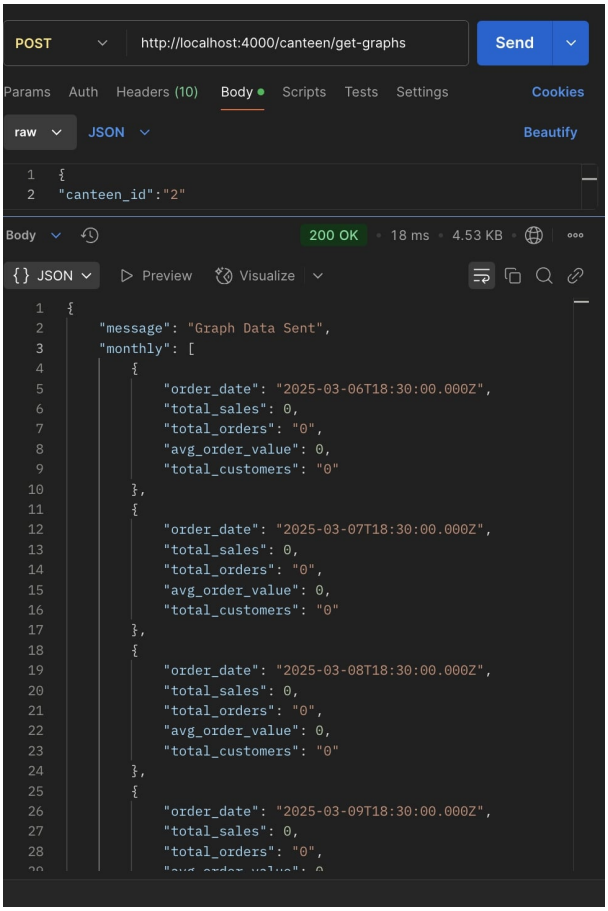

Figure 54: get-graphs request

3 Integration Testing

3.1 Authentication

3.1.1 Registering a customer

Module Details: This module integrates the user registration validation process, ensuring unique usernames and verifying the correctness and strength of user input during account creation
Test Owner: Yashaswi Raj
Test Date: 29/03/2025
Test Results: The integrated module is working as expected

- **Test Case 1:** The system checks that the username entered by the user is not already taken by another user.
  Result: An alert is displayed in red stating *"Username already taken"*

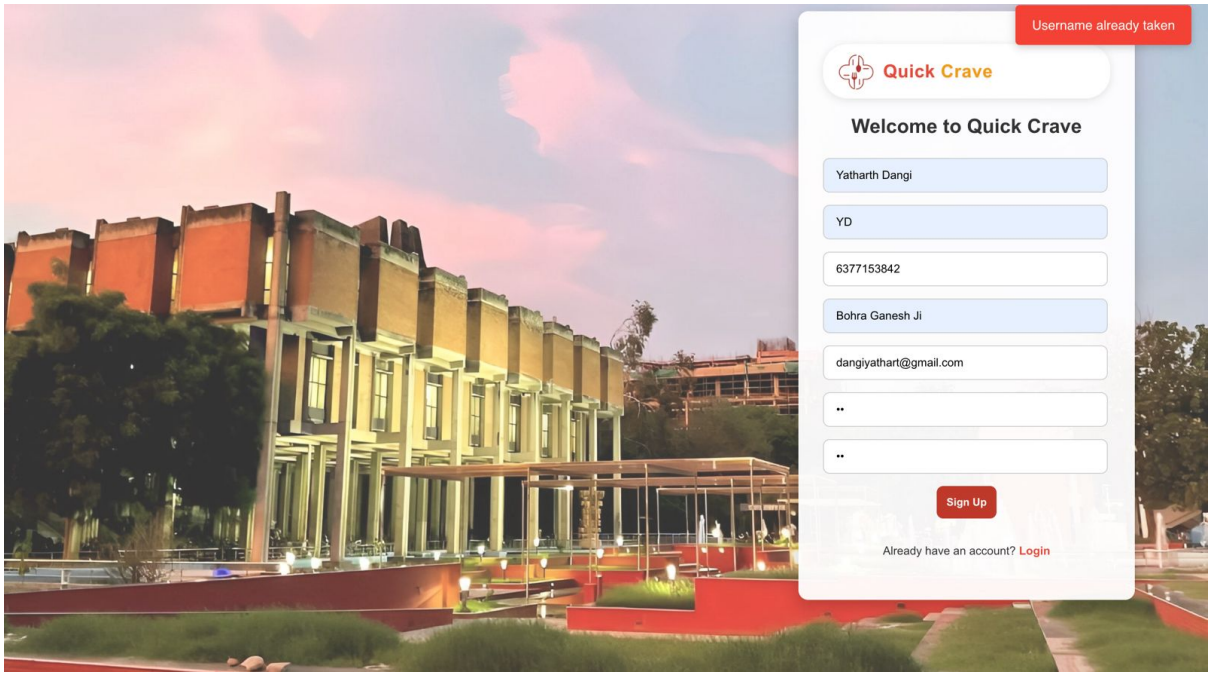

Figure 55: Customer Register user name already taken

- **Test Case 2:** The system checks that the email ID entered by the user is valid
  Result: If the email format is invalid, the system displays the error message *"Enter a valid email address."* beneath the email input field in red text. Examples of invalid formats (e.g., missing '@' or domain) trigger this message

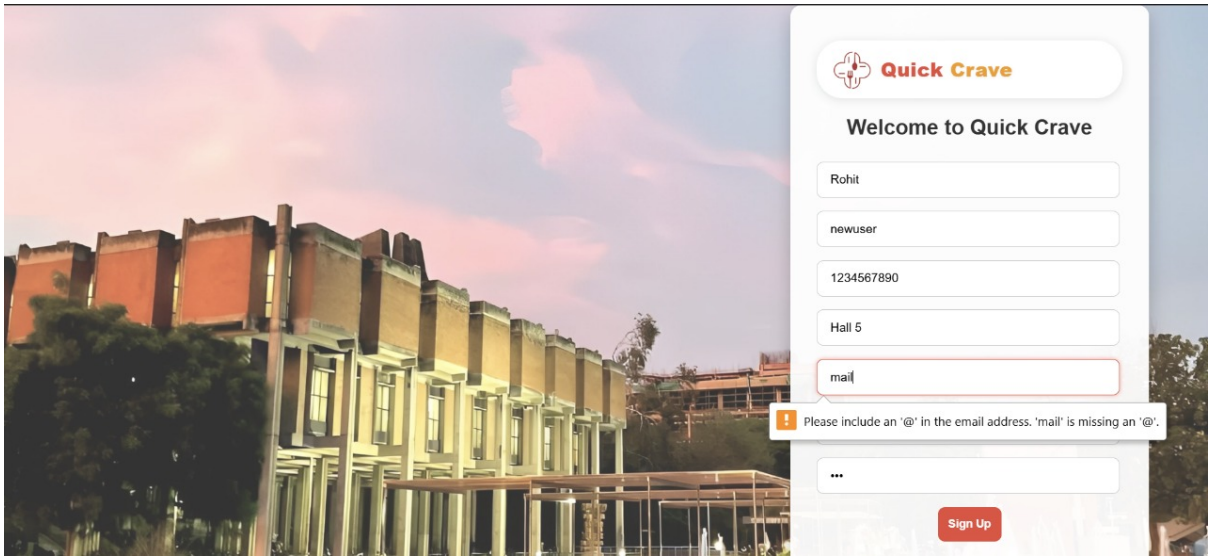

Figure 56: Customer register wrong email id

- **Test Case 3:** The system checks that the phone number entered by the user is valid. System checks if the phone number is of 10 digits.
  Result: If the phone number does not meet the required format (e.g., incorrect digit count or inclusion of alphabets), the error message *"Phone number should be of 10 digits."* is shown in the top right.

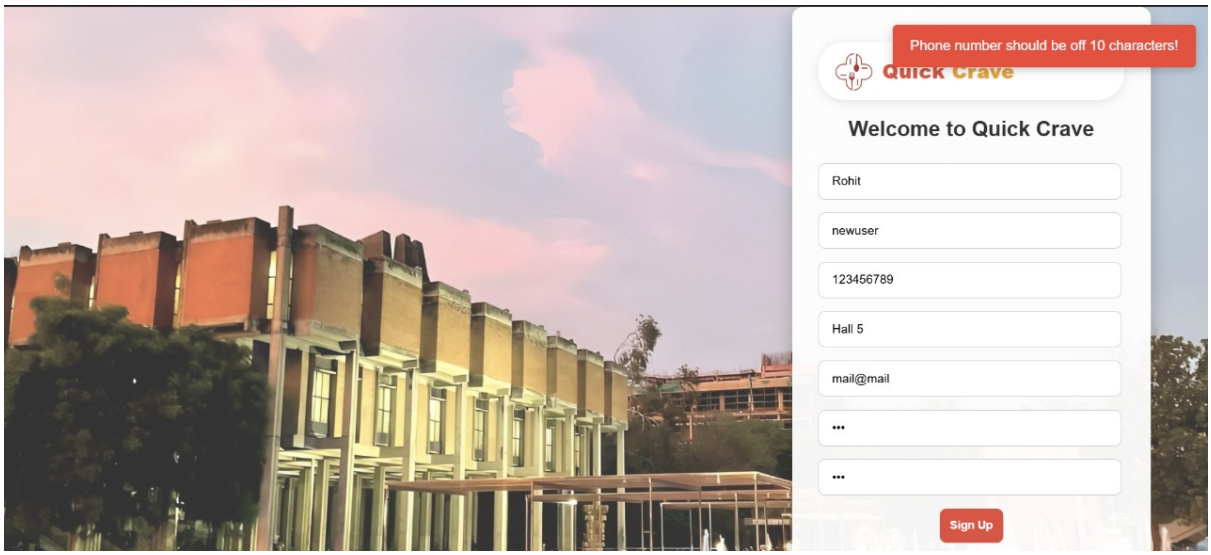

Figure 57: Customer Register wrong phone number

- **Test Case 4:** The system checks that the password set by the user follows standard protocols for a strong password
  Result: If the password does not meet the required strength (e.g., at least 8 characters, including uppercase, lowercase, digit, and special character), the system displays a list of unmet criteria. The user is only allowed to proceed once all password requirements are fulfilled.

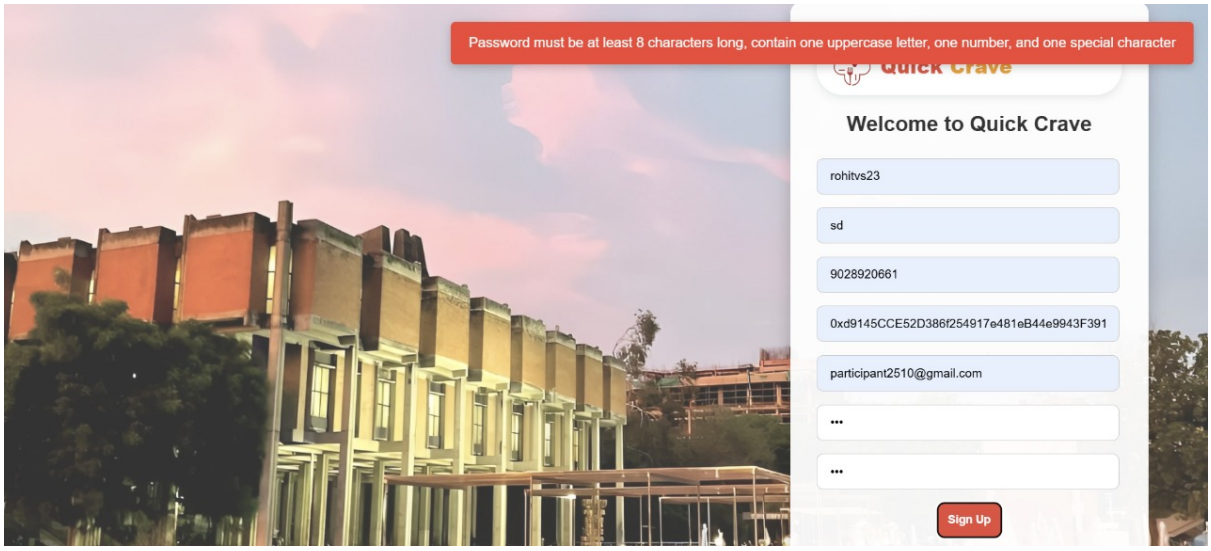

Figure 58: Customer Register Strong Password

3.1.2 Customer

- **Test Case 1:** The system checks that the OTP entered by the user is valid. System checks if the OTP is of 6 digits
  Result: If the OTP does not meet the required format (e.g incorrect digit count), the error message *"Please enter a valid 6-digit OTP."* is shown in the top right.

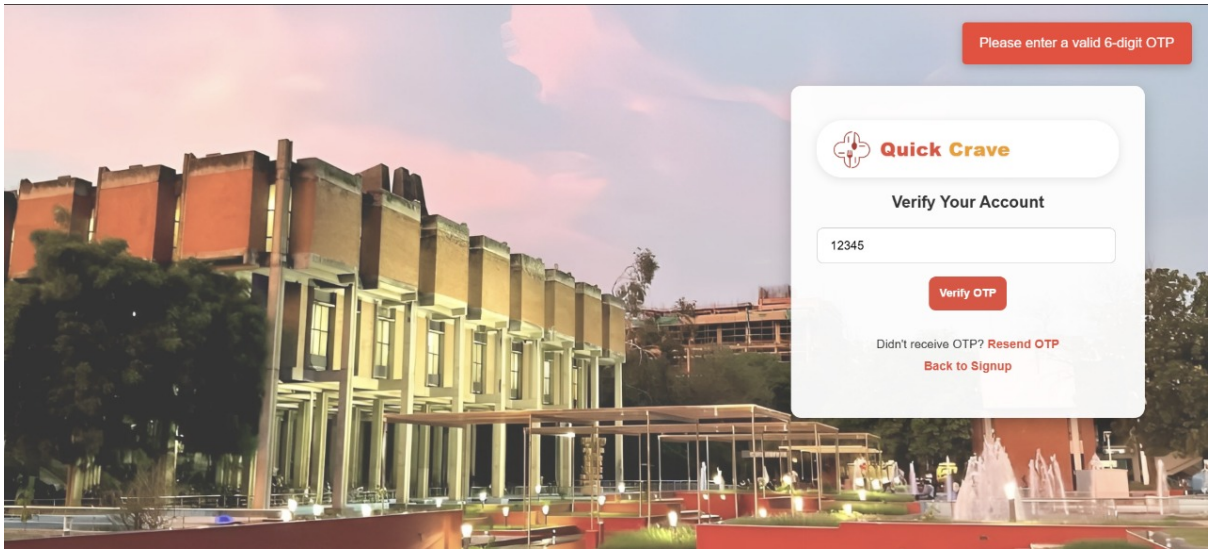

Figure 59: Customer register enter a valid six digit OTP

- **Test case 2:** The system checks that the 6-digit OTP entered by the user is valid.
  Result: If he enters the wrong 6-digit OTP in the initial attempt, the error message *"Incorrect OTP. Attempts left: 2"* is shown in the top right.

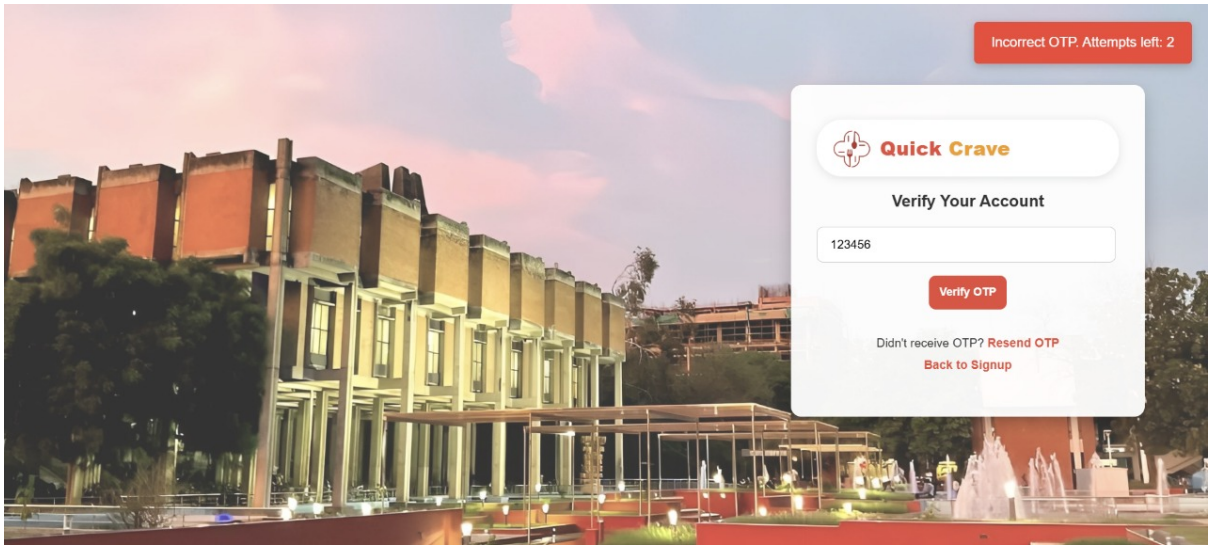

Figure 60: Customer register incorrect OTP

- **Test case 3:** The system checks that the sign up is successful.
  Result: If all the details entered are valid then the system redirects the user to the login page and, the message *"Account Created Successfully"* is shown in the top right.

