American International University-Bangladesh (AIUB)

Department of Computer Science

Faculty of Science &Technology (FST)

Spring 24 25

Section: B

Software Quality Assurance and Testing

SnackSnap

A Report submitted By

SN	Student Name	Student ID
1	Nahid Hasan Nobil	22-46321-1
2	Muntasir Maruf	22-46620-1
3	Talha Hossain Sifat	22-46344-1

Under the supervision of

Abhijit Bhowmik

Associate Professor, Faculty of Science & Technology, American International University-Bangladesh

Checked By Industry Personnel (Optional)

Name:

Designation:

Company:

Sign:

Date:

Software Test Plan

for

SnackSnap

Version 1.0 approved

Prepared by

Nahid Hasan Nobil

Muntasir Maruf

Talha Hossain Sifat

American International University-Bangladesh

25 June, 2025

Table of Contents

Revision History

Revision	Date	Updated by	Update Comments
0.1	2025.06.15	Nahid Hasan Nobil	First Draft
0.2	2025.06.16	Muntasir Maruf	Second Draft
0.3	2025.06.17	Talha Hossain Sifat	Third Draft
0.4	2025.06.18	Nahid Hasan Nobil	Fourth Draft
0.5	2025.06.19	Muntasir Maruf	Fifth Draft
0.6	2025.06.20	Talha Hossain Sifat	Sixth Draft
0.7	2025.06.21	Nahid Hasan Nobil	Seventh Draft
0.8	2025.06.23	Muntasir Maruf	Eighth Draft
0.9	2025.06.25	Talha Hossain Sifat	Final Draft

1. TEST PLAN IDENTIFIER: SnackSnap

2. REFERENCES

- Software Requirement Specification (SRS) Document

3. INTRODUCTION

Background to the Problem

The food delivery industry, particularly in the fast-paced snack and casual dining segment, has seen a dramatic shift in consumer expectations toward convenience, personalization, and speed. However, traditional food ordering methods still depend heavily on phone calls, manual notetaking, and static menus, resulting in order inaccuracies, long wait times, and limited restaurant visibility. These outdated systems create bottlenecks that hinder customer satisfaction and restaurant efficiency.

Root Cause: The core issue lies in the absence of a centralized, interactive, and real-time food ordering platform. Manual order taking often leads to incorrect orders, missed customizations, and delays. Additionally, customers are unable to browse the full range of available items or receive updates on the order status, creating a fragmented and frustrating experience. This lack of automation and feedback hinders both customer satisfaction and restaurant operations, especially during peak hours. This problem is critical in the context of urban lifestyles, where customers expect fast, accurate, and personalized service at their fingertips. Restaurants, especially small to mid-sized vendors, struggle to keep up with larger competitors who use advanced digital platforms. As a result, local businesses miss out on growth opportunities while consumers experience inconsistent service quality.

Solution to the Problem

To overcome these challenges, we propose **SnackSnap** — a dynamic, web-based snack ordering system that bridges the gap between customers and restaurants through a seamless digital experience. SnackSnap empowers users to browse, customize, and order snacks from a wide range of eateries with just a few clicks. The system also allows restaurants to manage menus, process orders, and track customer interactions through a dedicated admin dashboard.

By leveraging responsive design, real-time Ajax updates, and secure payment processing, SnackSnap delivers a fast, user-friendly, and mobile-compatible solution tailored for today’s tech-savvy consumers. This solution aligns with current trends in food tech and digital convenience, providing both customers and restaurant owners with an intuitive, accessible, and scalable platform. The feasibility of this solution is high due to its simple architecture, cross-device compatibility, and low infrastructure requirement. SnackSnap caters to a wide demographic, including busy professionals, students, and casual diners, while offering restaurants the tools they need to expand reach and streamline operations.

Software Description and Purpose

SnackSnap is a web-based snack ordering system developed to modernize and simplify the process of browsing, selecting, and ordering food online. The platform serves two main user groups: **customers**, who can register, browse dishes, place orders, and track them in real-time; and **restaurant administrators**, who can manage dishes, monitor orders, and update system content through a secure admin panel. Key features include:

- **User Authentication**: Secure login and registration for users and admins.
- **Browse and Order**: Customers can explore menus, filter by cuisine, and place customized orders.
- **Admin Dashboard**: Allows restaurants to manage items, prices, orders, and user data.
- **Real-Time Updates**: Orders and availability are updated dynamically using Ajax for smooth user experience.
- **Responsive Design**: Fully functional across desktops, tablets, and smartphones.
- **Secure Payments**: Multiple payment options including cards, wallets, and cash-on-delivery (COD).

The purpose of this software is to create a centralized and automated platform that connects food lovers with vendors in an efficient and satisfying manner. The goal is to enhance customer convenience, support small food businesses, and drive digital innovation in the snack ordering domain.

Existing Studies and Solutions

Various food ordering platforms exist in the market, such as **Foodpanda**, **Uber Eats**, and **Zomato**. While these platforms offer powerful features, they often involve high commissions, lack flexibility for smaller vendors, and don’t provide full control over branding and customer experience. Moreover, these systems are not always optimized for snack-specific vendors or student-run eateries that require a simpler, more affordable digital interface.

SnackSnap is designed to fill this gap by offering a customizable, cost-effective alternative for independent restaurants, food stalls, and home-based snack providers. It provides a modern yet simplified experience without the overhead of larger commercial platforms, ensuring local vendors can thrive while delivering high-quality service to their customers.

4. REQUEIREMNT SPECIFICATION

4.1 System Features

1. System Login

Functional Requirements:
1.1 The system shall allow users to log in using their registered email and password.
1.2 If the email and/or password is entered incorrectly more than three times, the system shall generate a verification challenge to proceed.
1.3 If the number of login attempts exceeds five times, the system shall lock the account temporarily for 30 minutes.

Priority Level: High
Precondition: The user must have a registered account with valid login credentials.

2. Registration

Functional Requirements:
2.1 The system shall provide a registration page for new users to create an account by providing their name, email, phone number, and password.
2.2 The system shall verify the user's email address during registration via an activation link.
2.3 The system shall enforce strong passwords (minimum 8 characters with uppercase, lowercase, numeric, and special characters).
2.4 The system shall allow users to log in after successful registration and email verification.
2.5 The system shall provide a password recovery option through a secure reset link sent via email.

Priority Level: High
Precondition: The user must provide valid and unique registration information.

3. Food Item Management (Inventory)

Functional Requirements:
3.1 The system shall allow restaurant admins to add new food items with details such as name, image, price, description, and availability.
3.2 The system shall allow admins to update existing food item information (e.g., price, availability, or image).
3.3 The system shall allow admins to remove food items from the catalog.
3.4 The system shall provide real-time visibility of available food items to customers.
3.5 The system shall support categorization (e.g., snacks, beverages, desserts) and allow filtering by category.

Priority Level: High
Precondition: The user must be an authenticated restaurant admin with edit privileges.

4. Admin Dashboard Management

Functional Requirements:
4.1 The system shall provide an admin dashboard to view and manage food items, users, and orders.
4.2 The dashboard shall display statistics (e.g., total orders, most ordered items, total users).
4.3 The system shall allow the admin to add or remove restaurant profiles and menu data.
4.4 The system shall include search, filter, and pagination capabilities for managing large datasets.
4.5 The system shall log all major admin actions for traceability.

Priority Level: High
Precondition: The user must be logged in as an admin.

5. Customer Profile Management

Functional Requirements:
5.1 The system shall allow customers to view and update their profile details, such as name, phone, and delivery address.
5.2 The system shall maintain order history and allow users to reorder from past purchases.
5.3 The system shall allow customers to save multiple addresses for faster checkout.
5.4 The system shall provide personalized dish recommendations based on order history.
5.5 The system shall let users change their password and view login history.

Priority Level: Medium
Precondition: The user must be logged in with a valid customer account.

6. Order Processing

Functional Requirements:
6.1 The system shall allow users to browse items and add them to the cart with quantity and preferences.
6.2 The system shall provide a real-time cart summary and allow customers to update or remove items.
6.3 The system shall support checkout with multiple payment methods (e.g., credit/debit card, wallet, COD).
6.4 The system shall send email/SMS notifications for order confirmation and updates.
6.5 The system shall allow users to cancel orders before they are processed or marked as out for delivery.

Priority Level: High
Precondition: The user must be logged in and have items in their cart with valid delivery information.

7. Real-Time Updates & Ajax Functionality

Functional Requirements:
7.1 The system shall use Ajax to dynamically load and update food items without reloading the page.
7.2 The cart, menu filters, and search results shall update in real-time for a seamless experience.
7.3 The system shall notify users of changes in order status using live updates.
7.4 Admins shall see real-time order placement and customer interactions.

Priority Level: High
Precondition: User must have a compatible browser and be actively using the site.

8. User Management

Functional Requirements:
8.1 The system shall allow admin users to manage all registered user accounts.
8.2 Admins can promote users to moderator roles (if required), or deactivate accounts.
8.3 The system shall store user activity logs for audit purposes.
8.4 The system shall implement role-based access control to restrict sensitive operations to authorized users only.
8.5 The system shall display a user list with search and filtering options for admin ease.

Priority Level: High
Precondition: Admin must be logged in with appropriate permissions.

4.2 System Quality Attributes

1. Usability

- A restaurant admin should be able to add a new food item within 2 minutes (maximum 4 minutes).
- A customer shall complete a snack order within 3 minutes (maximum 5 minutes).
- The system should have a user-friendly interface with clear navigation, tooltips, form validations, and informative error messages to guide users effectively.

2. Performance

- Web pages, including menu browsing and checkout, shall load in under 3 seconds.
- The system shall support up to 1,000 concurrent users without noticeable performance degradation.
- Cart updates, item filtering, and order placements shall occur in real-time with a delay not exceeding 2 seconds.
- Admin reports (e.g., daily sales, most ordered items) shall generate in under 10 seconds for up to 10,000 records.