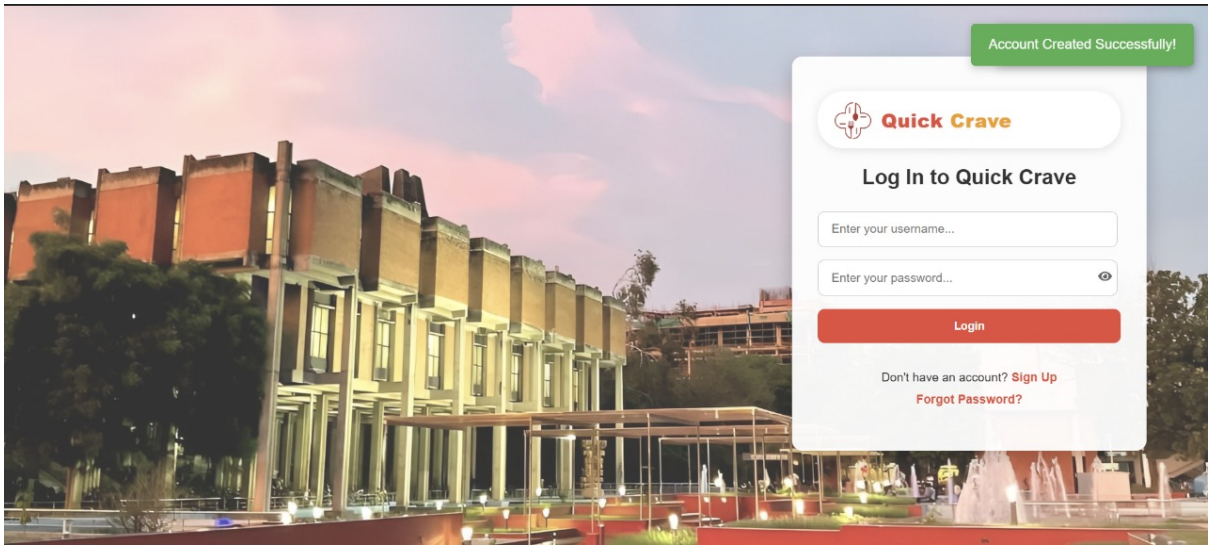

Figure 61: Customer registered successfully

3.1.3 Registering a canteen

Module Details: This module integrates the registration process for new canteens, providing necessary functionalities
Test Owner: Kshitij Gupta **Test Date:** 31/03/2025 **Test Results:** The integrated module is working as expected

- **Test Case :** Documents are added either by dragging them into the designated upload area or by searching and selecting from the local system
- **Test Case :** The system successfully sends the document through the app's mail service

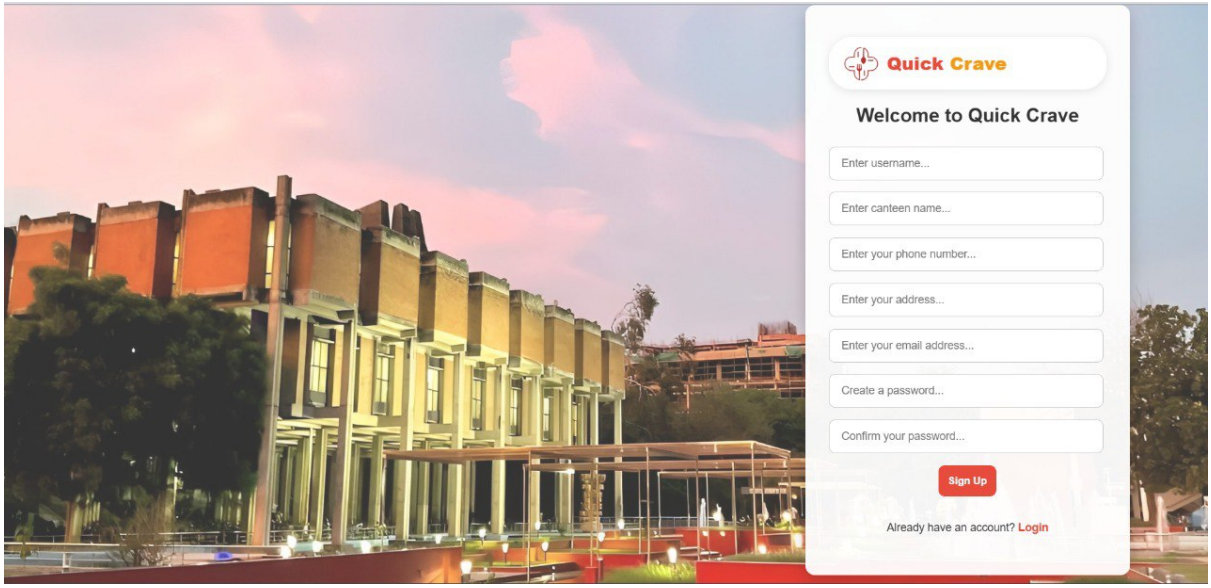

Figure 62: Customer sign up

3.1.4 Logging in as customer

Module Details: This module is integration of all components of Login Page
Test Owner: Dnyanesh Hemendra Shinde
Test Date: 31/03/2025
Test Results: The integrated module is working as expected

- **Test Case 1:** The system checks that the username and password entered by the customer is valid.
  Result: If the customer enters invalid username or password, the error message *Wrong username or password* is shown in the top right.

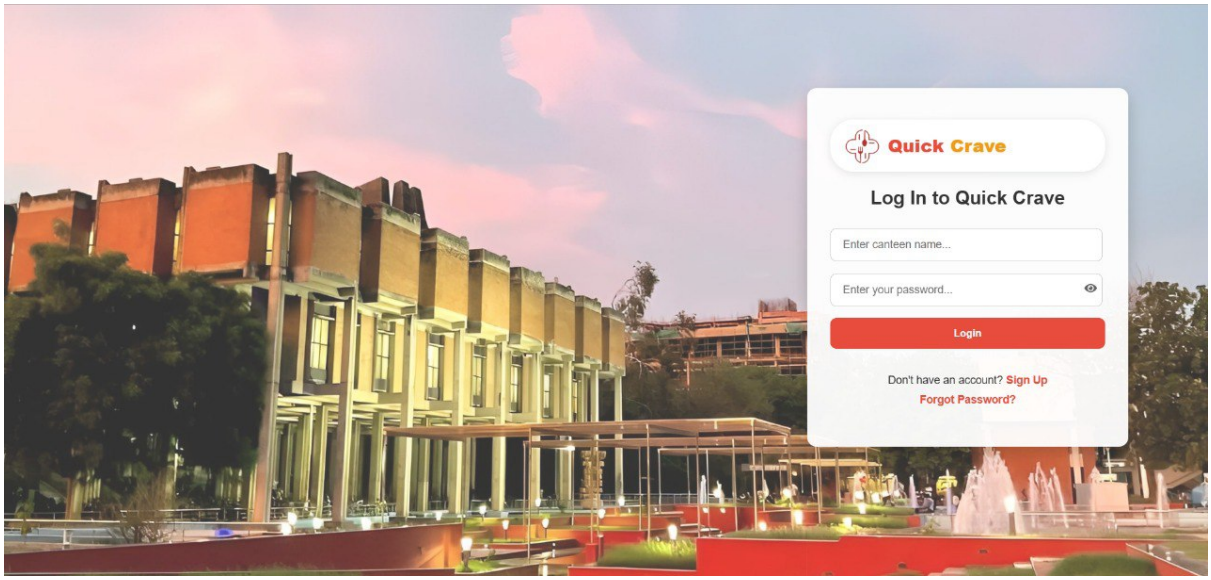

Figure 63: Customer Login Page

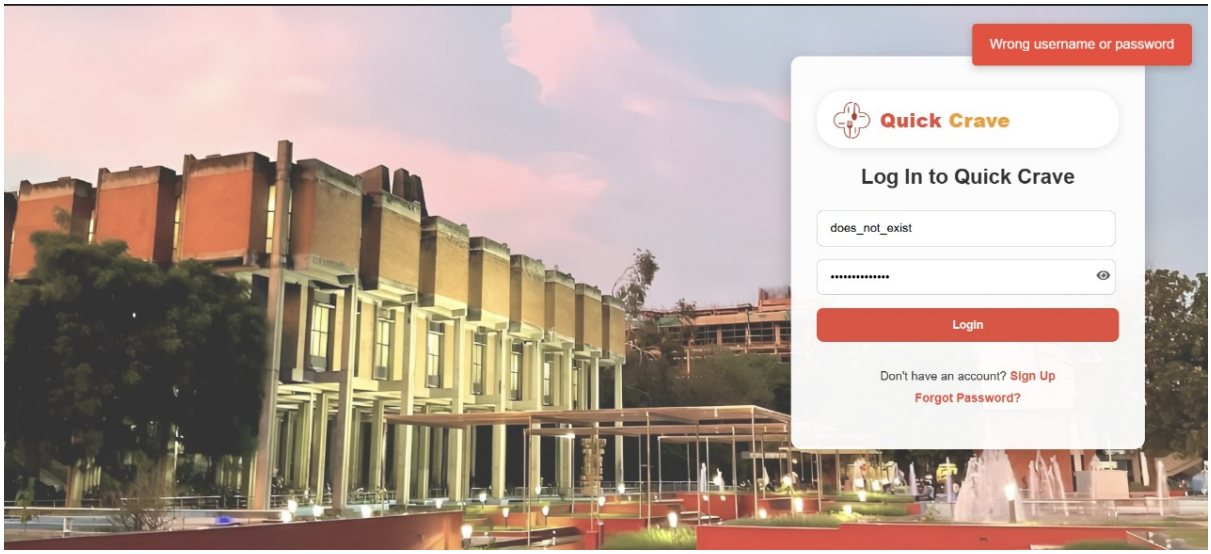

Figure 64: Customer Login wrong credentials

- **Test Case 2:** The system checks that the username and password entered by the customer is valid.
  Result If the customer enters valid username and password then the system redirects the customer to the customer home page and the message *"Login successful as customer"* is shown in the top right.

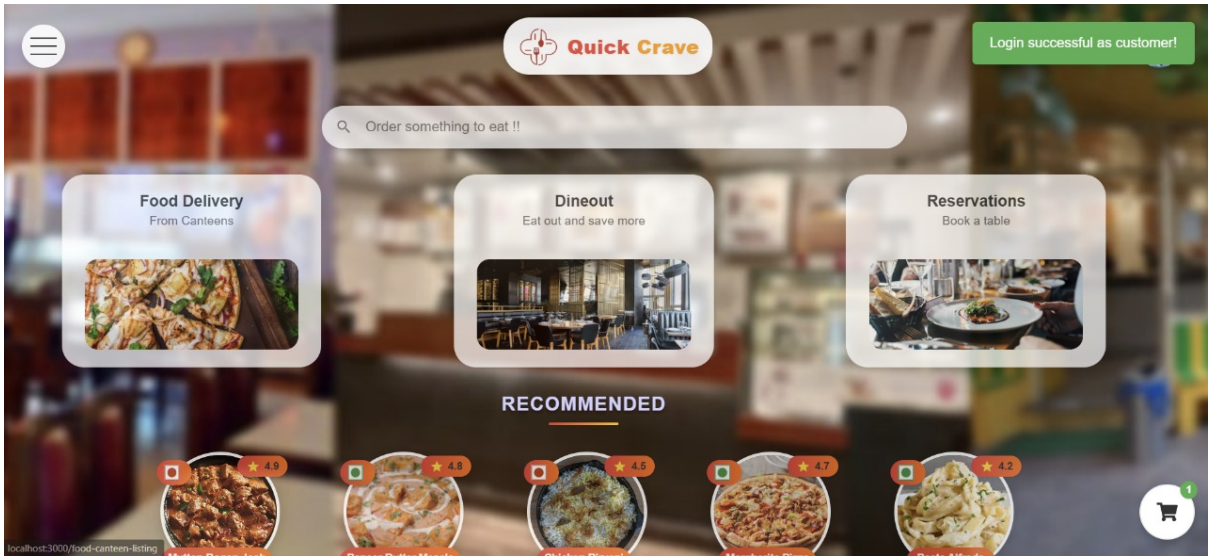

Figure 65: Customer Login Successful

3.1.5 Logging in as canteen manager

Module Details:
Test Owner: Yashaswi Raj **Test Date:** 01/04/2025 **Test Results:** The integrated module is working as expected

- **Test Case 1:** If incorrect credentials are entered
  Result: Error prompt appears
- **Test Case 2:** Correct credentials are entered
  Result: Successful Login

3.2 Customer Side

3.2.1 Customer Home

Module Details: This module integrates the Customer Home Page, providing necessary functionalities.
Test Owner: Nikhilesh Joshi
Test Date: 31/03/2025
Test Results: The integrated module loads all the components that are being called upon being directed to customer home page.

- **Test Case 1:** Upon loading the customer home page, the top 5 recommended dishes are being loaded and displayed on the home page of the customer.

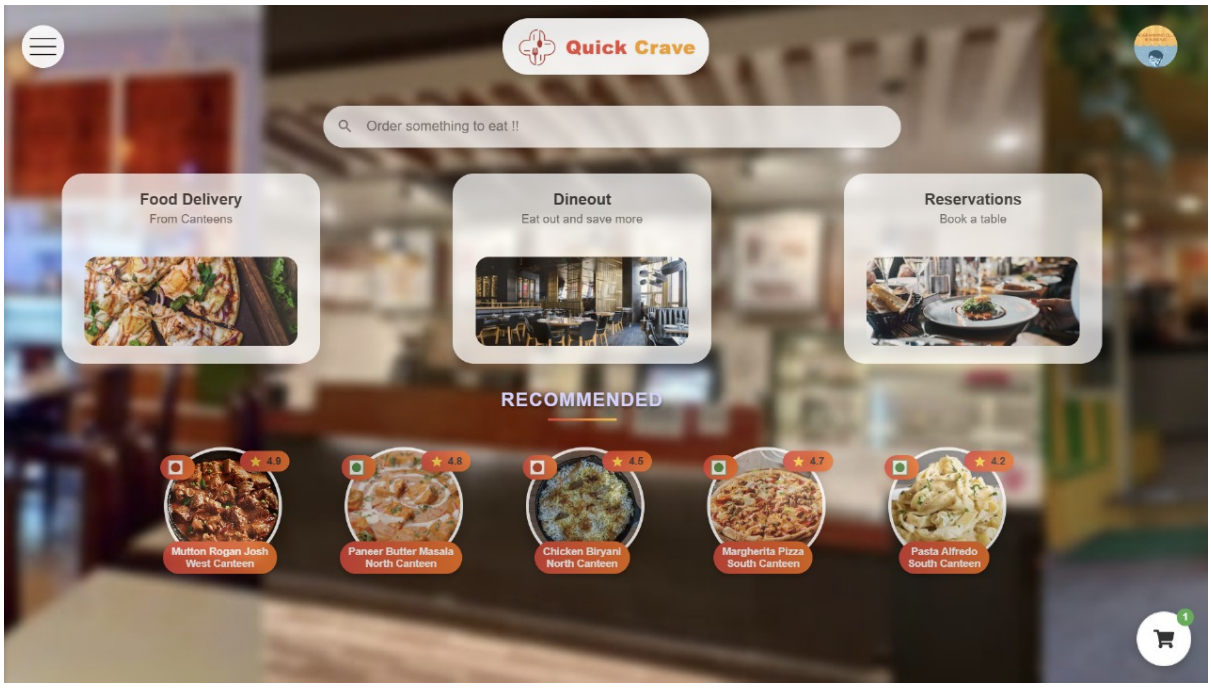

Figure 66: Customer Home Page

- **Test Case 2:** Upon loading the customer home page, the cart of the customer is also being loaded and displayed. The data is being taken from the database and then begin stored in the cart.
- **Test Case 3:** Upon loading the customer home page, the profile of the customer is also being loaded and displayed. The data is being taken from the database and begin stored in session storage and then called when required. Any changes being made to the profile is also being updated in the session storage so that data is consistent throughout.