3. Reliability

- The system shall maintain 99.9% uptime to ensure uninterrupted access.
- All orders and transactions shall be processed accurately without data loss or duplication.
- Any critical system errors (e.g., order failure, database disconnection) shall be detected and recovered within 5 minutes.
- The system shall ensure consistency in user data, order history, and menu listings across sessions.

4. Security

- All sensitive user data, including passwords and payment details, shall be encrypted using AES-256 encryption.
- The system shall implement role-based access control to restrict features based on user roles (e.g., customer, admin).
- Two-factor authentication (2FA) shall be enabled for admin access and critical operations such as financial reporting.
- User activities, including login attempts, order modifications, and admin actions, shall be recorded in secure audit logs.

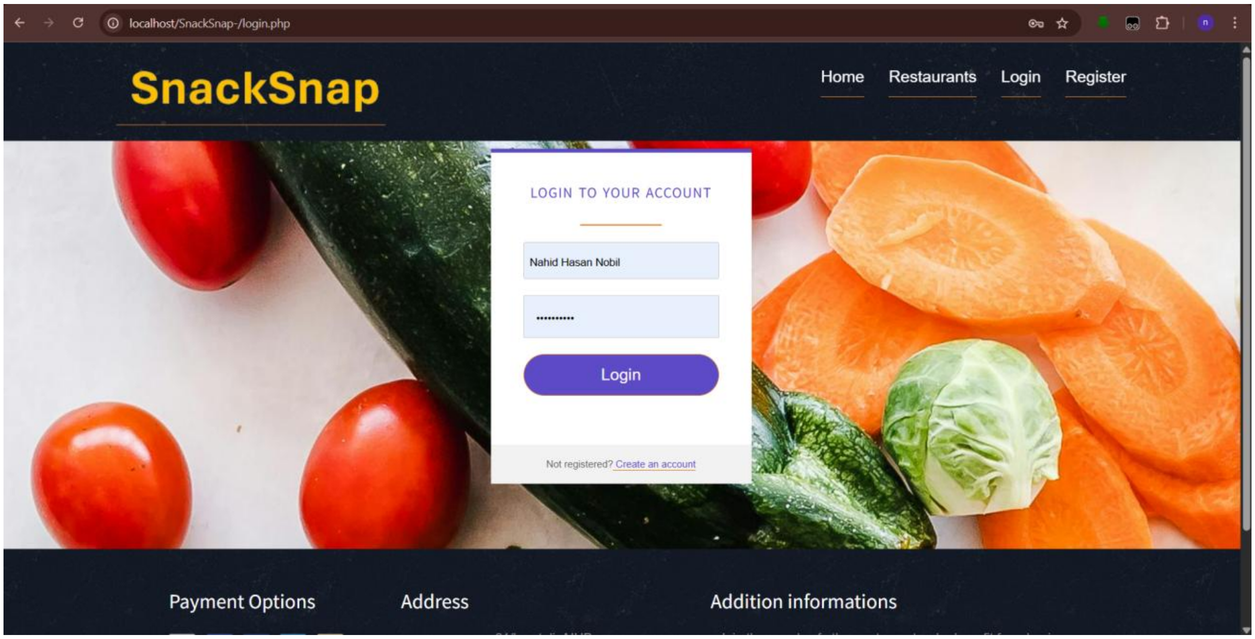
5. Scalability

- The system architecture shall support the addition of new features (e.g., loyalty points, discount coupons) with minimal downtime.
- SnackSnap shall be compatible with third-party service integration, including SMS gateways, payment processors, and email APIs.

6. Accessibility

- The system shall conform to WCAG 2.1 Level AA standards to ensure accessibility.
- It shall support screen readers, high-contrast modes, and full keyboard navigation to assist users with disabilities.

4.3 System Interface


localhost/SnackSnap-/login.php

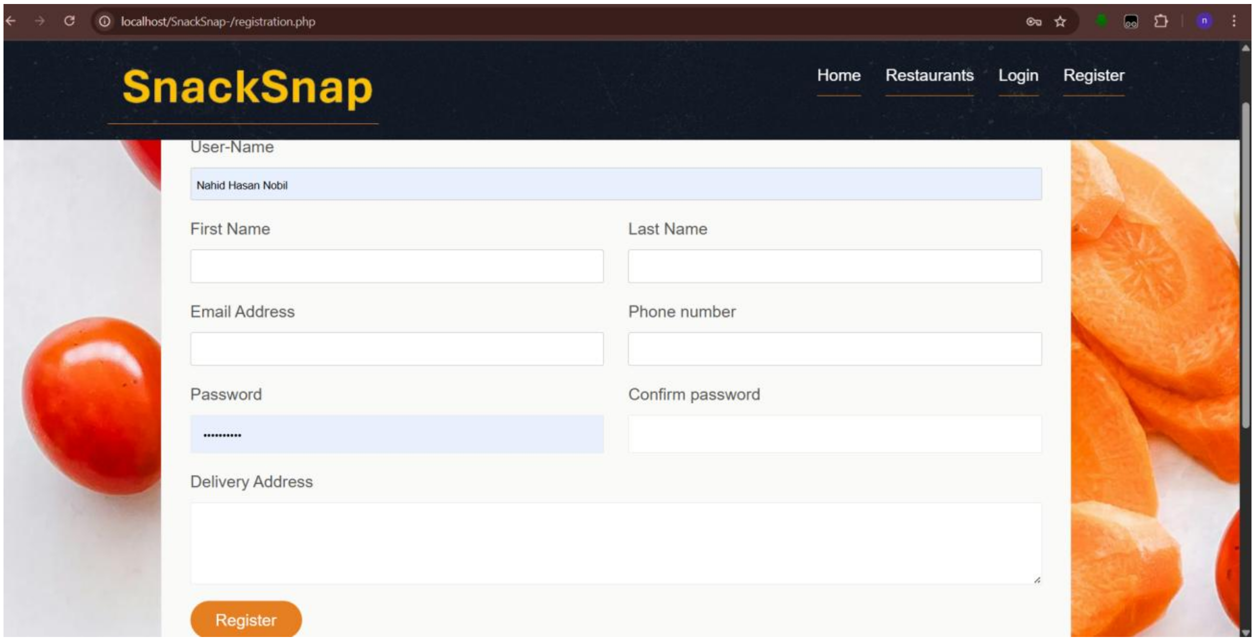
LOGIN TO YOUR ACCOUNT

Nahid Hasan Nobil

••••••••••

Not registered? Create an account

Fig1: Login Page


localhost/SnackSnap-/registration.php

User-Name: Nahid Hasan Nobil

First Name:

Last Name:

Email Address:

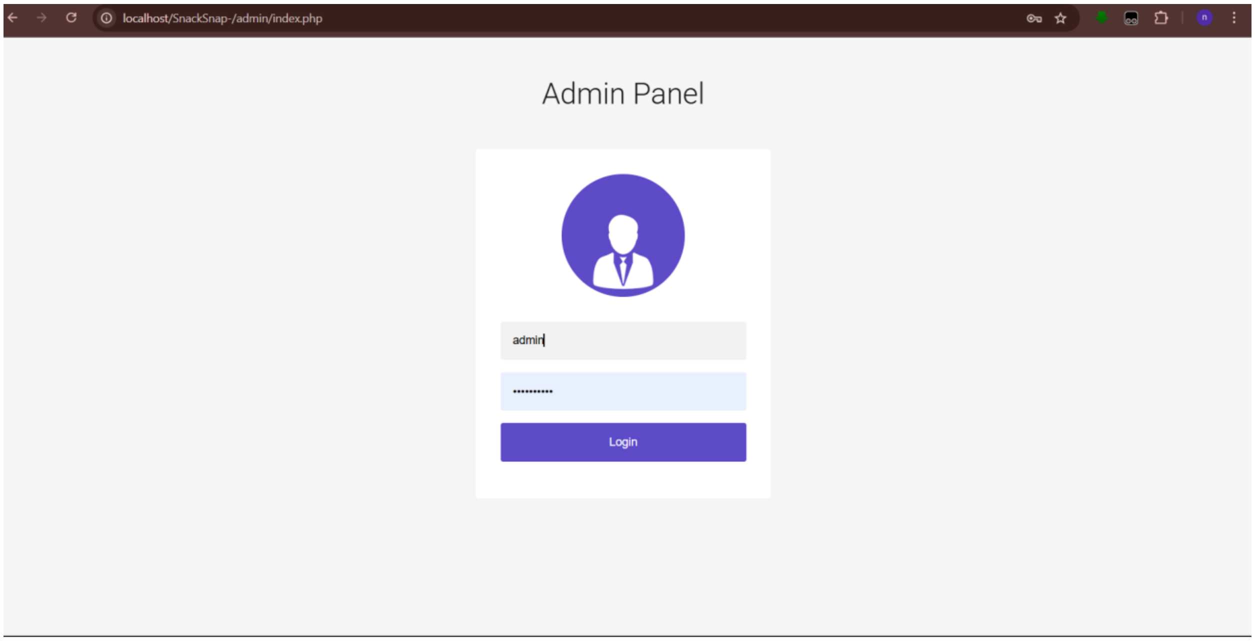
Phone number:

Password: ••••••••••

Confirm password:

Delivery Address:

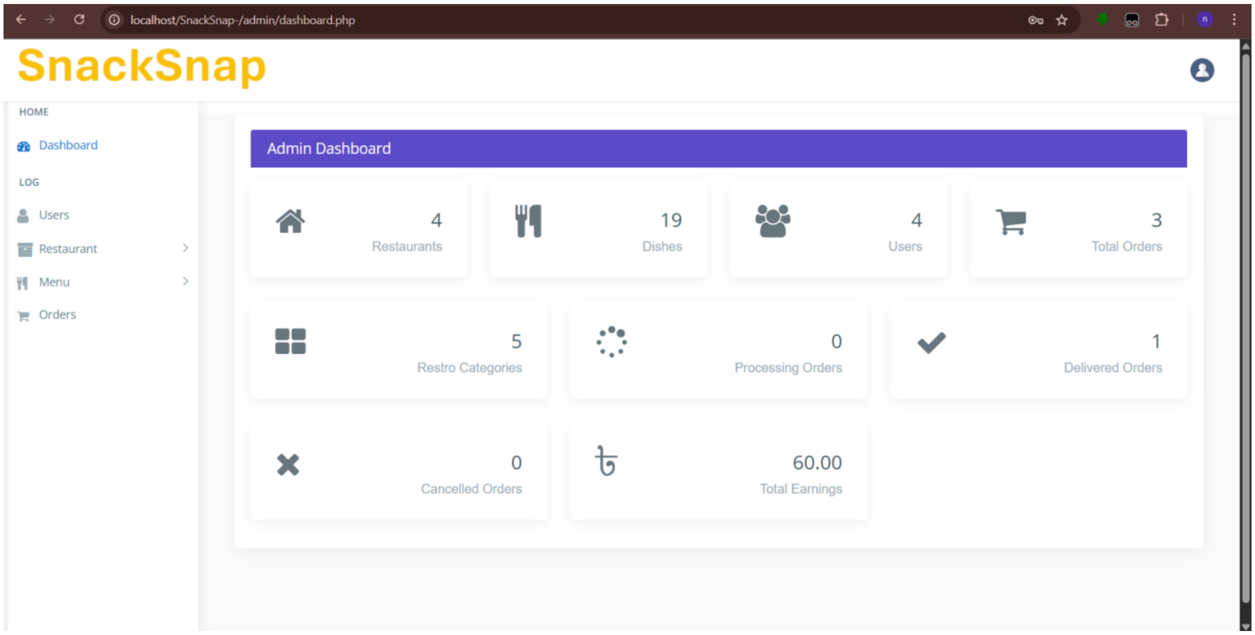
Fig2: Registration Page



admin

••••••••••

Fig3: Admin Login Page



4 Restaurants

19 Dishes

4 Users

3 Total Orders

5 Restro Categories

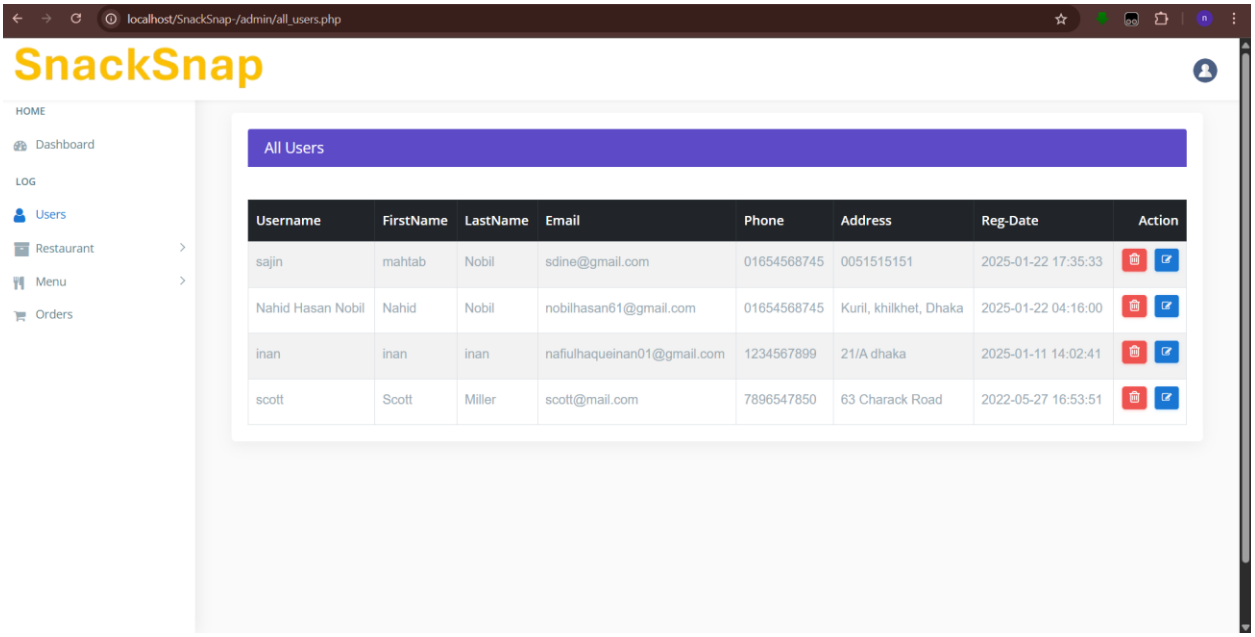
0 Processing Orders

1 Delivered Orders

0 Cancelled Orders

60.00 Total Earnings

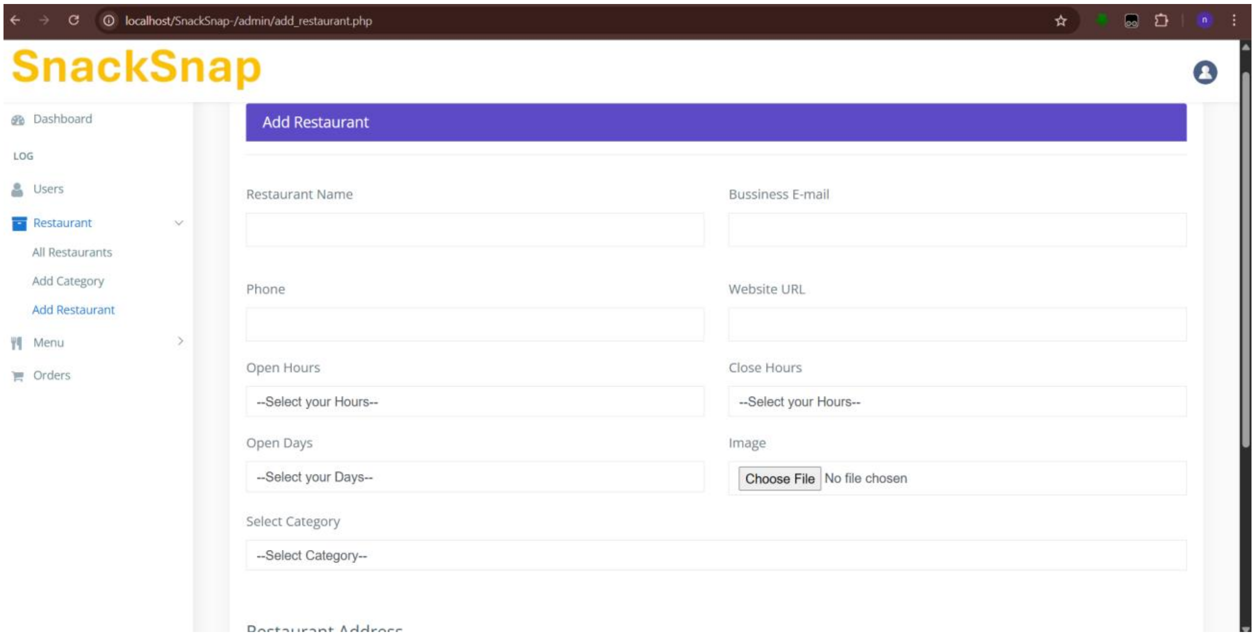
Fig4: Admin Dashboard Page



All Users

Username	FirstName	LastName	Email	Phone	Address	Reg-Date	Action
sajin	mahtab	Nobil	sdine@gmail.com	01654568745	0051515151	2025-01-22 17:35:33	🗑 ✎
Nahid Hasan Nobil	Nahid	Nobil	nobilhasan61@gmail.com	01654568745	Kuril, khilkhet, Dhaka	2025-01-22 04:16:00	🗑 ✎
inan	inan	inan	nafiulhaqueinan01@gmail.com	1234567899	21/A dhaka	2025-01-11 14:02:41	🗑 ✎
scott	Scott	Miller	scott@mail.com	7896547850	63 Charack Road	2022-05-27 16:53:51	🗑 ✎

Fig5: User management Page



Add Restaurant

Restaurant Name

Bussiness E-mail

Phone

Website URL

Open Hours

--Select your Hours--

Close Hours

--Select your Hours--

Open Days

--Select your Days--

Image

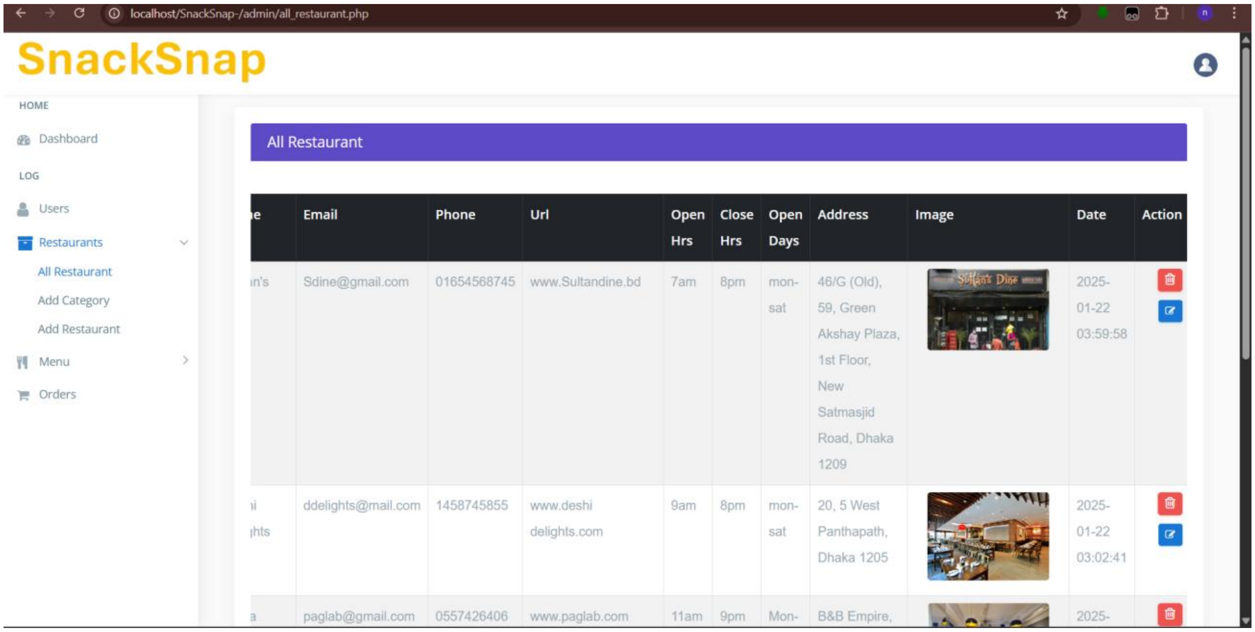
Choose File No file chosen

Select Category

--Select Category--

Restaurant Address

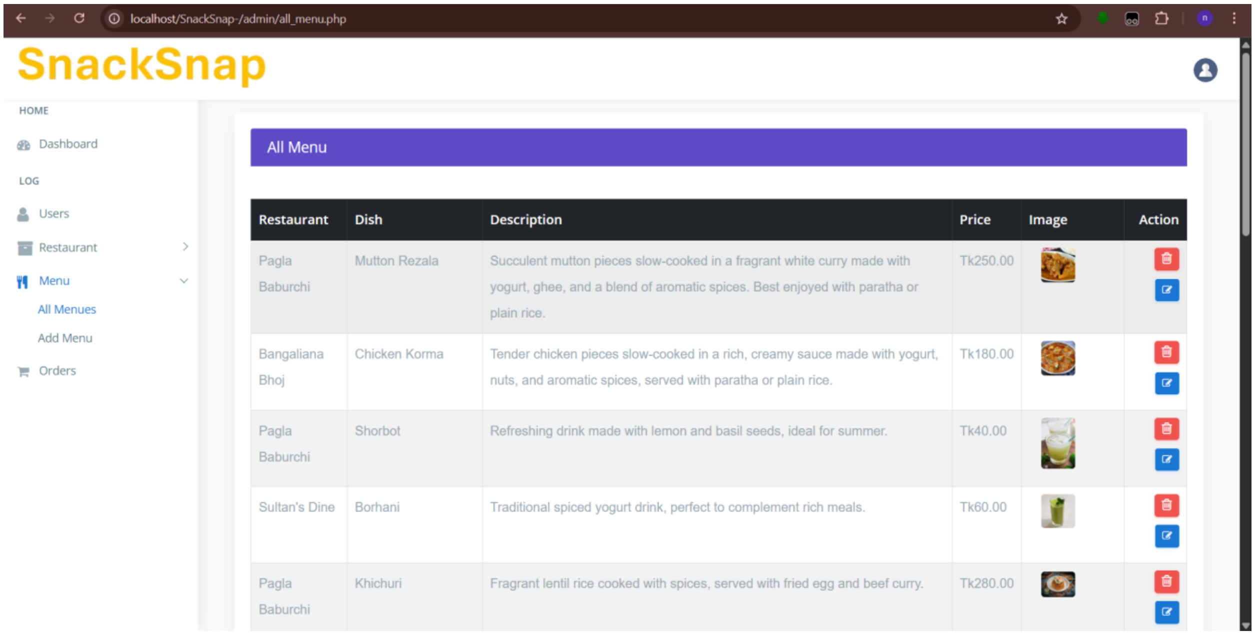
Fig6: Add Restaurant Page



All Restaurant

e	Email	Phone	Url	Open Hrs	Close Hrs	Open Days	Address	Image	Date	Action
n's	Sdine@gmail.com	01654568745	www.Sultandine.bd	7am	8pm	mon-sat	46/G (Old), 59, Green Akshay Plaza, 1st Floor, New Satmasjid Road, Dhaka 1209		2025-01-22 03:59:58	
i hts	ddelights@mail.com	1458745855	www.deshi delights.com	9am	8pm	mon-sat	20, 5 West Panthapath, Dhaka 1205		2025-01-22 03:02:41	
a	paglab@gmail.com	0557426406	www.paglab.com	11am	9pm	Mon-	B&B Empire,		2025-	

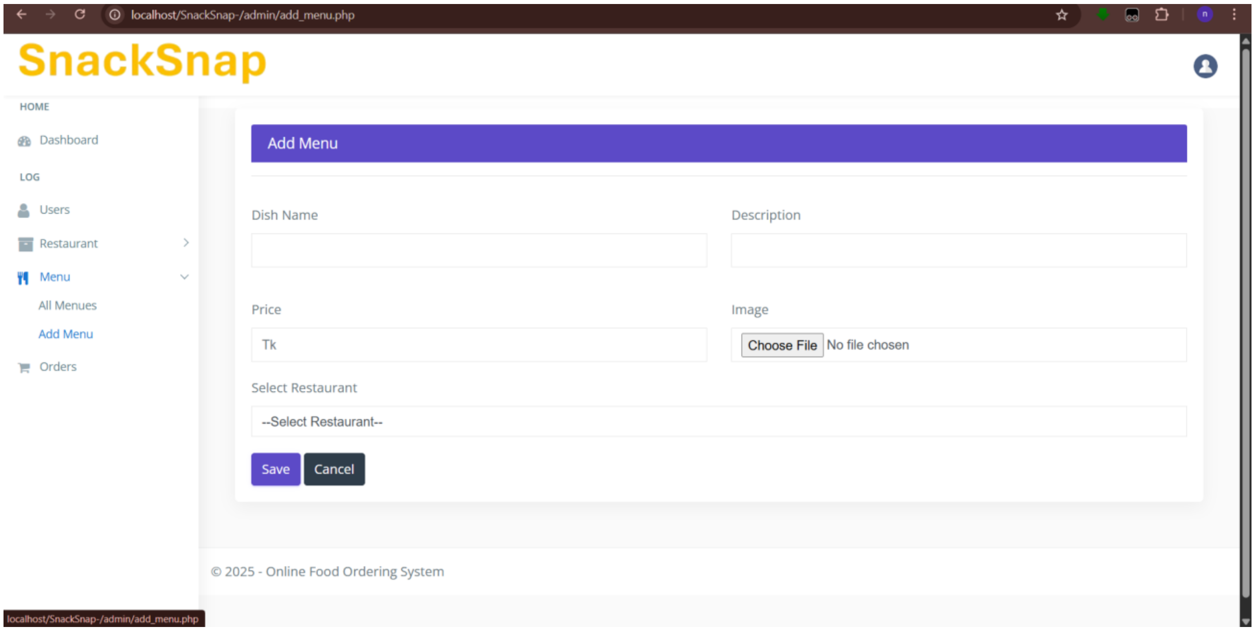
Fig7: All Restaurant Page



All Menu

Restaurant	Dish	Description	Price	Image	Action
Pagla Baburchi	Mutton Rezala	Succulent mutton pieces slow-cooked in a fragrant white curry made with yogurt, ghee, and a blend of aromatic spices. Best enjoyed with paratha or plain rice.	Tk250.00		
Bangaliana Bhoj	Chicken Korma	Tender chicken pieces slow-cooked in a rich, creamy sauce made with yogurt, nuts, and aromatic spices, served with paratha or plain rice.	Tk180.00		
Pagla Baburchi	Shorbot	Refreshing drink made with lemon and basil seeds, ideal for summer.	Tk40.00		
Sultan's Dine	Borhani	Traditional spiced yogurt drink, perfect to complement rich meals.	Tk60.00		
Pagla Baburchi	Khichuri	Fragrant lentil rice cooked with spices, served with fried egg and beef curry.	Tk280.00		