3.2.2 Customer Profile

Module Details: This module integrates the Customer Profile Page, providing necessary functionalities.
Test Owner: Nikhilesh Joshi
Test Date: 01/04/2025
Test Results: The integrated module loads all the components that are being called upon being redirected to view profile page.

- **Test Case 1:** The details of the customer are correctly loaded along with the correct wallet balance as well the order details of the customer. Profile photo is also being loaded from the link provided through Cloudinary.

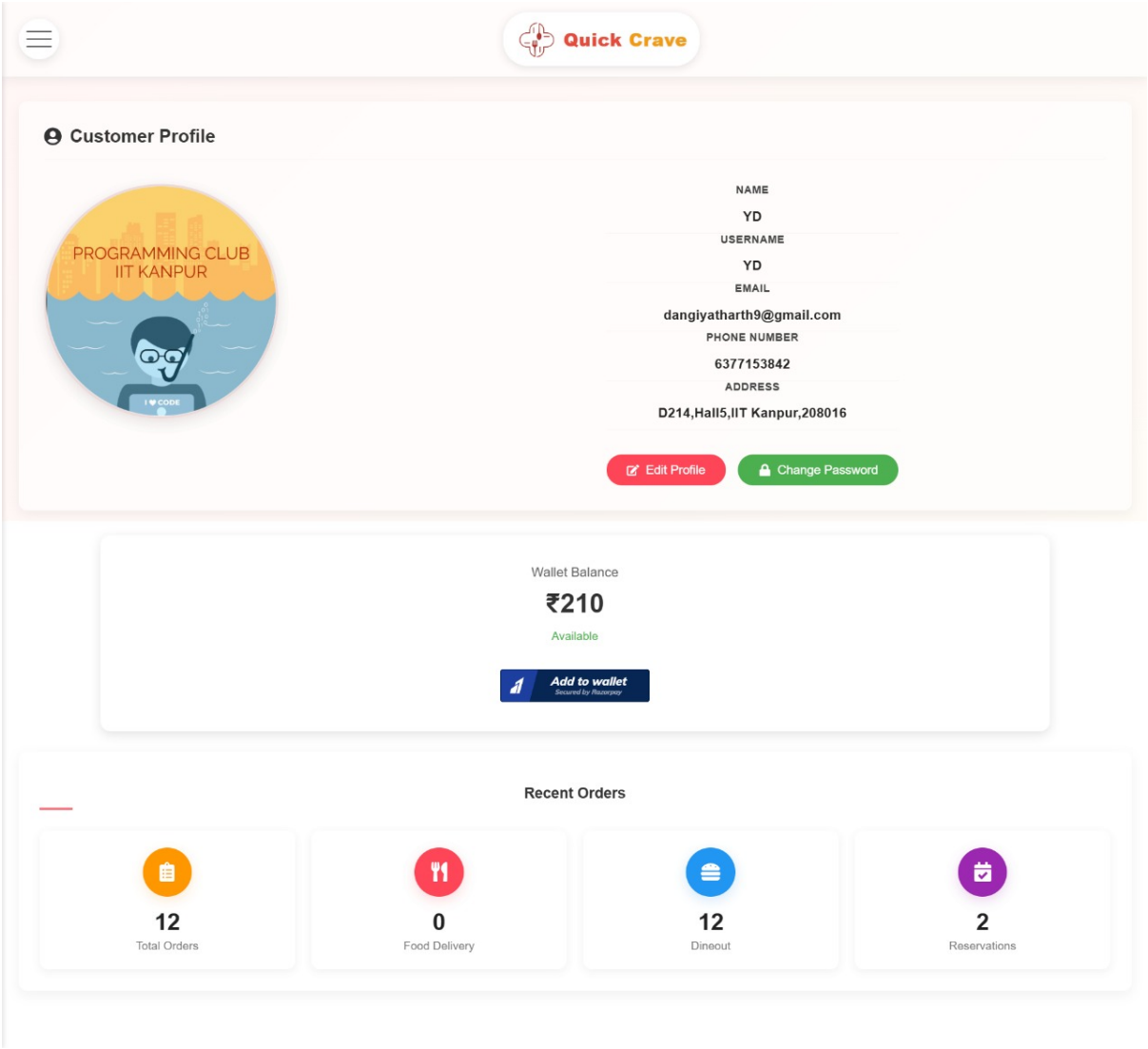

Figure 67: Customer Profile Page

- **Test Case 2:** The system checks that the customer is able to edit the profile.
  Result: When the customer presses the edit profile button, the system navigates him to the edit profile page.

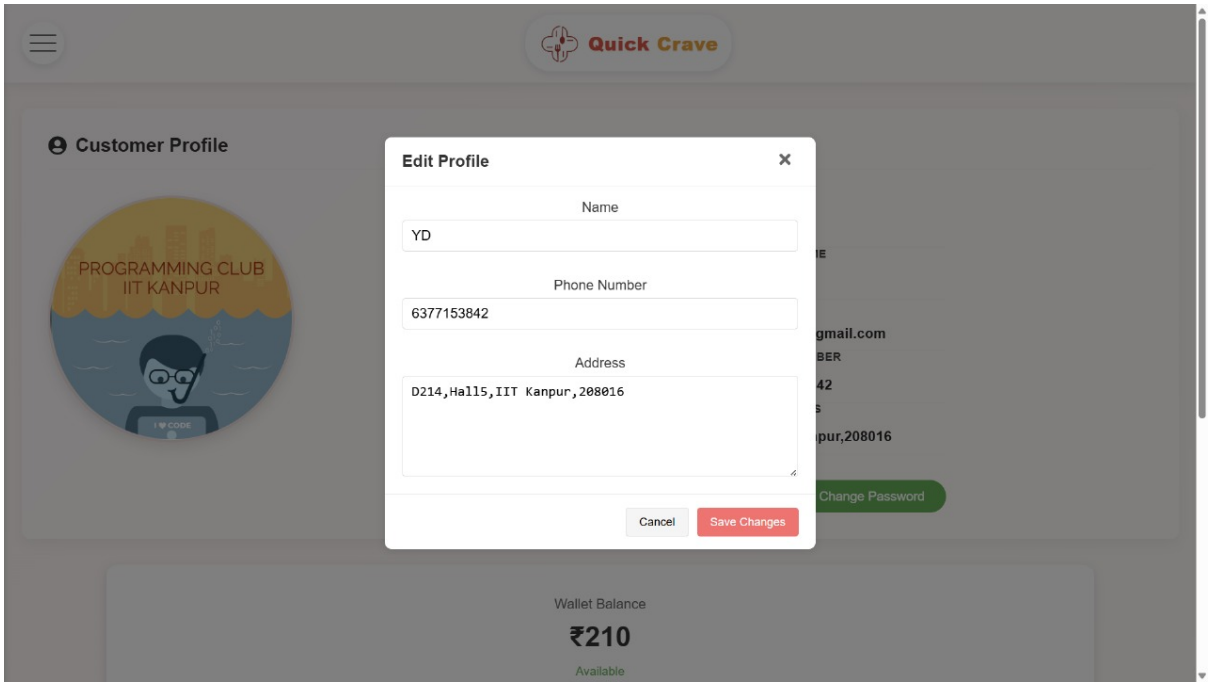

Figure 68: Customer Edit Profile

- **Test Case 3:**The system checks that the profile is updated successfully.
  Result: When the profile is updated successfully the system navigates the customer back to customer profile page and the message *"Profile updated successfully"* is shown in the top right.

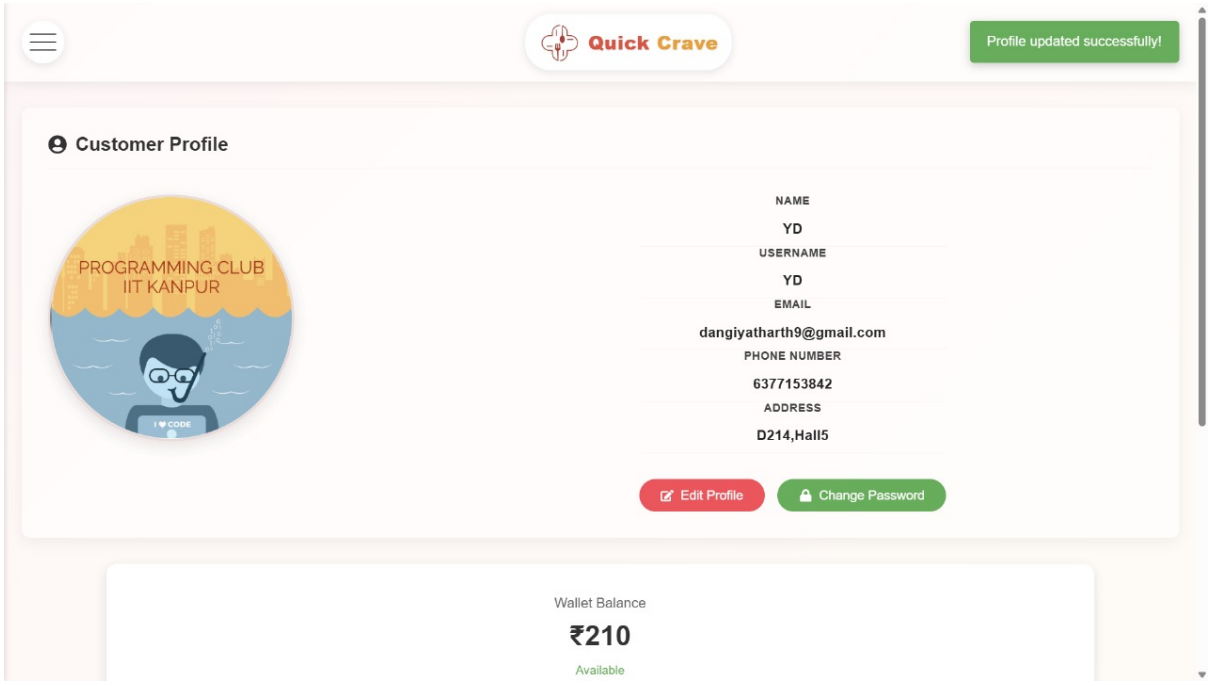

Figure 69: Customer Profile Updated Successfully

Test Case 4: The system checks that the password is changed successfully.
Result: When the customer enters the new password and confirms the new password and presses Update Password then the message *"Password changed successfully! Logging you*

out...” is shown in the top right.

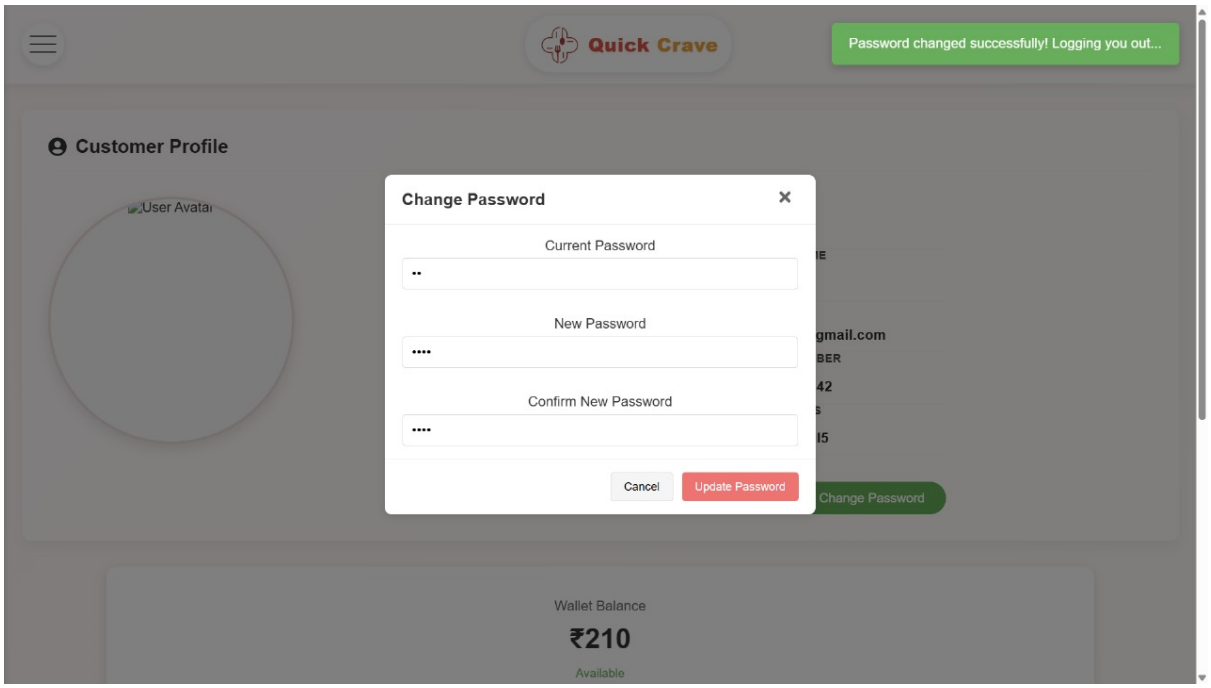

Figure 70: Customer Password Changed Successfully

Test Case 5: The system checks that the customer is able to add money to the wallet.
Result: When the customer presses the add to wallet button the system navigates the customer to ”Add to wallet” page, and the customer enters the amount here.

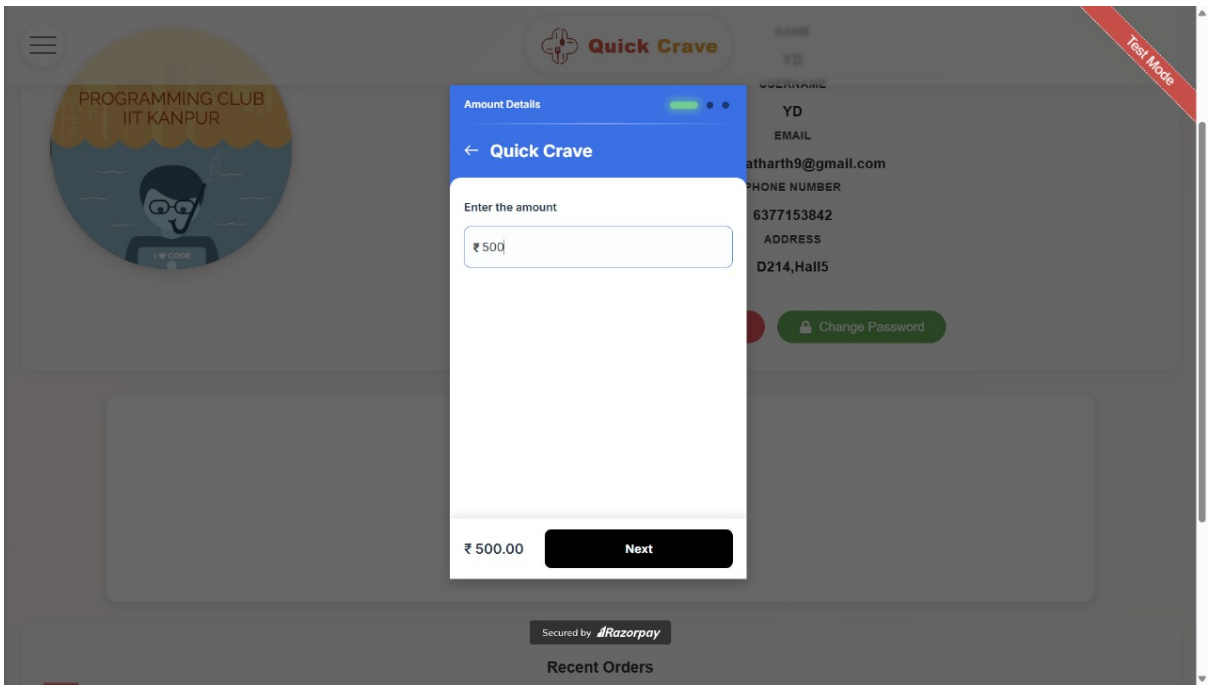

Figure 71: Customer Add to Wallet page

Test Case 6: The system checks that the money added to the wallet is updated in the customer’s profile.