Fig8: All Menu Page



Add Menu

Dish Name

Description

Price

Tk

Image

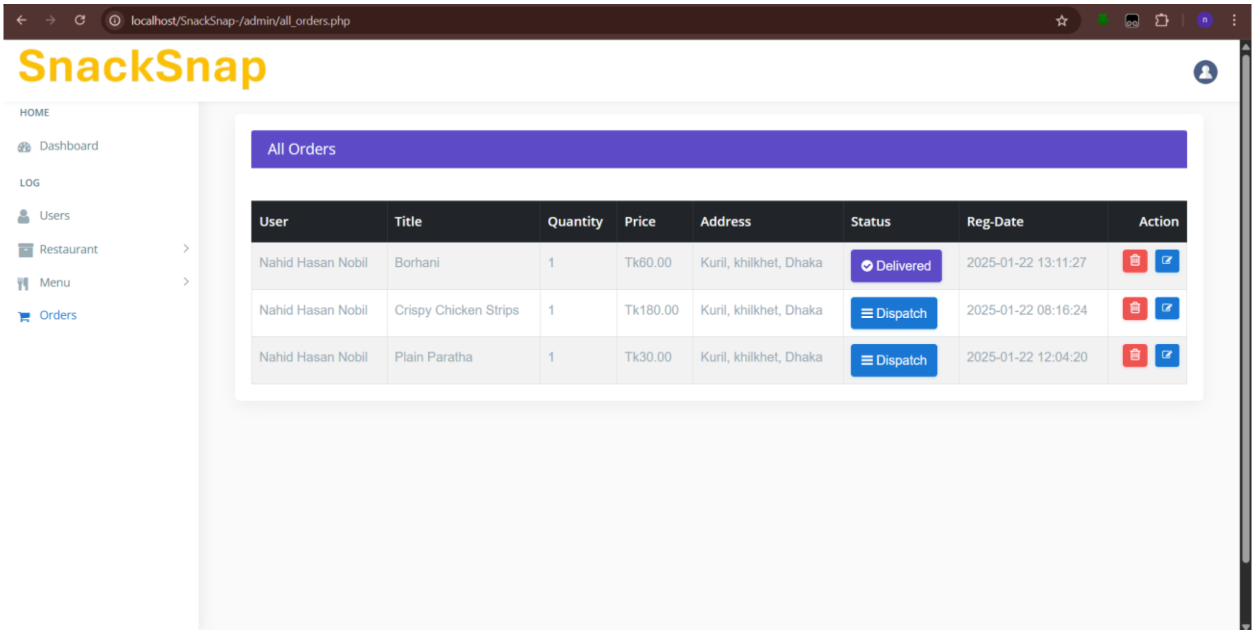
No file chosen

Select Restaurant

--Select Restaurant--

© 2025 - Online Food Ordering System

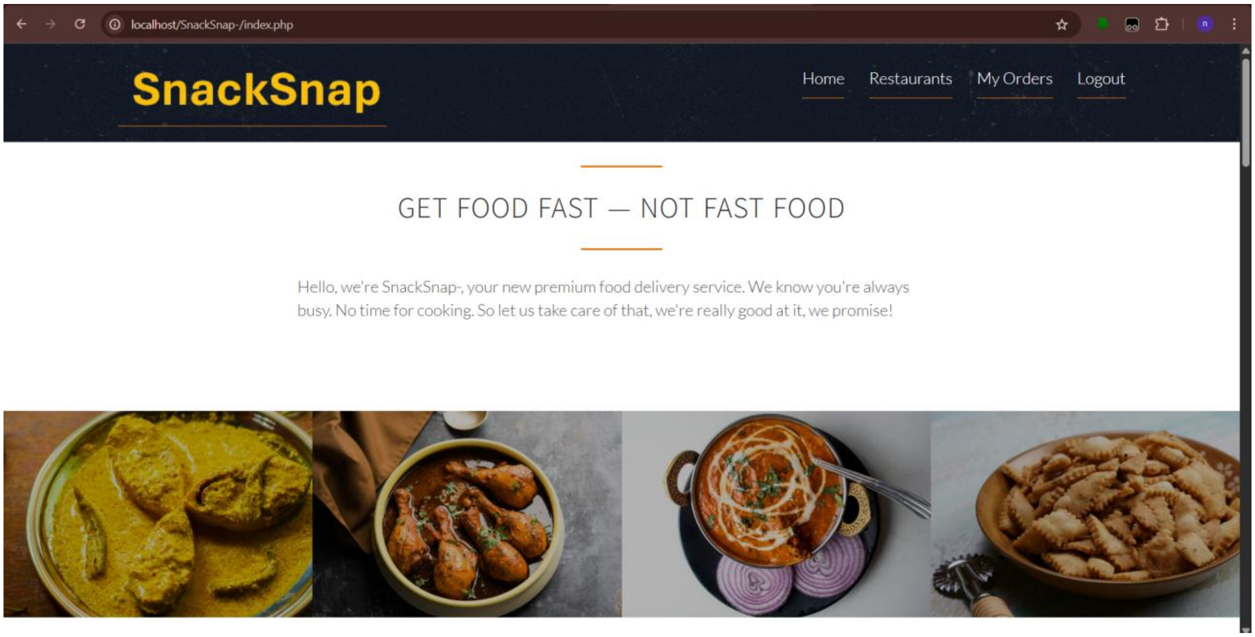
Fig9: Add Menu Page



All Orders

User	Title	Quantity	Price	Address	Status	Reg-Date	Action
Nahid Hasan Nobil	Borhani	1	Tk60.00	Kuril, khilkhet, Dhaka	Delivered	2025-01-22 13:11:27	
Nahid Hasan Nobil	Crispy Chicken Strips	1	Tk180.00	Kuril, khilkhet, Dhaka	Dispatch	2025-01-22 08:16:24	
Nahid Hasan Nobil	Plain Paratha	1	Tk30.00	Kuril, khilkhet, Dhaka	Dispatch	2025-01-22 12:04:20	

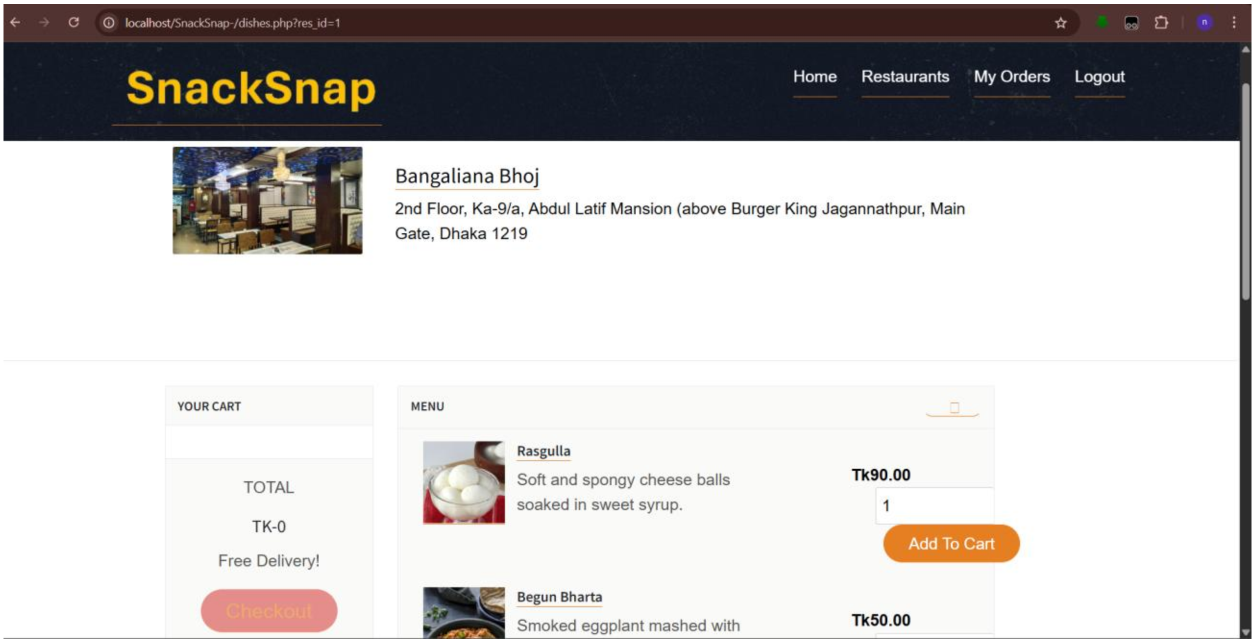
Fig10: Add Order Page



GET FOOD FAST — NOT FAST FOOD

Hello, we're SnackSnap-, your new premium food delivery service. We know you're always busy. No time for cooking. So let us take care of that, we're really good at it, we promise!

Fig11: Home Page



Bangaliana Bhoj

2nd Floor, Ka-9/a, Abdul Latif Mansion (above Burger King Jagannathpur, Main Gate, Dhaka 1219

TOTAL

TK-0

Free Delivery!

Rasgulla

Soft and spongy cheese balls soaked in sweet syrup.

Tk90.00

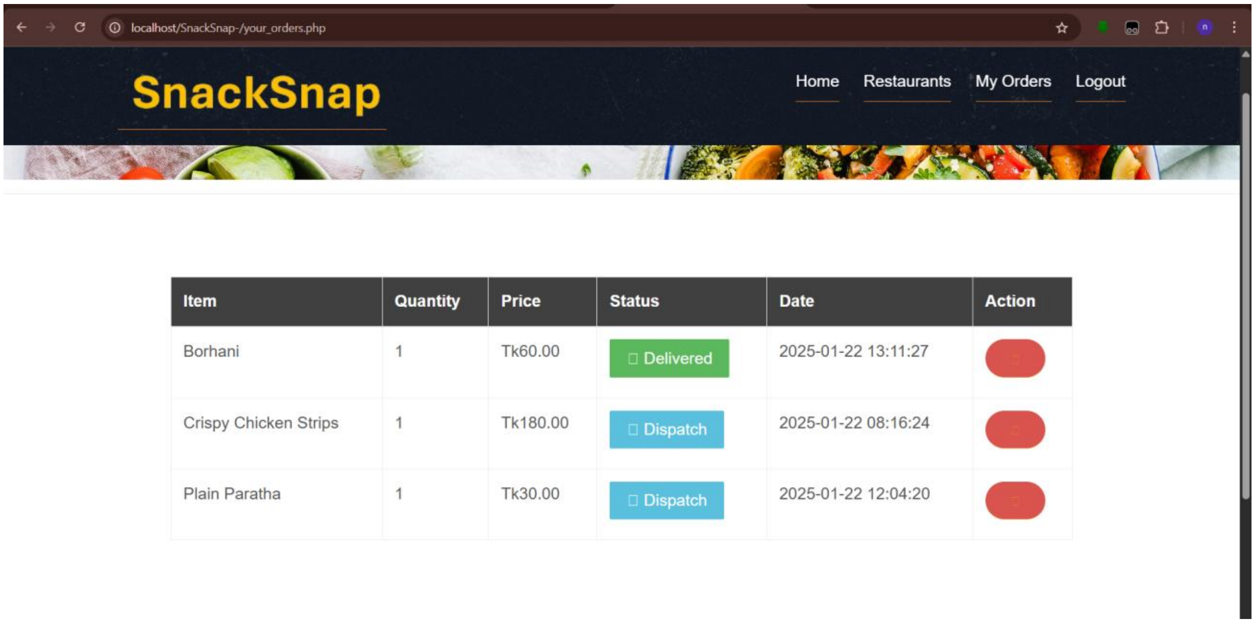
1

Begun Bharta

Smoked eggplant mashed with

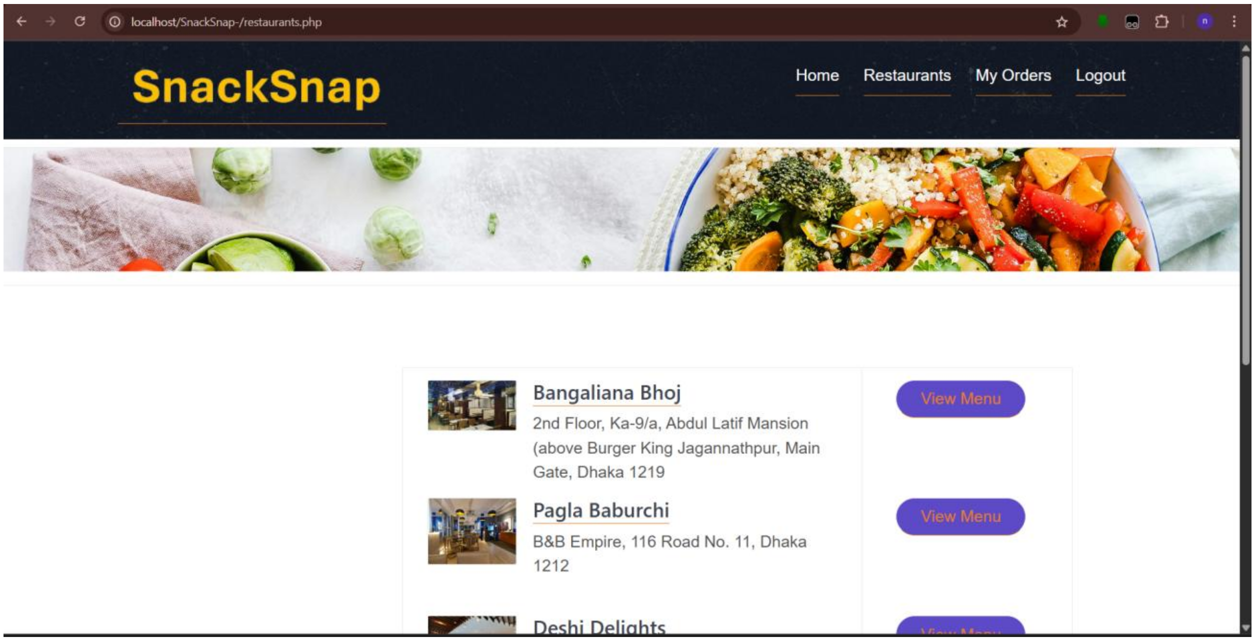
Tk50.00

Fig12: Order Page


localhost/SnackSnap-/your_orders.php

Item	Quantity	Price	Status	Date	Action
Borhani	1	Tk60.00	□ Delivered	2025-01-22 13:11:27	
Crispy Chicken Strips	1	Tk180.00	□ Dispatch	2025-01-22 08:16:24	
Plain Paratha	1	Tk30.00	□ Dispatch	2025-01-22 12:04:20	

Fig13: My Order Page

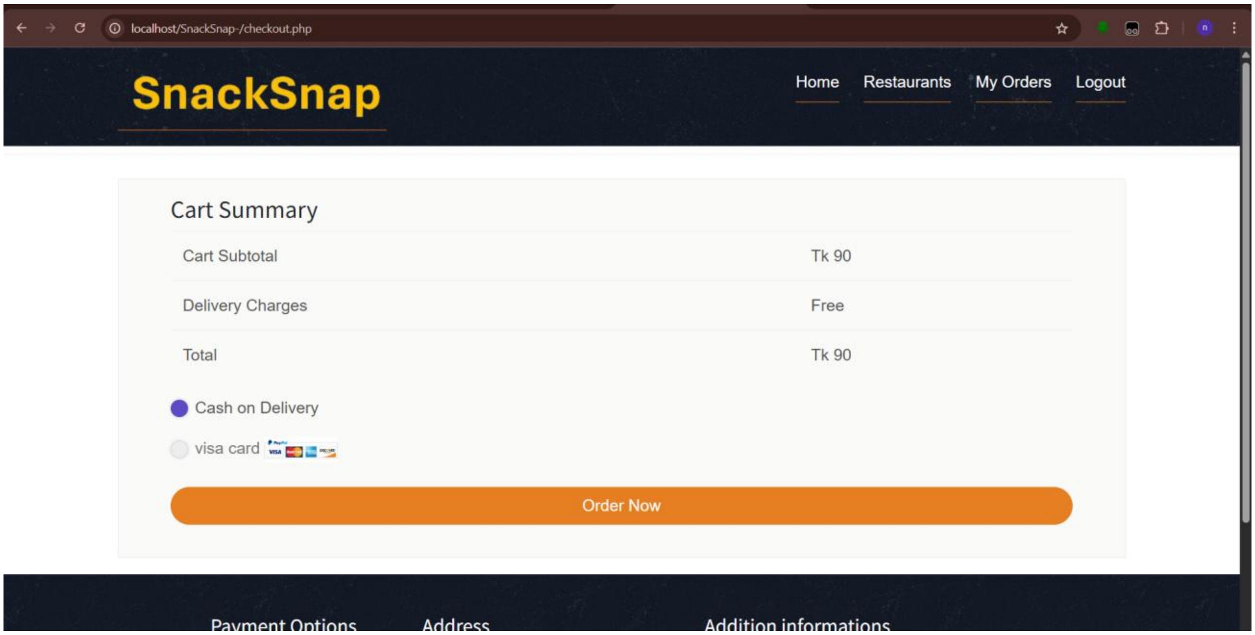

localhost/SnackSnap-/restaurants.php

Bangaliana Bhoj
2nd Floor, Ka-9/a, Abdul Latif Mansion (above Burger King Jagannathpur, Main Gate, Dhaka 1219

Pagla Baburchi
B&B Empire, 116 Road No. 11, Dhaka 1212

Deshi Delights

Fig14: Restaurant Page



Cart Summary

Cart Subtotal	Tk 90
Delivery Charges	Free
Total	Tk 90

● Cash on Delivery

○ visa card

Fig15: Payment Page

4.4 Project Requirements

The **Constructive Cost Model (COCOMO)** is an algorithmic software cost estimation model. For the SnackSnap project, the **semi-detached** model is appropriate, as it falls between simple organic projects and complex embedded systems. It involves moderate experience, a medium-sized development team, and intermediate software complexity.

Using the semi-detached model coefficients:

- Coefficient<Effort Factor> = 3.0
- P (Exponent) = 1.12
- T (Time exponent) = 0.35

The SnackSnap project consists of approximately **28,000 lines of code (SLOC)**, based on the estimation of frontend (HTML, CSS, JS) and backend (PHP, Ajax, MySQL) components.

Effort Estimation:

Effort (PM)=3.0×(28,000/1000)1.12

=3.0×281.12=3.0×43.76

=131.28 labor working hours

Development Time (DM):

DM=2.5×(131.28)0.35

=2.5×5.53

=13.83 months

≈14 months

Required Number of People (Average Team Size):

ST=PMDM

=131.2813.83

≈9.49

≈10 people

Cost Estimation:

Assuming the **average monthly cost per developer is 50,000 BDT**:

- **Salary Budget** = 10 × 50,000 BDT = **5,00,000 BDT**
- **Transport Cost** = 20,000 BDT
- **Miscellaneous Expenses** = 15,000 BDT

Total Project Budget: 5,00,000+20,000+15,000=5,35,000 BDT

5. FEATURES NOT TO BE TESTED

Certain aspects of the SnackSnap online snack ordering system may be excluded from testing, either due to their limited relevance to the core functionality or because they fall outside the scope of the current testing phase. For example, features such as third-party payment gateways, SMS/email delivery services, **or** external map APIs (for location-based services) may not be tested directly, assuming they have been independently verified by their providers.

Additionally, non-functional features such as UI design consistency, color themes, animation effects, or advanced performance benchmarks may be deferred to later testing stages. These elements, while important for user experience, do not impact the fundamental operation of the system and can be evaluated in future user feedback or aesthetic refinement cycles. The focus of this testing phase will remain on validating core functionalities, security mechanisms, and data flow integrity to ensure a stable and usable release.

6. TESTING APPROACH

6.1 Testing Levels

The testing for the SnackSnap online snack ordering system will involve multiple levels, including Unit Testing**,** Integration Testing**,** System Testing, and User Acceptance Testing (UAT). Given the modular structure of the application—comprising user authentication, food menu management, order processing, admin operations, and payment handling—a comprehensive testing strategy will be employed to ensure the system is reliable, secure, and user-friendly. Both the development team and dedicated testers will be actively involved in each phase of the testing lifecycle.

Unit Testing

Unit testing will be conducted by the developers for each individual component of the SnackSnap system. The primary goal is to verify the correctness of individual modules, such as login/logout logic, food item listing, cart functionality, and order placement. Each developer will write test cases to validate the expected behavior, handle edge cases, and confirm proper input-output flow. This stage ensures each module performs as intended before integration with other parts of the system. Unit test results will be documented and shared with the QA lead.

Integration Testing

Integration Testing will focus on how individual modules interact with each other when assembled into the complete system. For example, this includes verifying the smooth interaction between the user registration system**,** order cart**,** checkout page**,** admin backend**,** and payment module. The integration with third-party services such as payment gateways and email/SMS notification APIs will also be tested. The objective is to ensure proper data flow, security, and transaction consistency across all modules. The development team, along with the QA lead, will execute these tests.