Result: When the customer enters the amount and presses the ”Next” button, the system navigates the customer back to the profile page, and the wallet balance is updated.

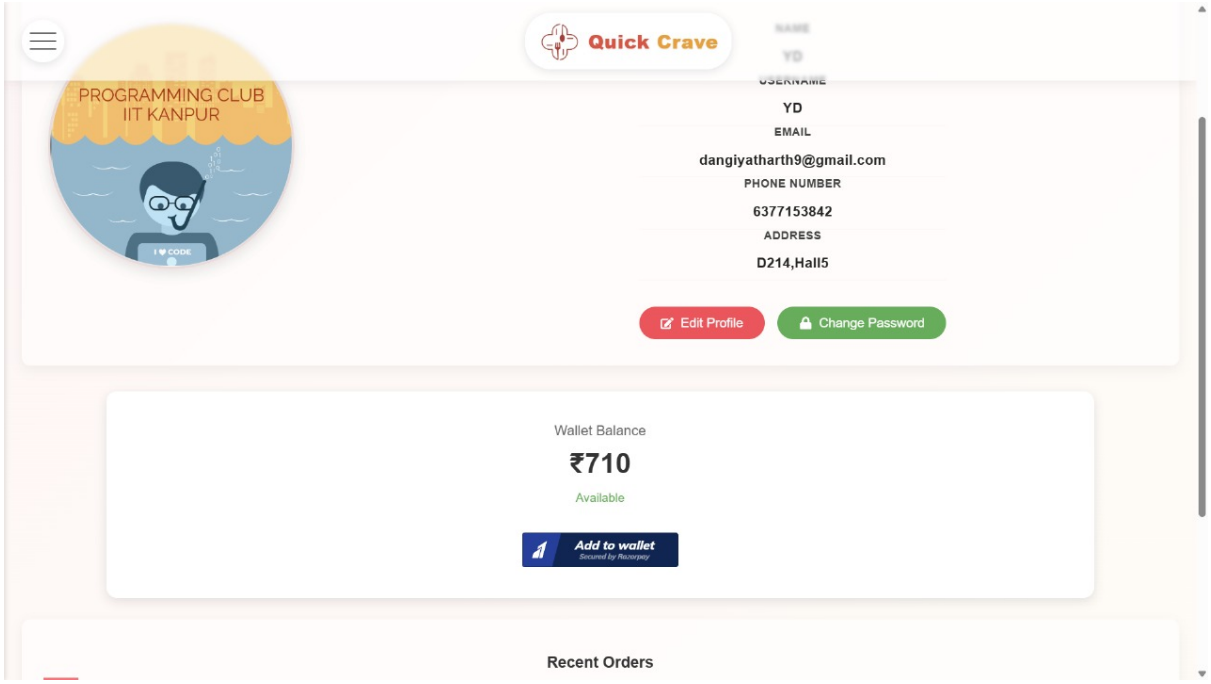

Figure 72: Customer wallet updated

Test Case 7: The system checks that the customer is able to update profile photo.
Result: When the customer presses over the profile photo he is directed to ”File Explorer” where image is selected.

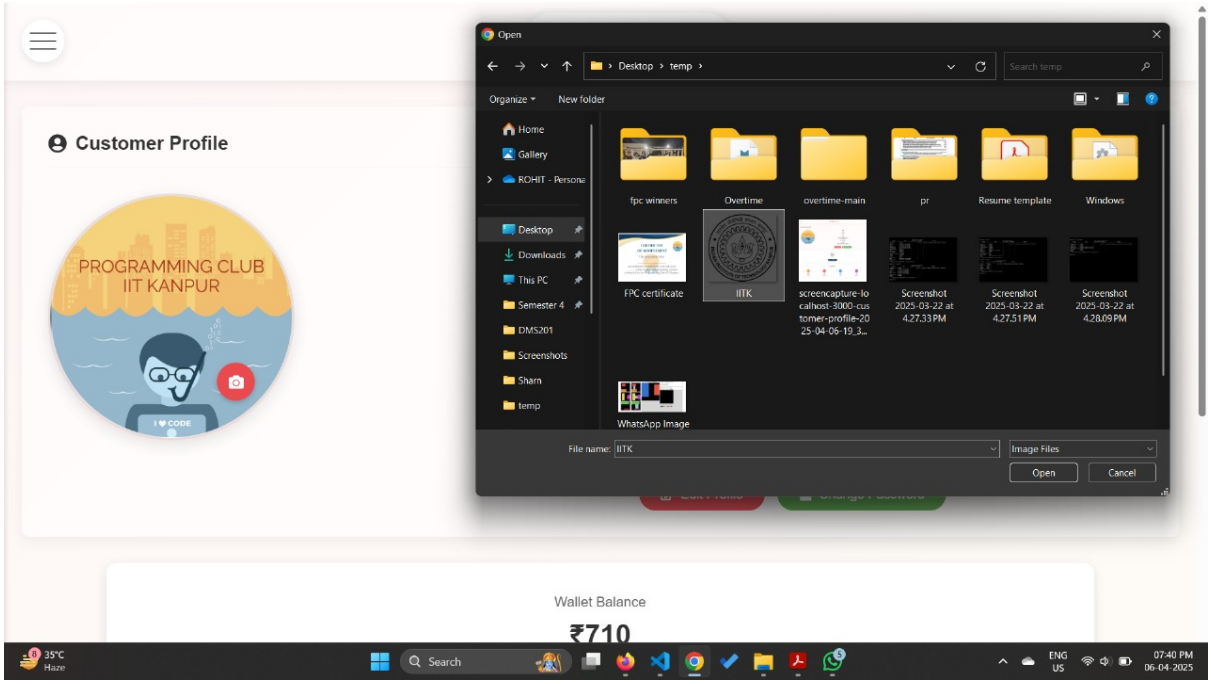

Figure 73: Customer update profile photo

Test case 8: The system checks that the profile photo is updated.
Result: When the customer selects the image from ”File Explorer,” he is navigated back to the customer profile page where the profile photo is updated.

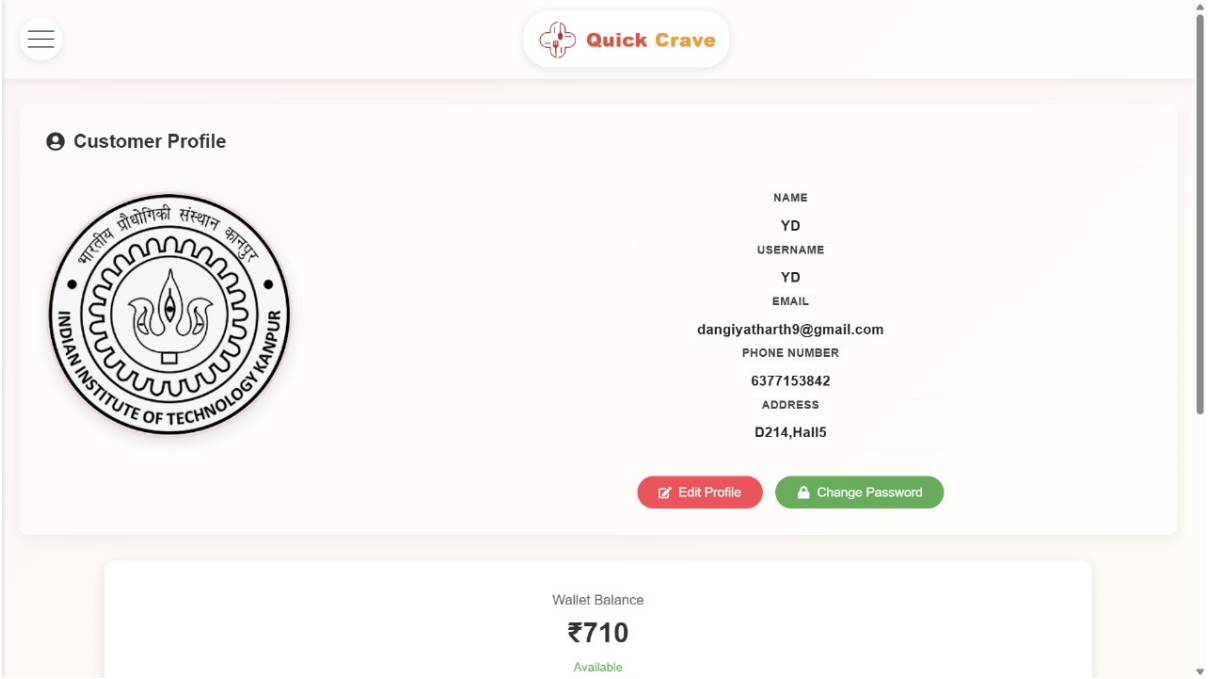

Figure 74: Customer profile photo updated

3.2.3 Cart

Module Details: This module is the integration of the complete Cart Page showing all the food items added from a certain canteen and the final submission of the order
Test Owner: Arihant Kumar
Test Date: 03-04-2025

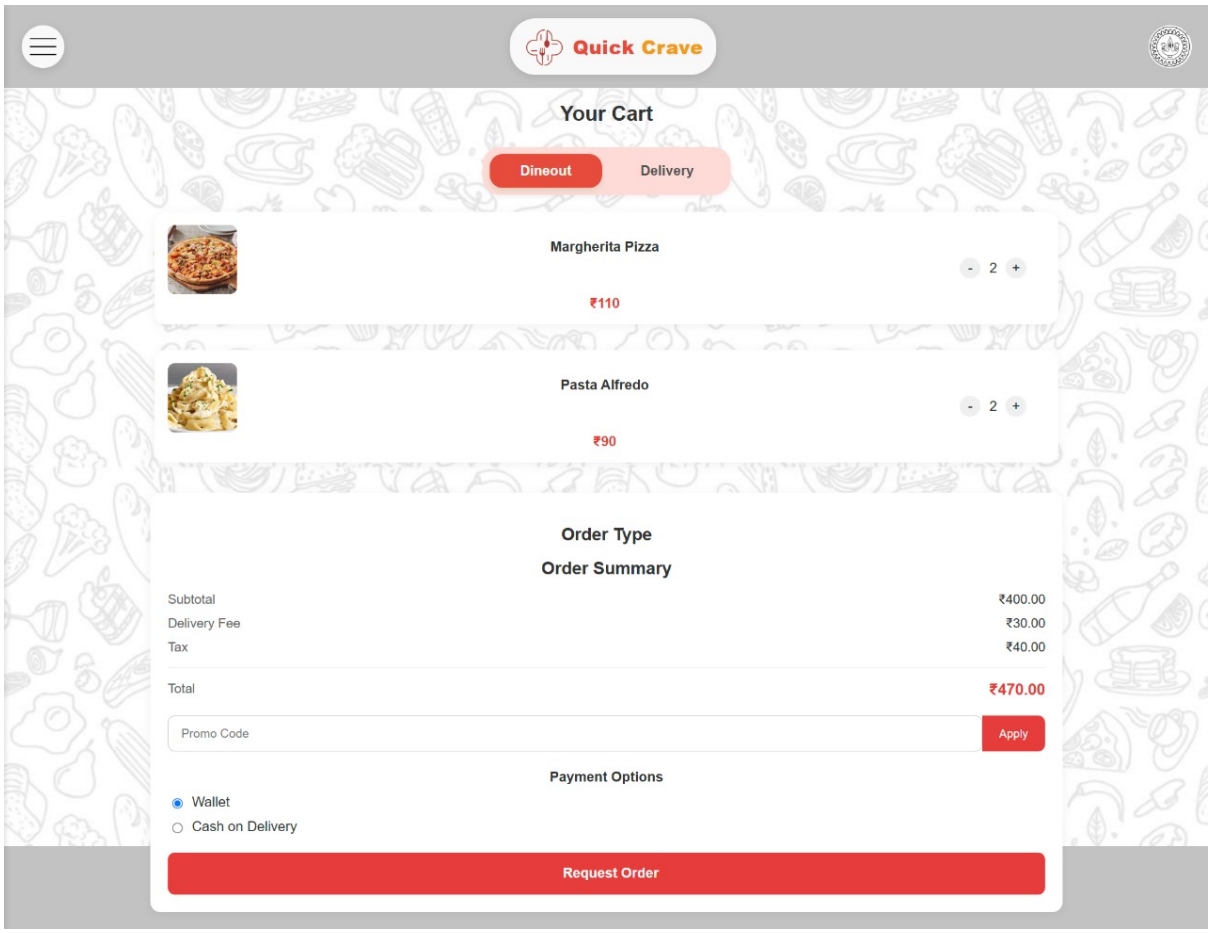

Figure 75: The Cart Page

Test Results: The module correctly loads all the details from the cart of the customer.

- **Test Case** : The module correctly checks if the request of order is being correctly placed. **Result** : Once the items in the Cart are finalized, on clicking the **Request Order** button, the Order is successful sent to the Canteen Manager to be accepted. After requesting the order, the page shows a countdown which automatically closes after 5 minutes. However, if the auto accept of the canteen from which the dish is requested is on, then the order is automatically placed.

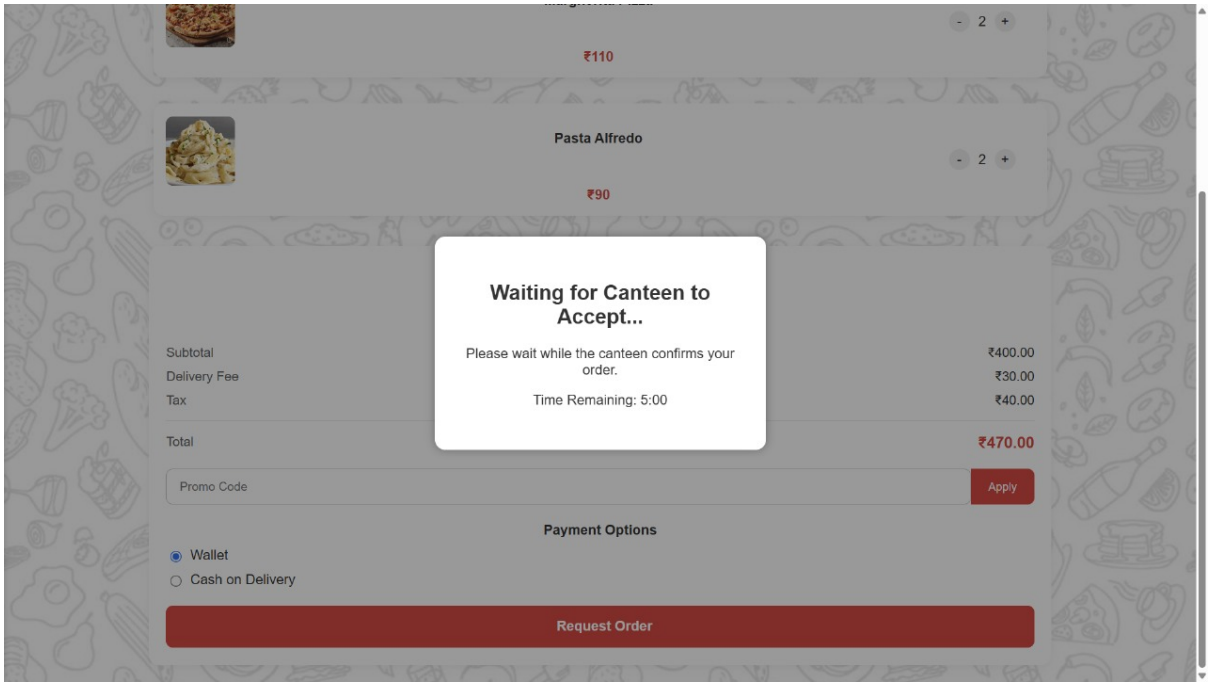

Figure 76: Successful Submission of Order

3.2.4 Sidebar

Module Details: This module is the integration of the complete Sidebar where different functionalities are available
Test Owner: Yatharth Dangi
Test Date: 23-03-2025

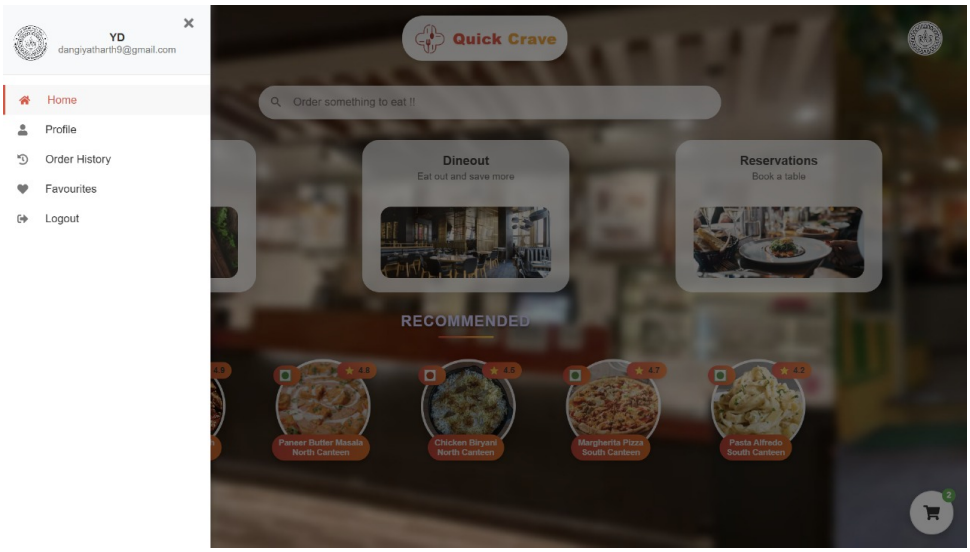

Figure 77: Sidebar

Test Results: The integrated module is working as expected fetching the user details successfully from backend.