System Testing

System Testing will assess whether the entire SnackSnap system functions cohesively and aligns with the defined requirements and specifications. This phase includes end-to-end testing of both frontend (e.g., responsive UI, user interactions) and backend (e.g., database operations, order lifecycle). Performance under load, system behavior during high traffic, and data integrity checks will be conducted to ensure robustness. Stress and recovery tests will be part of this level to measure how the system responds under extreme conditions.

User Acceptance Testing (UAT)

User Acceptance Testing will involve actual end users, including customers and admin users, testing the SnackSnap system in a simulated real-world environment. Users will complete tasks such as browsing items**,** placing an order**,** receiving confirmation, and admin actions like adding a new dish or managing orders. The goal of UAT is to validate that the system is **easy** to use, operationally effective, and meets all practical business requirements. Feedback gathered during UAT will be used to make final refinements. The QA team and developers will assist users during this phase to ensure a smooth testing experience.

This multi-layered testing approach ensures that SnackSnap is not only functionally sound and secure but also user-centric and ready for deployment in a live environment.

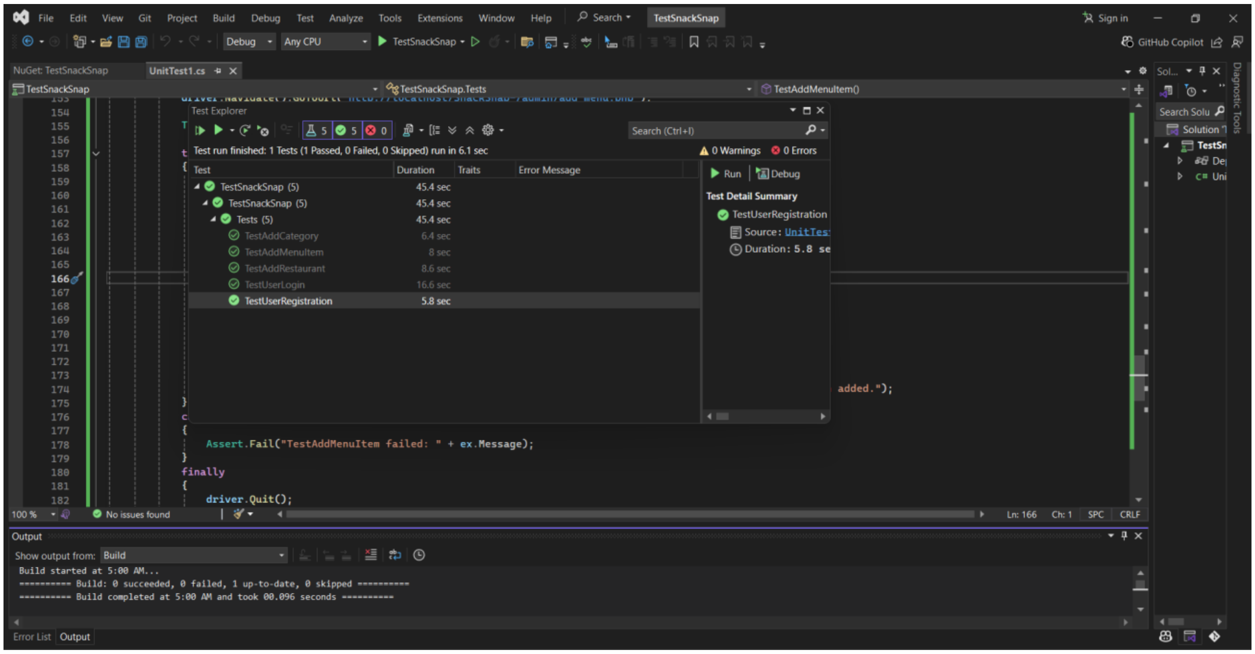
6.2 Test Tools

For testing the SnackSnap project, we used Selenium WebDriver integrated with Visual Studio in an NUnit Test Project (C#) environment. Selenium WebDriver is a robust automation framework that allows us to simulate user interactions and test web-based functionalities effectively. We developed and executed five automated test methods to verify the core functionalities of both customer and admin roles in the system. These tests were scripted in C# and run using the NUnit test runner in Visual Studio.

Automated Test Coverage:

- Customer Tests:
    - Login Test – Verifies that registered customers can log in with valid credentials.
    - Registration Test – Validates new user sign-up and successful account creation.
- Admin Tests**:**
    - Add Restaurant Test – Tests the ability of admin to add a new restaurant to the system.
    - Add Restaurant Category Test – Ensures the admin can define and assign categories (e.g., Snacks, Beverages).
    - Add Menu Item Test – Verifies that new food items can be added under appropriate categories.

Each of these tests was executed through automated browser sessions, where the test scripts mimicked real user behavior—interacting with the UI elements, submitting forms, validating data entry, and capturing results. Selenium helped identify any functional issues by checking DOM elements, form responses, and navigation flows. All tests were executed locally in Chrome using Visual Studio's built-in test explorer.



Test run finished: 1 Tests (1 Passed, 0 Failed, 0 Skipped) run in 6.1 sec

0 Warnings 0 Errors

Test	Duration	Traits	Error Message
TestSnackSnap (5)	45.4 sec		
TestSnackSnap (5)	45.4 sec		
Tests (5)	45.4 sec		
TestAddCategory	6.4 sec		
TestAddMenuItem	8 sec		
TestAddRestaurant	8.6 sec		
TestUserLogin	16.6 sec		
TestUserRegistration	5.8 sec		

Test Detail Summary

TestUserRegistration

Source: UnitTes

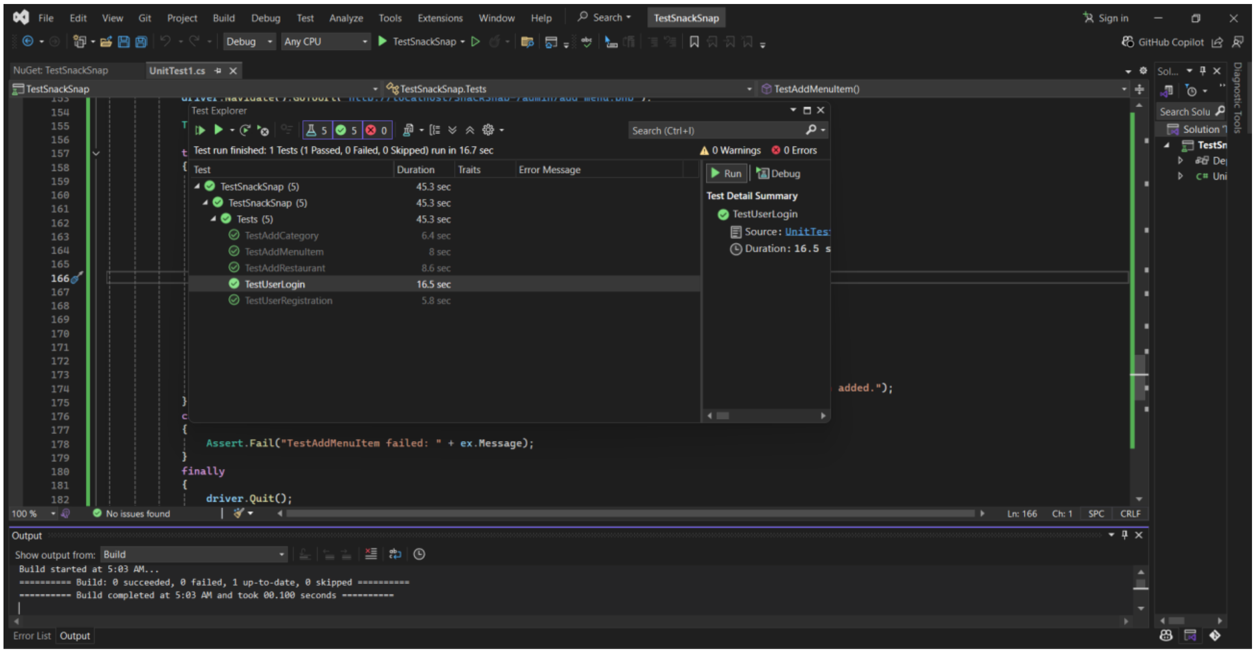
Duration: 5.8 se

```
                    added.");
        }
        c
        {
            Assert.Fail("TestAddMenuItem failed: " + ex.Message);
        }
        finally
        {
            driver.Quit();
```

Show output from: Build

```
Build started at 5:00 AM...
========== Build: 0 succeeded, 0 failed, 1 up-to-date, 0 skipped ==========
========== Build completed at 5:00 AM and took 00.096 seconds ==========
```

Fig1: Customer Registration test



Test run finished: 1 Tests (1 Passed, 0 Failed, 0 Skipped) run in 16.7 sec

0 Warnings 0 Errors

Test	Duration	Traits	Error Message
TestSnackSnap (5)	45.3 sec		
TestSnackSnap (5)	45.3 sec		
Tests (5)	45.3 sec		
TestAddCategory	6.4 sec		
TestAddMenuItem	8 sec		
TestAddRestaurant	8.6 sec		
TestUserLogin	16.5 sec		
TestUserRegistration	5.8 sec		