- **Test Case** : The system checks the functioning of various options of Sidebar
  Result : On opening the Sidebar, various options are available where the **Home** button takes back to the Home Page as demonstrated above; **Profile** option directs to the Profile as demonstrated above; **Order History** takes to the Order History Page shown below; **Favourites** takes to the Favourites Page shown below; **Log Out** logs out from the current account and takes back to the Login Page

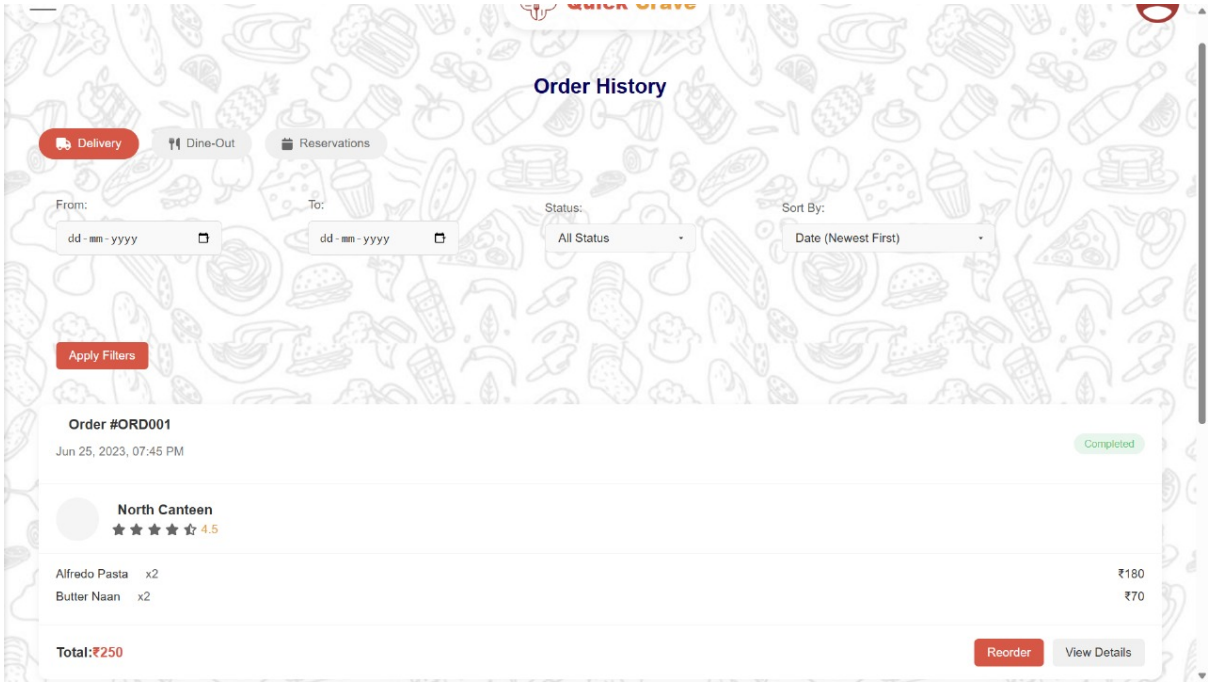

Figure 78: Order History

3.2.5 Reservation

Module Details: This module is the integration of the complete Reservations page to reserve a place in a Canteen for a certain number of people at a certain date & time.
Test Owner: Parnika Mittal
Test Date: 01-04-2025

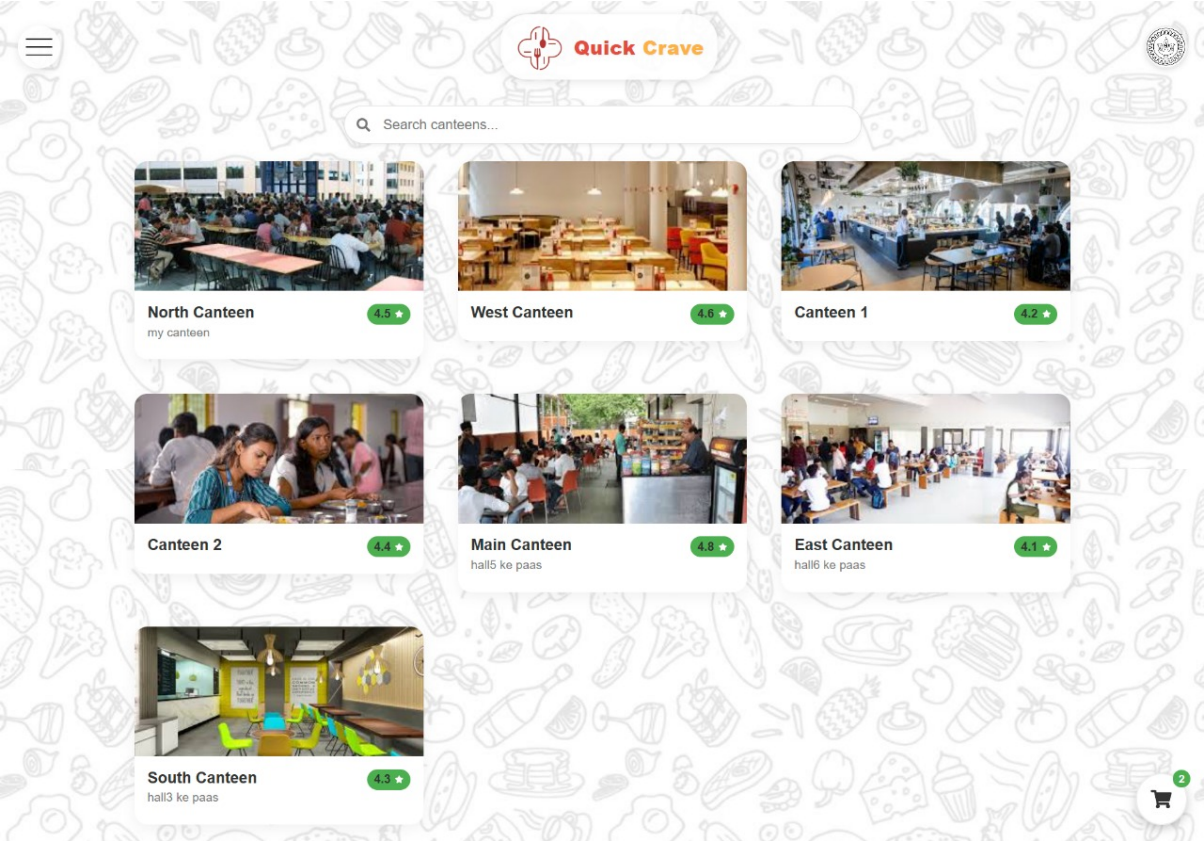

Figure 79: selecting Canteen for Reservation

Test Results: The integrated module is working as expected

- **Test Case 1** : From the available list of canteens for reservation, after selecting the particular canteen it successfully directs to the Reservation Page of that canteen
- **Test Case 2** : After filling in all the details for reservation and clicking the **Reserve Now** button, it successfully submits the Reservation to the Canteen Manager to be verified further.

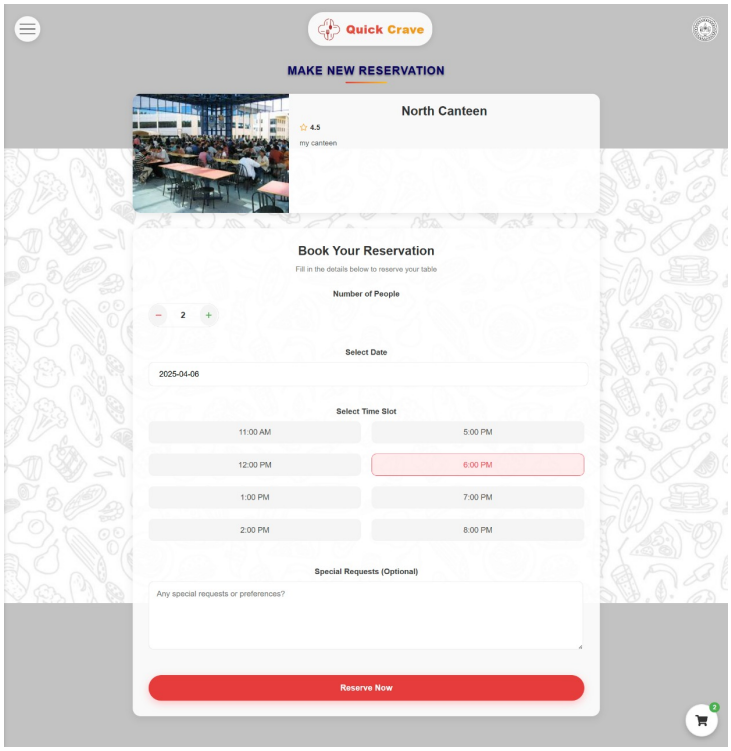

Figure 80: Filling in all the Details for Reservation

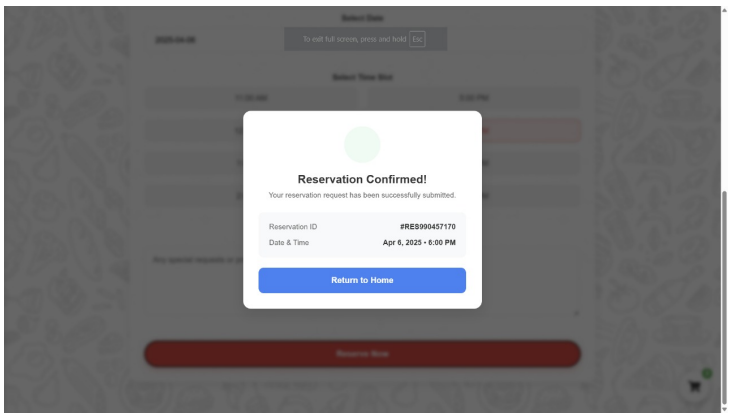

Figure 81: Successful Reservation Made

3.2.6 Food Delivery and Dineout

Module Details: This module is the integration of the complete Food Delivery and Dineout page to display all the food items across different canteens as well as all the canteens available for food delivery and dineout and to add food items to the Cart for such purposes.
Test Owner: Rohit Vijaykumar Somani
Test Date: 01/04/2025

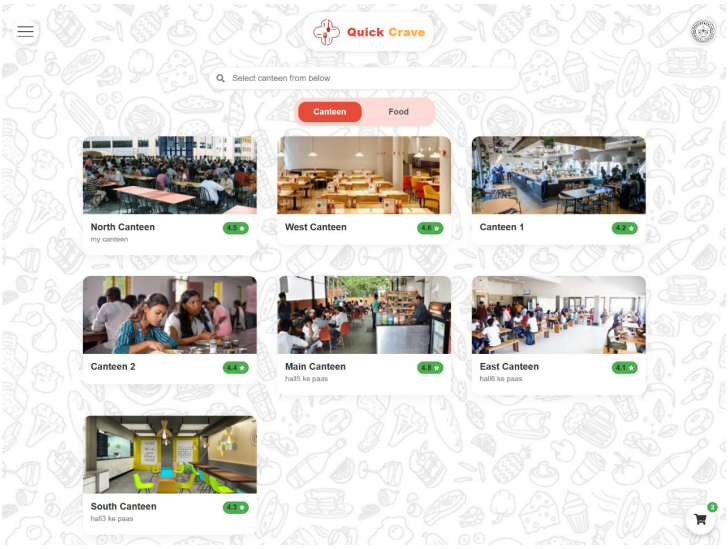

Figure 82: All the different canteen available

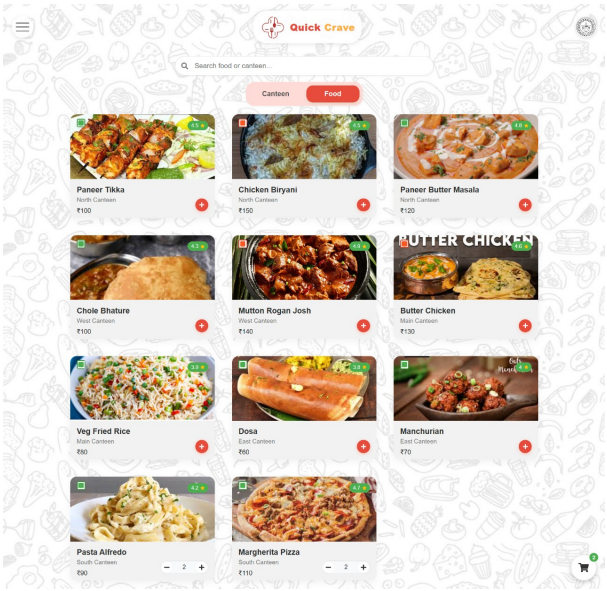

Figure 83: All the different food items available

Test Results: The integrated module is working as expected

- **Added to Empty Cart** : If the Cart is empty, any food item from any canteen is successfully added to the cart.
- **Adding to Non-Empty Cart** : If the Cart is not empty, and the new food item is from the same canteen as that of the items in that Cart, the food item is added successfully. If the new food item added is not from the same canteen, that food item is not added.

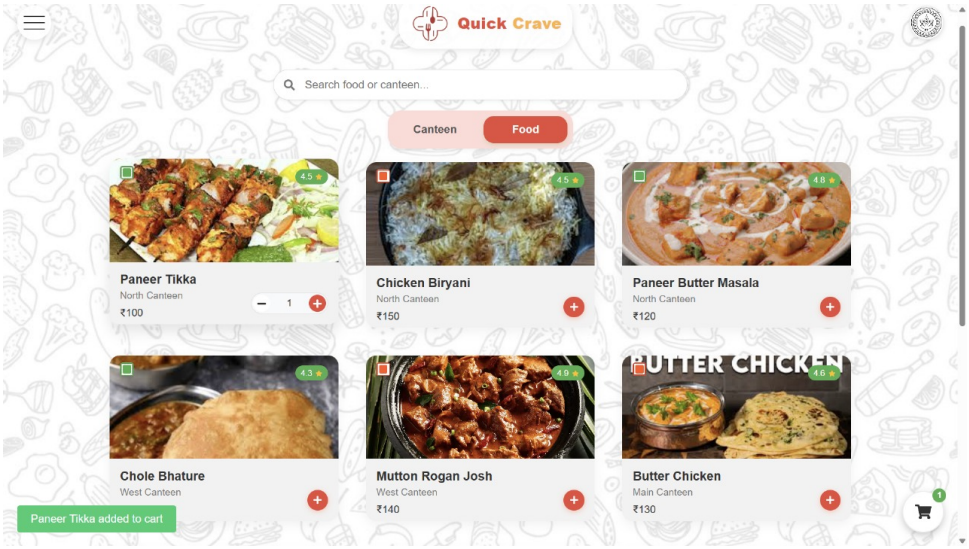

Figure 84: Selected Food Item Added

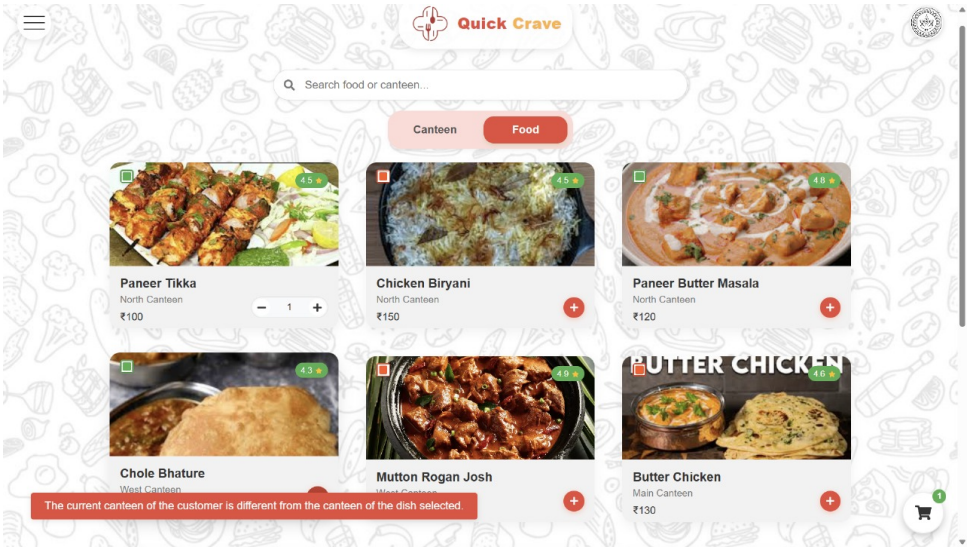

Figure 85: selected Food Item from Different Canteen not added

3.2.7 My Favourites

Module Details: This module is the integration of the complete My Favourites page to display all the favourite food items across different canteens and to display confirmation pop-up when any particular food item is to be removed by deselecting the Red Heart
Test Owner: Dnyanesh Hemendra Shinde
Test Date: 03/04/2025

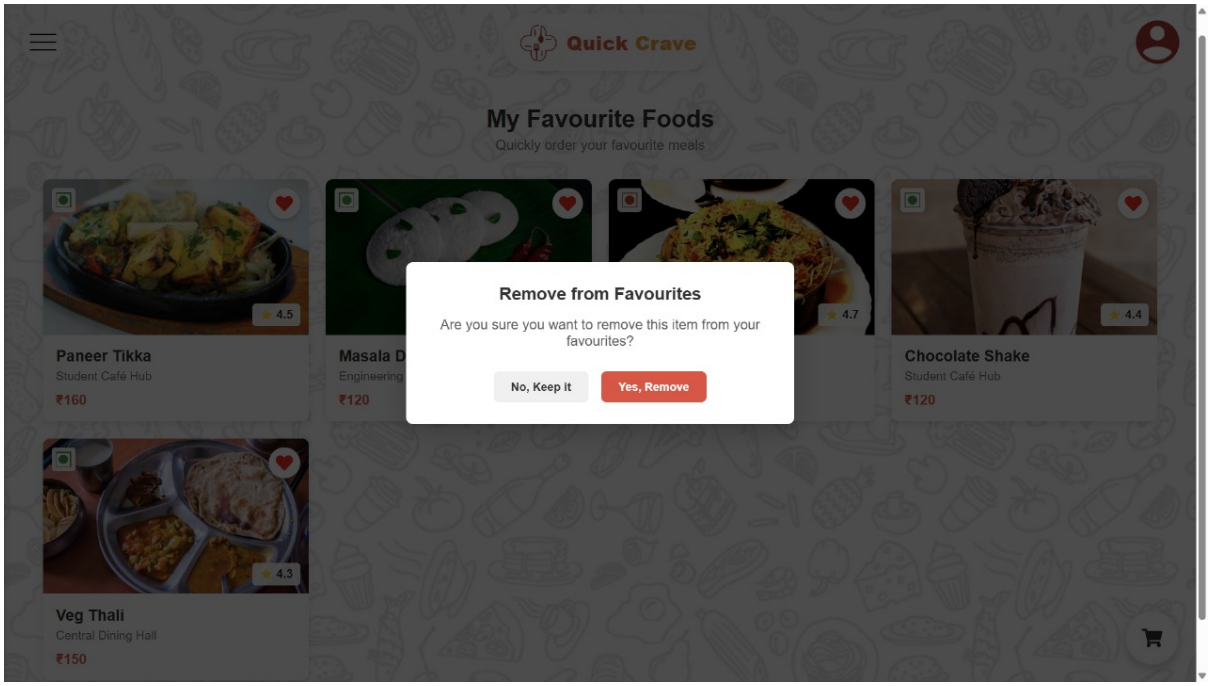

Figure 86: Removing Food Item from Favourites

Test Results: The integrated module is working as expected

- **Test Case 1** : If the **Yes, Remove** option is selected on the pop-up, it removes the item from the Favourites page
- **Test Case 2** : If the **No, Keep It** option is selected on the pop-up, it continues to keep it on the Favourites Page.

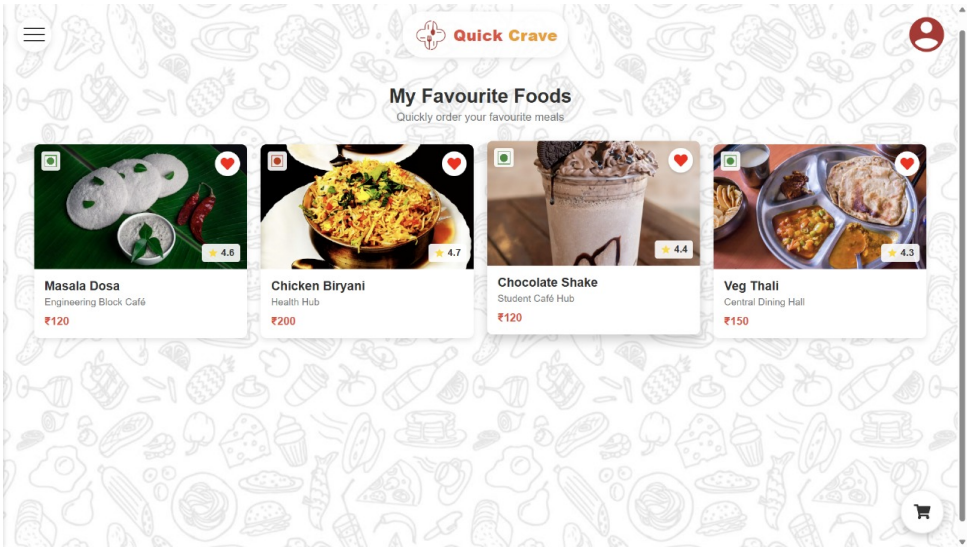

Figure 87: Selected Food Item Removed

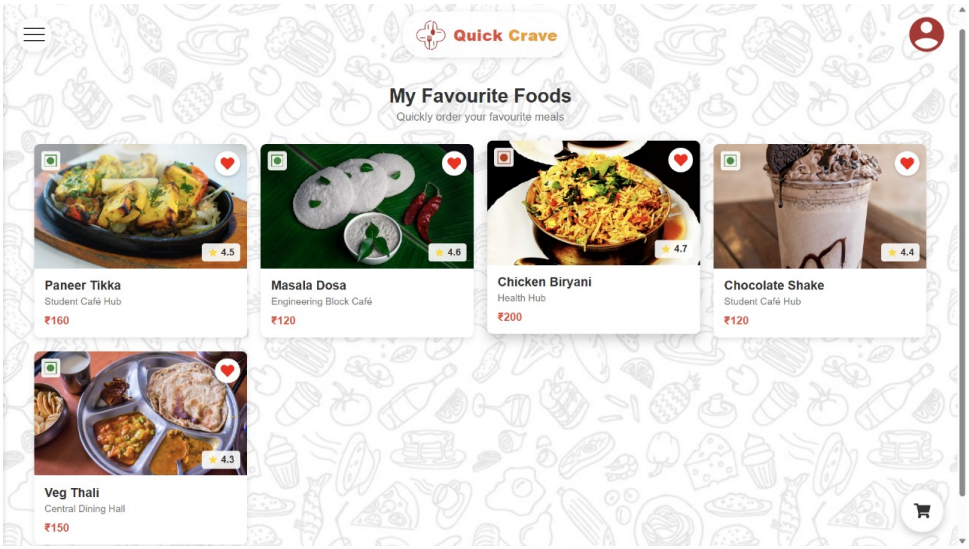

Figure 88: Selected Favourite Food Item Kept

3.3 Canteen Side

3.3.1 Canteen Home Page

Module Details: This module integrates all the relevant components to display the data on order queue, reservation queue , discount & coupons cards and the trending picks of the canteen
Test Owner: Parnika Mittal
Test Date: 03/04/2025
Test Results: The integrated module is working as expected

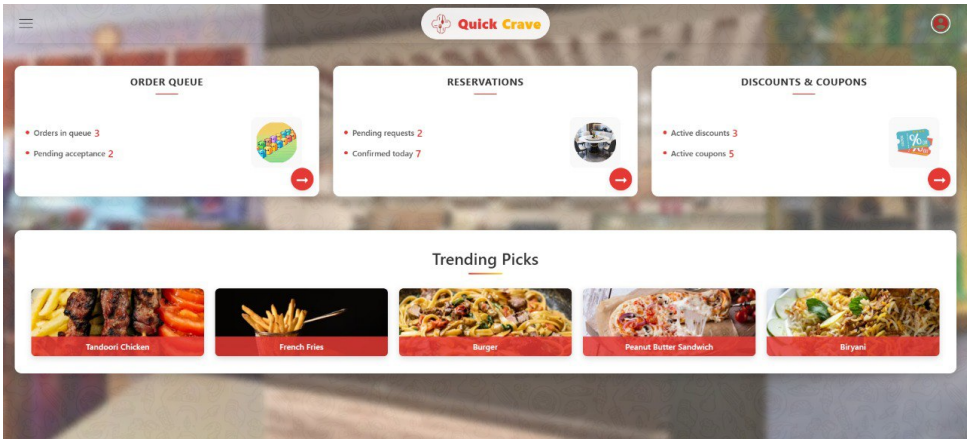

Figure 89: Canteen Home Page

3.3.2 Order Queue Page

Module Details: This module integrates all the relevant components to display and manage the order queue along with the current status of orders placed in the last 24 hours
Test Owner: Arihant Kumar
Test Date: 02/04/2025
Test Results: The integrated module is working as expected

- **Test Case 1:** All Orders filter selected
  Result: All orders correctly displayed
- **Test Case 2:** Waiting For Approval filter selected
  Result: Orders waiting for approval correctly displayed
- **Test Case 3:** Waiting for payment filter selected
  Result: Orders currently waiting for payment correctly displayed
- **Test Case 4:** Cooking filter selected
  Result: Orders currently being cooked correctly displayed
- **Test Case 5:** Ready for delivery filter selected
  Result: Orders ready for delivery correctly displayed
- **Test Case 6 :** Rejected/Failed filter selected
  Result: Rejected/Failed Orders correctly displayed

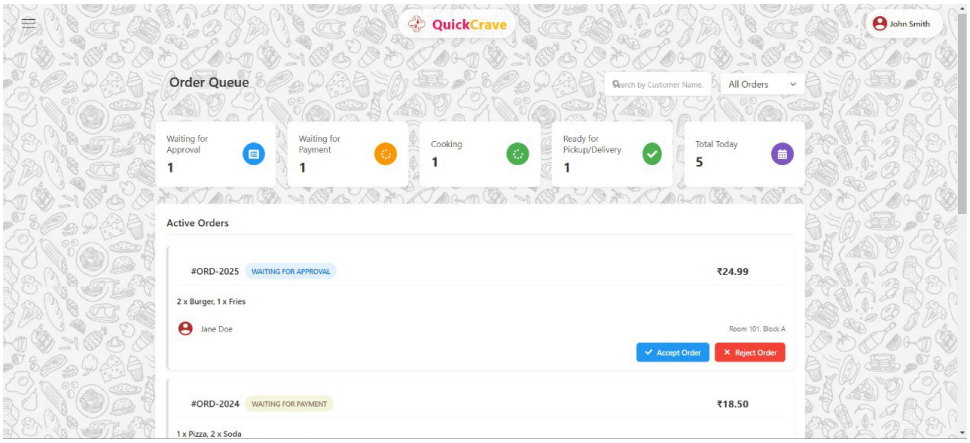

Figure 90: Order Queue

3.3.3 Reservation Queue Page

Module Details: This module integrates all the relevant components to display the reservations in the reservation queue and implemented filters
Test Owner: Suvid Goel
Test Date: 01/04/2025
Test Results: The integrated module is working as expected

- **Test Case 1:** All Reservations filter selected
  Result: All reservations correctly displayed
- **Test Case 2:** Pending Requests filter selected
  Result: Requests rec correctly displayed
- **Test Case 3:** Accepted filter selected
  Result: Reservations accepted in the past week correctly displayed
- **Test Case 4:** Rejected filter selected
  Result: Reservations rejected in the past week correctly displayed

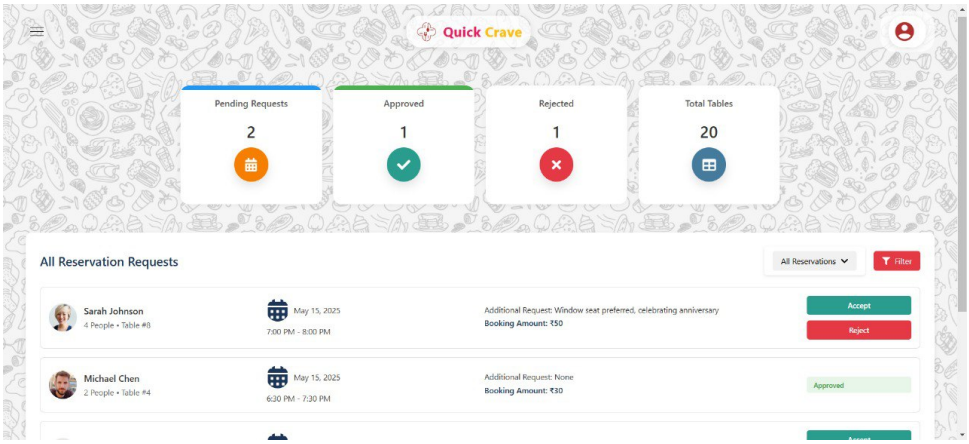

Figure 91: Reservation Queue

3.3.4 Discounts and Coupons Page

Module Details: This module integrates all the relevant components to display existing discounts and coupons and creating new discounts and coupons
Test Owner: Akshay Kamatam Reddy
Test Date: 04/04/2025
Test Results: The integrated module is working as expected

- **Test Case 1:** Discount % is greater than 100
  Result: Error prompt appears that discount % cannot be greater than 100
- **Test Case 2:** Discount % is between 0 and 100
  Result: Discount is applied to dish
- **Test Case 3:** Discount value in coupon is greater than or equal to Minimum Order Value
  Result: Error prompt appears
- **Test Case 4:** Discount value in coupon is less than Minimum Order Value
  Result: Coupon is created

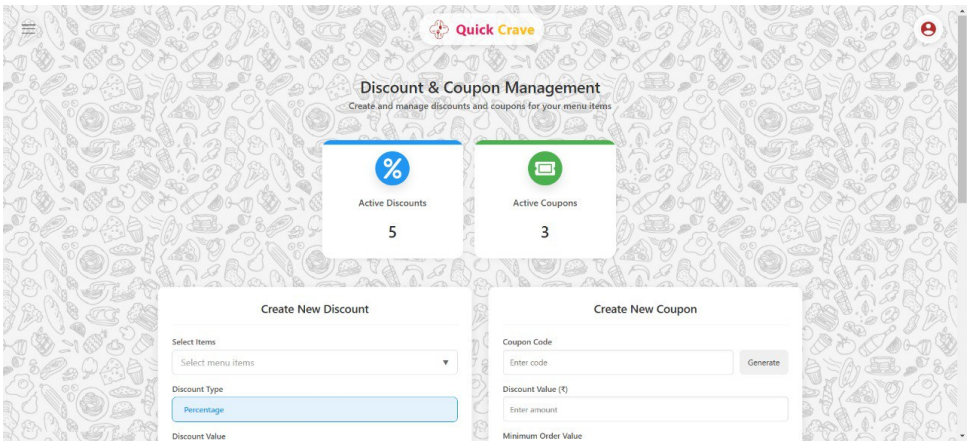

Figure 92: Discounts Page

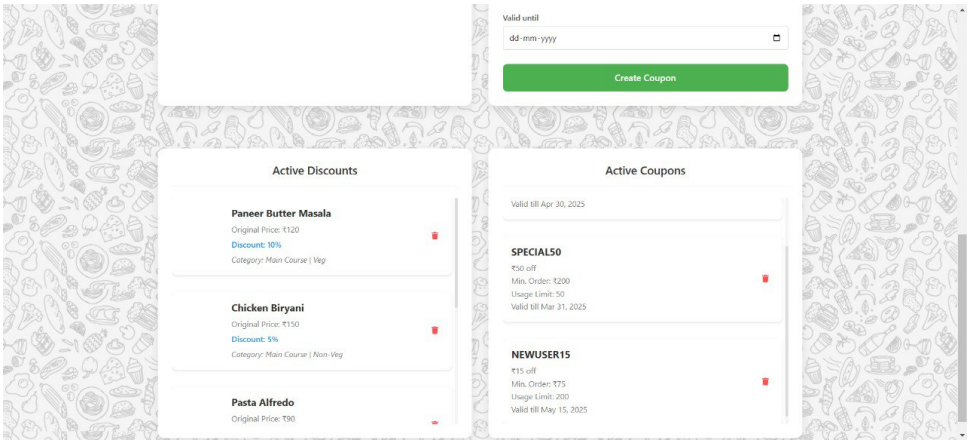

Figure 93: Discounts Page - 2

3.3.5 Menu Management Page

Module Details: This module integrates all the relevant components to display and modify the menu for the canteen
Test Owner: Kshitij Gupta
Test Date: 04/04/2025
Test Results: The integrated module is working as expected

- **Test Case 1:** Adding a new dish item on the Menu
  Result: After filling all the relevant details the dish is successfully added
- **Test Case 2:** Discount % is greater than 100
  Result: Error prompt appears that discount % cannot be greater than 100
- **Test Case 3:** Discount % is between 0 and 100
  Result: Discount is applied to dish

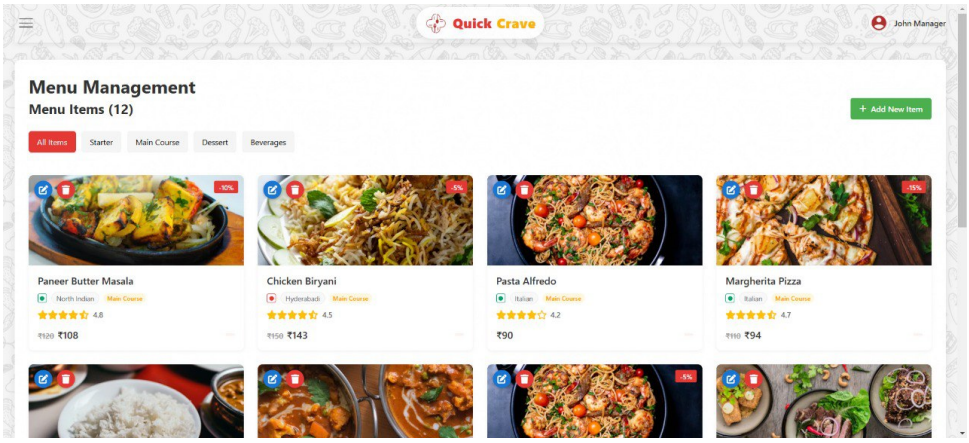

Figure 94: Menu Management Page

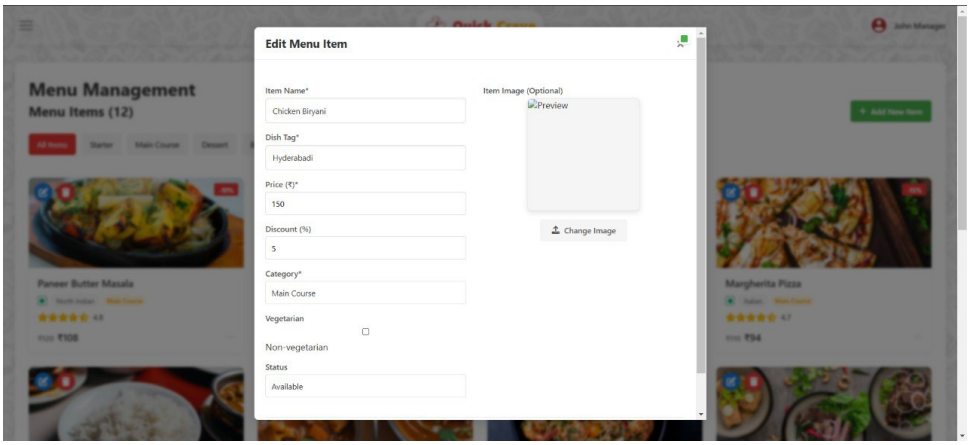

Figure 95: Adding Dish Menu Management Page

3.3.6 Profile Page

Module Details: This module integrates all the relevant components to display the orders in the order queue and implemented filters

Test Owner: Dnyanesh Hemendra Shinde
Test Date: 29/03/2025

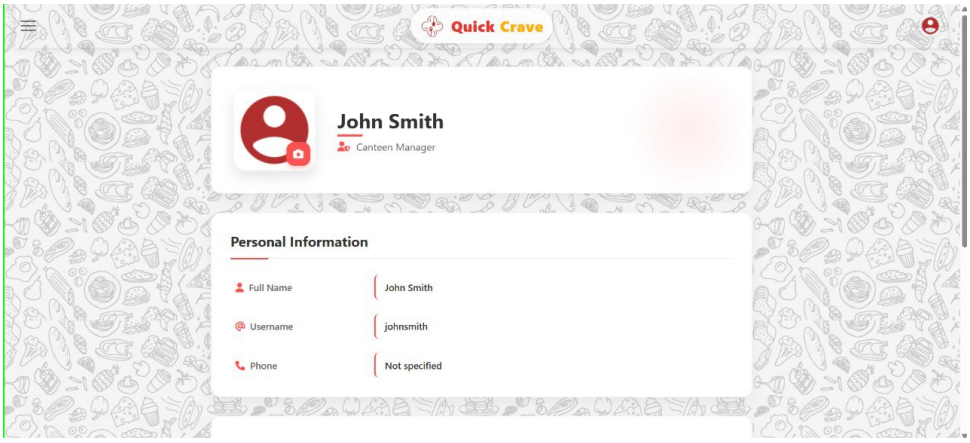

Figure 96: Canteen Manager Profile Page

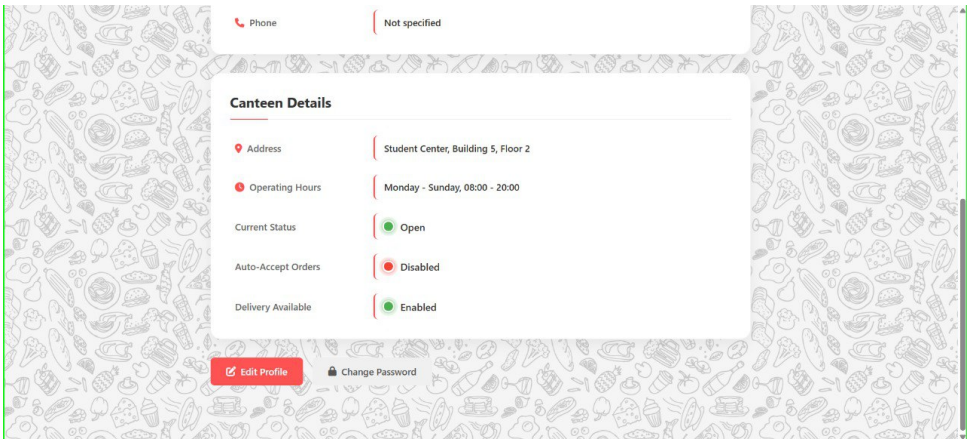

Figure 97: Canteen Manager Profile Page

Test Results: The integrated module is working as expected

- **Test Case 1:** If new username entered in edit profile is already taken
  Result: Error prompt appears that username is already taken
- **Test Case 2:** If password does not follow security standards
  Result: Error prompt appears displaying the security standard not followed

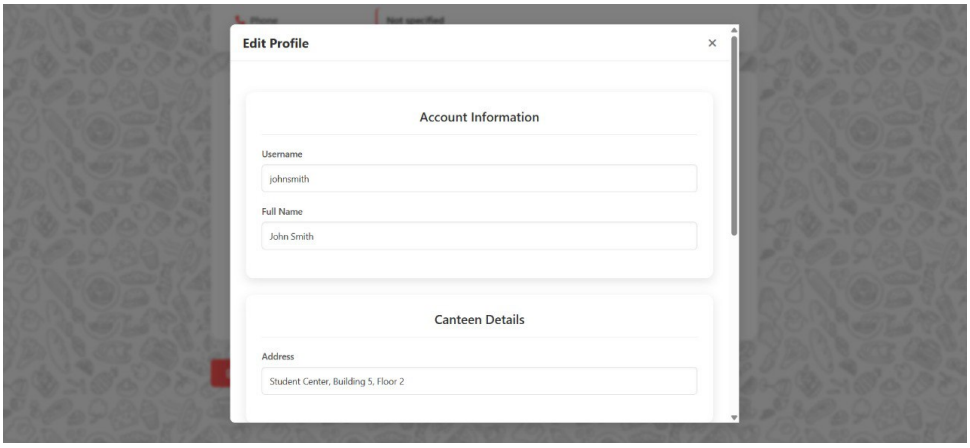

Figure 98: Edit Canteen Profile Page

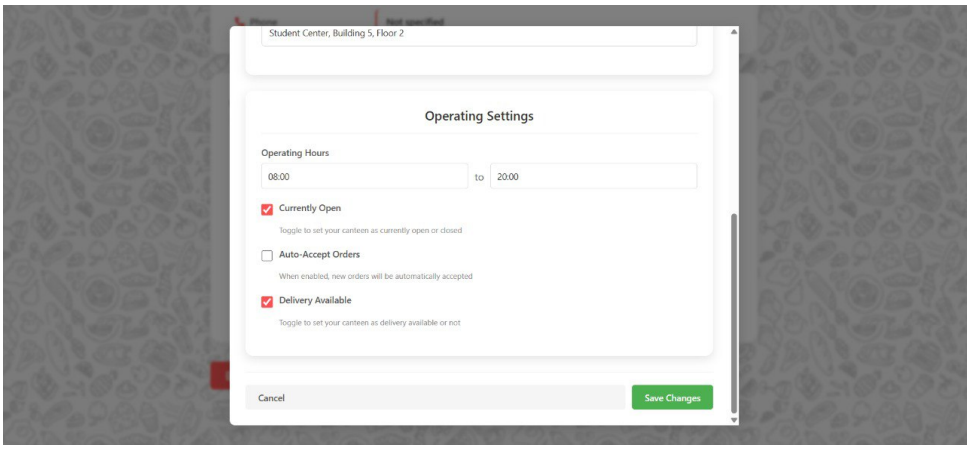

Figure 99: Edit Canteen Profile Page

3.3.7 Statistics Page

Module Details: This module integrates all the relevant components to display the statistics and implemented filters
Test Owner: Yashaswi Raj
Test Date: 03/04/2025
Test Results: The integrated module is working as expected

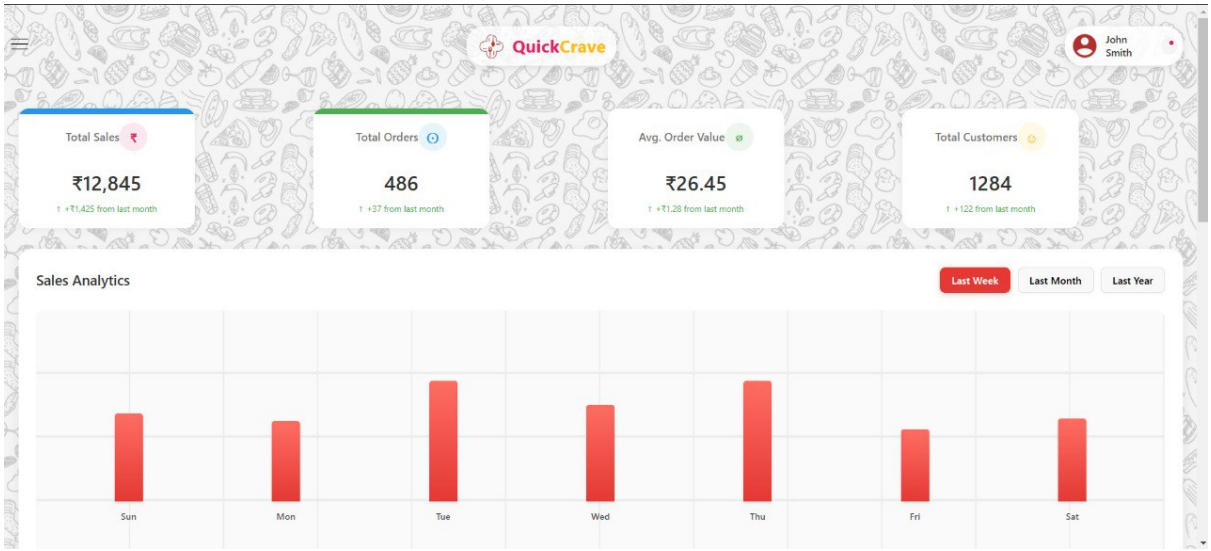

Figure 100: Canteen Manager Statistics Page

Figure 101: Canteen Manager Statistics Page

4 System Testing

4.1 Load testing

4.1.1 customer/customer-search-canteen

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type: application/json" \
  -H "Cookie: token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk1ODYwNywiZXh
     ._It7Pjc9AHKs0IiX_dSkYsXu-fb14e06i7IgYMS-1x8" \
  -d '{"canteen_name": "North Canteen"}' \
  http://localhost:4000/customer/customer-search-canteen
Summary:
  Total:        0.3585 secs
  Slowest:      0.1893 secs
  Fastest:      0.0014 secs
  Average:      0.0153 secs
  Requests/sec: 5579.1488

  Total data:   418000 bytes
  Size/request: 209 bytes

Response time histogram:
  0.001 [1]     |
  0.020 [1901]  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.039 [24]  ■   |
  0.058 [7]     |
  0.077 [8]     |
  0.095 [9]     |
  0.114 [9]     |
  0.133 [10]    |
  0.152 [11]    |
  0.171 [10]    |
  0.189 [10]    |


Latency distribution:
  10% in 0.0069 secs
  25% in 0.0087 secs
  50% in 0.0112 secs
  75% in 0.0148 secs
  90% in 0.0165 secs
  95% in 0.0196 secs
  99% in 0.1541 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0004 secs, 0.0014 secs, 0.1893 secs
  DNS-lookup:   0.0002 secs, 0.0000 secs, 0.0110 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0061 secs
  resp wait:    0.0150 secs, 0.0014 secs, 0.1768 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0007 secs
```

```
Status code distribution:
  [200] 2000 responses
```

4.1.2 customer/customer-search-dish

```
    hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk1ODYwNywiZXh
     ._It7Pjc9AHKs0IiX_dSkYsXu-fb14e06i7IgYMS-1x8" \
  -d '{"dishNameToSearch":␣"paneer"}' \
  http://localhost:4000/customer/customer-search-dish

Summary:
  Total:        0.4378 secs
  Slowest:      0.1093 secs
  Fastest:      0.0134 secs
  Average:      0.0215 secs
  Requests/sec: 4568.6700

  Total data:   956000 bytes
  Size/request: 478 bytes

Response time histogram:
  0.013 [1]     |
  0.023 [1757]  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.033 [142]   ■■■   |
  0.042 [0]     |
  0.052 [0]     |
  0.061 [0]     |
  0.071 [0]     |
  0.081 [0]     |
  0.090 [30]    ■    |
  0.100 [37]    ■    |
  0.109 [33]    ■    |


Latency distribution:
  10% in 0.0145 secs
  25% in 0.0154 secs
  50% in 0.0168 secs
  75% in 0.0201 secs
  90% in 0.0238 secs
  95% in 0.0819 secs
  99% in 0.1040 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0003 secs, 0.0134 secs, 0.1093 secs
  DNS-lookup:   0.0001 secs, 0.0000 secs, 0.0026 secs
```

```
  req write:    0.0000 secs, 0.0000 secs, 0.0013 secs
  resp wait:    0.0210 secs, 0.0134 secs, 0.0995 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0010 secs

Status code distribution:
  [200] 2000 responses
```

4.1.3 /customer/customer-fetch-reservations

```
    hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk1ODYwNywiZXh
     ._It7Pjc9AHKs0IiX_dSkYsXu-fb14e06i7IgYMS-1x8" \
  http://localhost:4000/customer/customer-fetch-reservations

Summary:
  Total:        0.4234 secs
  Slowest:      0.0937 secs
  Fastest:      0.0036 secs
  Average:      0.0209 secs
  Requests/sec: 4723.5470

  Total data:   38000 bytes
  Size/request: 19 bytes

Response time histogram:
  0.004 [1]     |
  0.013 [99]   ■■■   |
  0.022 [1491] ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.031 [268]  ■■■■■■■   |
  0.040 [41]   ■   |
  0.049 [0]     |
  0.058 [7]     |
  0.067 [14]    |
  0.076 [19]   ■   |
  0.085 [29]   ■   |
  0.094 [31]   ■   |


Latency distribution:
  10% in 0.0140 secs
  25% in 0.0155 secs
  50% in 0.0177 secs
  75% in 0.0199 secs
  90% in 0.0265 secs
  95% in 0.0555 secs
  99% in 0.0882 secs

Details (average, fastest, slowest):
```

```
  DNS+dialup:   0.0004 secs, 0.0036 secs, 0.0937 secs
  DNS-lookup:   0.0002 secs, 0.0000 secs, 0.0047 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0018 secs
  resp wait:    0.0204 secs, 0.0036 secs, 0.0817 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0009 secs

Status code distribution:
  [200] 2000 responses
```

4.1.4 /customer/customer-add-dish

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk10DYwNywiZXh
     ._It7Pjc9AHKs0IiX_dSkYsXu-fb14e06i7IgYMS-1x8" \
  -d '{"dish_id":␣2}' \
  http://localhost:4000/customer/customer-add-dish

Summary:
  Total:        0.6073 secs
  Slowest:      0.1299 secs
  Fastest:      0.0079 secs
  Average:      0.0302 secs
  Requests/sec: 3293.0897

  Total data:   215549 bytes
  Size/request: 107 bytes

Response time histogram:
  0.008 [1]     |
  0.020 [393] ■■■■■■■■■■■■■■■  |
  0.032 [1209] ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.045 [177] ■■■■■■■  |
  0.057 [71] ■■   |
  0.069 [49] ■■   |
  0.081 [0]     |
  0.093 [0]     |
  0.106 [17] ■   |
  0.118 [55] ■■   |
  0.130 [28] ■   |


Latency distribution:
  10% in 0.0193 secs
  25% in 0.0204 secs
  50% in 0.0238 secs
  75% in 0.0297 secs
  90% in 0.0472 secs
  95% in 0.1028 secs
```

```
  99% in 0.1183 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0009 secs, 0.0079 secs, 0.1299 secs
  DNS-lookup:   0.0007 secs, 0.0000 secs, 0.0147 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0011 secs
  resp wait:    0.0291 secs, 0.0079 secs, 0.1075 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0009 secs

Status code distribution:
  [200] 1900 responses
  [201] 100 responses
```

4.1.5 /customer/customer-remove-dish

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type: application/json" \
  -H "Cookie: token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk10DYwNywiZXh
     ._It7Pjc9AHKs0IiX_dSkYsXu-fb14e06i7IgYMS-1x8" \
  -d '{"dish_id": 2}' \
  http://localhost:4000/customer/customer-remove-dish



Summary:
  Total:        0.6347 secs
  Slowest:      0.1409 secs
  Fastest:      0.0059 secs
  Average:      0.0316 secs
  Requests/sec: 3150.8933

  Total data:   208000 bytes
  Size/request: 104 bytes

Response time histogram:
  0.006 [1]     |
  0.019 [388] ■■■■■■■■■■■■■■■  |
  0.033 [1165] ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.046 [225] ■■■■■■■■■  |
  0.060 [13]    |
  0.073 [69] ■■   |
  0.087 [28] ■   |
  0.100 [32] ■   |
  0.114 [39] ■   |
  0.127 [2]     |
  0.141 [38] ■   |


Latency distribution:
```

```
  10% in 0.0187 secs
  25% in 0.0197 secs
  50% in 0.0236 secs
  75% in 0.0315 secs
  90% in 0.0650 secs
  95% in 0.0923 secs
  99% in 0.1342 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0010 secs, 0.0059 secs, 0.1409 secs
  DNS-lookup:   0.0008 secs, 0.0000 secs, 0.0176 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0013 secs
  resp wait:    0.0303 secs, 0.0059 secs, 0.1155 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0039 secs

Status code distribution:
  [200] 2000 responses
```

4.1.6 /customer/customer-view-order-history

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk1ODYwNywiZXh
     ._It7Pjc9AHKs0IiX_dSkYsXu-fb14e06i7IgYMS-1x8" \
  http://localhost:4000/customer/customer-view-order-history

Summary:
  Total:        0.3258 secs
  Slowest:      0.0645 secs
  Fastest:      0.0015 secs
  Average:      0.0160 secs
  Requests/sec: 6139.1492

  Total data:   110000 bytes
  Size/request: 55 bytes

Response time histogram:
  0.002 [1]     |
  0.008 [30]    ■|
  0.014 [1369]  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■|
  0.020 [285]   ■■■■■■■■|
  0.027 [126]   ■■■■|
  0.033 [89]    ■■■|
  0.039 [0]     |
  0.046 [15]    |
  0.052 [21]    ■|
  0.058 [39]    ■|
  0.064 [25]    ■|
```

```
Latency distribution:
  10% in 0.0106 secs
  25% in 0.0117 secs
  50% in 0.0129 secs
  75% in 0.0147 secs
  90% in 0.0247 secs
  95% in 0.0396 secs
  99% in 0.0619 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0002 secs, 0.0015 secs, 0.0645 secs
  DNS-lookup:   0.0001 secs, 0.0000 secs, 0.0024 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0021 secs
  resp wait:    0.0157 secs, 0.0015 secs, 0.0573 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0009 secs

Status code distribution:
  [200] 2000 responses
```

4.1.7 /customer/customer-view-favourites

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk1ODYwNywiZXh
     ._It7Pjc9AHKs0IiX_dSkYsXu-fb14e06i7IgYMS-1x8" \
  http://localhost:4000/customer/customer-view-favourites

Summary:
  Total:        0.3882 secs
  Slowest:      0.1590 secs
  Fastest:      0.0007 secs
  Average:      0.0184 secs
  Requests/sec: 5151.8619

  Total data:   52000 bytes
  Size/request: 26 bytes

Response time histogram:
  0.001 [1]     |
  0.017 [1159]  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.032 [749]   ■■■■■■■■■■■■■■■■■■■■■■■■■■   |
  0.048 [44]    ■■   |
  0.064 [7]     |
  0.080 [7]     |
  0.096 [7]     |
  0.111 [6]     |
  0.127 [5]     |
  0.143 [6]     |
```

```
  0.159 [9]     |


Latency distribution:
  10% in 0.0095 secs
  25% in 0.0139 secs
  50% in 0.0159 secs
  75% in 0.0185 secs
  90% in 0.0269 secs
  95% in 0.0296 secs
  99% in 0.1137 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0003 secs, 0.0007 secs, 0.1590 secs
  DNS-lookup:   0.0001 secs, 0.0000 secs, 0.0031 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0010 secs
  resp wait:    0.0179 secs, 0.0007 secs, 0.1498 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0005 secs

Status code distribution:
  [200] 2000 responses
```

4.1.8 /customer/customer-wallet

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk2MjQzMiwiZXh
     .L9c90JZqCzLeNYQRu61Nt6yeFHrKBDICuYKLJ5NTE_k" \
  http://localhost:4000/customer/customer-wallet

Summary:
  Total:        1.0402 secs
  Slowest:      0.2956 secs
  Fastest:      0.0038 secs
  Average:      0.0499 secs
  Requests/sec: 1922.6818

  Total data:   158000 bytes
  Size/request: 79 bytes

Response time histogram:
  0.004 [1]     |
  0.033 [259] ■■■■■■■■■■  |
  0.062 [1222] ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.091 [404] ■■■■■■■■■■■■■■■■  |
  0.121 [68] ■■■   |
  0.150 [23] ■■   |
  0.179 [4]     |
  0.208 [1]     |
```

```
  0.237 [4]     |
  0.266 [7]     |
  0.296 [7]     |


Latency distribution:
  10% in 0.0324 secs
  25% in 0.0344 secs
  50% in 0.0403 secs
  75% in 0.0635 secs
  90% in 0.0718 secs
  95% in 0.0992 secs
  99% in 0.1599 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0003 secs, 0.0038 secs, 0.2956 secs
  DNS-lookup:   0.0001 secs, 0.0000 secs, 0.0037 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0008 secs
  resp wait:    0.0496 secs, 0.0038 secs, 0.2879 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0010 secs

Status code distribution:
  [200] 2000 responses
```

4.1.9 /customer/customer-view-profile

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type: application/json" \
  -H "Cookie: token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk2MjQzMiwiZXh
     .L9c90JZqCzLeNYQRu61Nt6yeFHrKBDICuYKLJ5NTE_k" \
  http://localhost:4000/customer/customer-view-profile

Summary:
  Total:        8.3892 secs
  Slowest:      1.7591 secs
  Fastest:      0.0357 secs
  Average:      0.4030 secs
  Requests/sec: 238.4019

  Total data:   794000 bytes
  Size/request: 397 bytes

Response time histogram:
  0.036 [1]     |
  0.208 [156]   ■■■■■   |
  0.380 [466]   ■■■■■■■■■■■■■■   |
  0.553 [1315]  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.725 [7]     |
  0.897 [10]    |
```

```
  1.070 [7]     |
  1.242 [10]    |
  1.414 [9]     |
  1.587 [8]     |
  1.759 [11]    |


Latency distribution:
  10% in 0.2326 secs
  25% in 0.3762 secs
  50% in 0.3915 secs
  75% in 0.4422 secs
  90% in 0.4668 secs
  95% in 0.4803 secs
  99% in 1.4128 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0002 secs, 0.0357 secs, 1.7591 secs
  DNS-lookup:   0.0001 secs, 0.0000 secs, 0.0031 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0011 secs
  resp wait:    0.4026 secs, 0.0357 secs, 1.7505 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0006 secs

Status code distribution:
  [200] 2000 responses
```

4.1.10 /customer/customer-fetch-top5

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk2MjQzMiwiZXh
     .L9c90JZqCzLeNYQRu61Nt6yeFHrKBDICuYKLJ5NTE_k" \
  http://localhost:4000/customer/customer-fetch-top5

Summary:
  Total:        0.7536 secs
  Slowest:      0.1182 secs
  Fastest:      0.0229 secs
  Average:      0.0372 secs
  Requests/sec: 2654.0399

  Total data:   2182000 bytes
  Size/request: 1091 bytes

Response time histogram:
  0.023 [1]     |
  0.032 [1257]  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.042 [285]   ■■■■■■■■■ |
  0.052 [192]   ■■■■■■ |
```

```
  0.061 [66] ■■   |
  0.071 [15]    |
  0.080 [56] ■■   |
  0.090 [49] ■■   |
  0.099 [20] ■   |
  0.109 [30] ■   |
  0.118 [29] ■   |


Latency distribution:
  10% in 0.0250 secs
  25% in 0.0261 secs
  50% in 0.0288 secs
  75% in 0.0381 secs
  90% in 0.0603 secs
  95% in 0.0841 secs
  99% in 0.1116 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0003 secs, 0.0229 secs, 0.1182 secs
  DNS-lookup:   0.0002 secs, 0.0000 secs, 0.0038 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0008 secs
  resp wait:    0.0368 secs, 0.0229 secs, 0.1094 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0004 secs

Status code distribution:
  [200] 2000 responses
```

4.1.11 /customer/customer-fetch-dishes

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk2MjQzMiwiZXh
     .L9c90JZqCzLeNYQRu61Nt6yeFHrKBDICuYKLJ5NTE_k" \
  http://localhost:4000/customer/customer-fetch-dishes

Summary:
  Total:        0.4597 secs
  Slowest:      0.0728 secs
  Fastest:      0.0073 secs
  Average:      0.0228 secs
  Requests/sec: 4350.8618

  Total data:   4722000 bytes
  Size/request: 2361 bytes

Response time histogram:
  0.007 [1]     |
  0.014 [70] ■■   |
```

```
  0.020 [1219] ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.027 [235] ■■■■■■■■■  |
  0.033 [233] ■■■■■■■■■  |
  0.040 [142] ■■■■■■  |
  0.047 [0]     |
  0.053 [10]    |
  0.060 [29] ■   |
  0.066 [40] ■   |
  0.073 [21] ■   |


Latency distribution:
  10% in 0.0148 secs
  25% in 0.0165 secs
  50% in 0.0184 secs
  75% in 0.0250 secs
  90% in 0.0348 secs
  95% in 0.0507 secs
  99% in 0.0692 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0003 secs, 0.0073 secs, 0.0728 secs
  DNS-lookup:   0.0001 secs, 0.0000 secs, 0.0035 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0009 secs
  resp wait:    0.0223 secs, 0.0073 secs, 0.0634 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0008 secs

Status code distribution:
  [200] 2000 responses
```

4.1.12 /customer/customer-fetch-menu

```
hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk2MjQzMiwiZXh
     .L9c90JZqCzLeNYQRu61Nt6yeFHrKBDICuYKLJ5NTE_k" \
 -d '{"canteen_id":␣2␣}'\
  http://localhost:4000/customer/customer-fetch-menu

hey -n 2000 -c 100 -m POST \
  -H "Content-Type:␣application/json" \
  -H "Cookie:␣token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
     eyJ1c2VyX2lkIjoyNiwidXNlclR5cGUiOiJjdXN0b21lciIsImlhdCI6MTc0Mzk2MjQzMiwiZXh
     .L9c90JZqCzLeNYQRu61Nt6yeFHrKBDICuYKLJ5NTE_k" \
 -d '{"canteen_id":␣2␣}'\
  http://localhost:4000/customer/customer-fetch-menu

Summary:
  Total:        0.4881 secs
```

```
  Slowest:      0.1329 secs
  Fastest:      0.0010 secs
  Average:      0.0238 secs
  Requests/sec: 4097.3986

  Total data:   1248000 bytes
  Size/request: 624 bytes

Response time histogram:
  0.001 [1]     |
  0.014 [145]   ■■■■    |
  0.027 [1410]  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |
  0.041 [229]   ■■■■■■    |
  0.054 [148]   ■■■■    |
  0.067 [31]    ■    |
  0.080 [11]    |
  0.093 [6]     |
  0.106 [5]     |
  0.120 [6]     |
  0.133 [8]     |


Latency distribution:
  10% in 0.0152 secs
  25% in 0.0168 secs
  50% in 0.0188 secs
  75% in 0.0251 secs
  90% in 0.0408 secs
  95% in 0.0475 secs
  99% in 0.0925 secs

Details (average, fastest, slowest):
  DNS+dialup:   0.0008 secs, 0.0010 secs, 0.1329 secs
  DNS-lookup:   0.0007 secs, 0.0000 secs, 0.0142 secs
  req write:    0.0000 secs, 0.0000 secs, 0.0019 secs
  resp wait:    0.0228 secs, 0.0010 secs, 0.1119 secs
  resp read:    0.0000 secs, 0.0000 secs, 0.0020 secs

Status code distribution:
  [200] 2000 responses
```