Test Detail Summary

TestUserLogin

Source: UnitTes

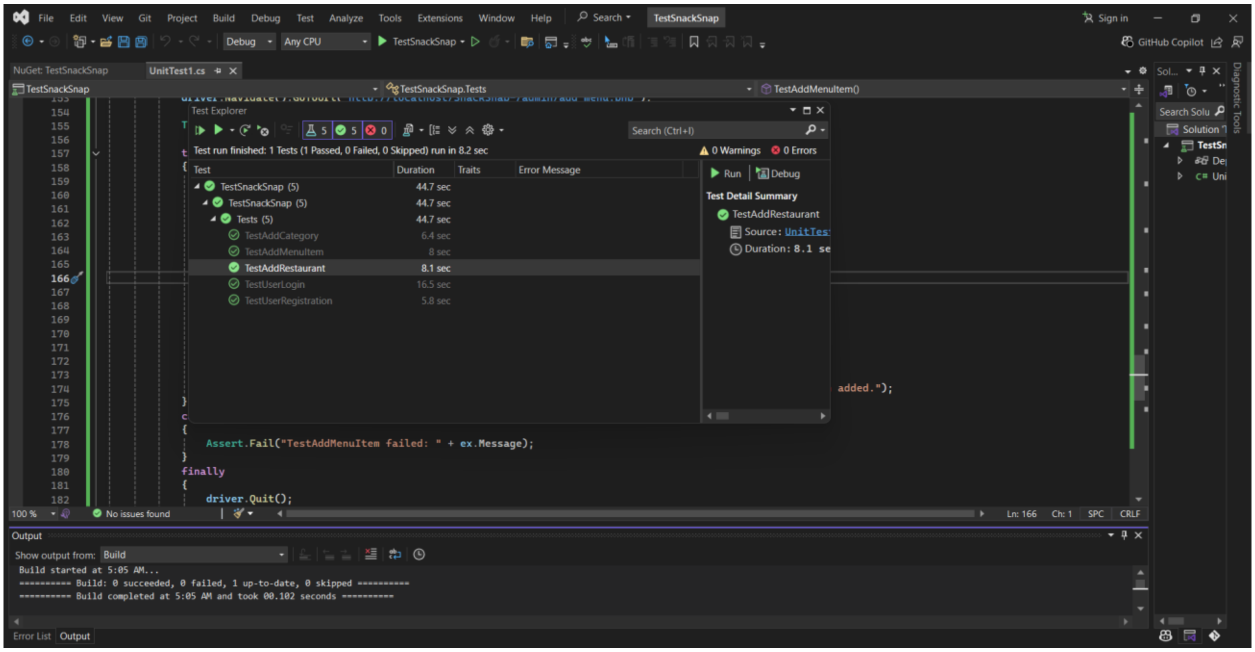
Duration: 16.5 s

```
                                  added.");
        }
        c
        {
            Assert.Fail("TestAddMenuItem failed: " + ex.Message);
        }
        finally
        {
            driver.Quit();
```

Show output from: Build

```
Build started at 5:03 AM...
========== Build: 0 succeeded, 0 failed, 1 up-to-date, 0 skipped ==========
========== Build completed at 5:03 AM and took 00.100 seconds ==========
```

Fig2: Customer login test



Test run finished: 1 Tests (1 Passed, 0 Failed, 0 Skipped) run in 8.2 sec

0 Warnings 0 Errors

Test	Duration	Traits	Error Message
TestSnackSnap (5)	44.7 sec		
TestSnackSnap (5)	44.7 sec		
Tests (5)	44.7 sec		
TestAddCategory	6.4 sec		
TestAddMenuItem	8 sec		
TestAddRestaurant	8.1 sec		
TestUserLogin	16.5 sec		
TestUserRegistration	5.8 sec		

Test Detail Summary

TestAddRestaurant

Source: UnitTes

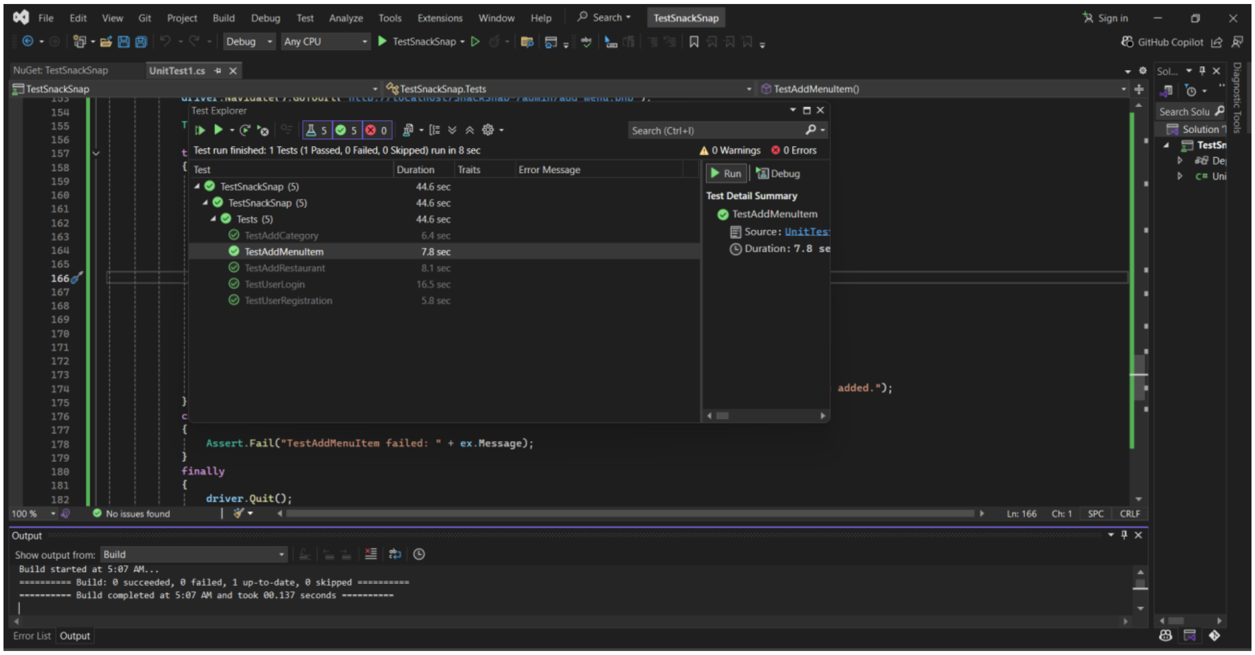
Duration: 8.1 se

```
                                  added.");
        }
        c
        {
            Assert.Fail("TestAddMenuItem failed: " + ex.Message);
        }
        finally
        {
            driver.Quit();
```

Show output from: Build

```
Build started at 5:05 AM...
========== Build: 0 succeeded, 0 failed, 1 up-to-date, 0 skipped ==========
========== Build completed at 5:05 AM and took 00.102 seconds ==========
```

Fig3: Add Restaurant test



Test run finished: 1 Tests (1 Passed, 0 Failed, 0 Skipped) run in 8 sec

0 Warnings 0 Errors

Test	Duration	Traits	Error Message
TestSnackSnap (5)	44.6 sec		
TestSnackSnap (5)	44.6 sec		
Tests (5)	44.6 sec		
TestAddCategory	6.4 sec		
TestAddMenuItem	7.8 sec		
TestAddRestaurant	8.1 sec		
TestUserLogin	16.5 sec		
TestUserRegistration	5.8 sec		

Test Detail Summary

TestAddMenuItem

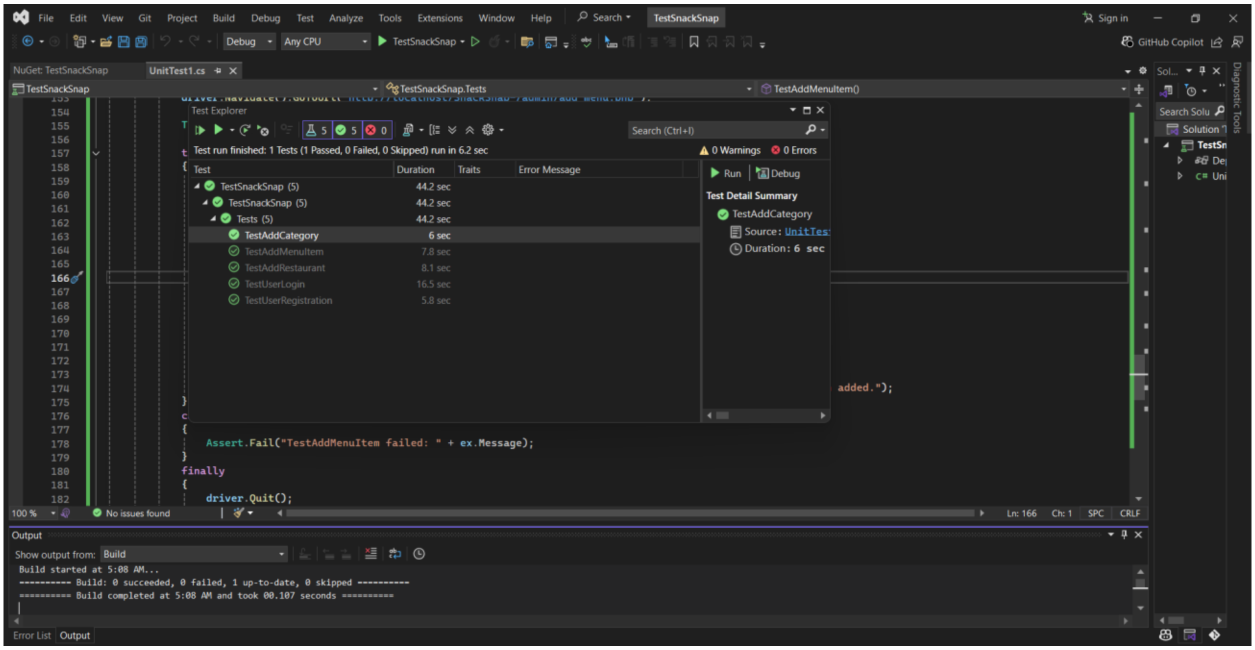
Source: UnitTes

Duration: 7.8 se

```
added.");
        }
        {
            Assert.Fail("TestAddMenuItem failed: " + ex.Message);
        }
        finally
        {
            driver.Quit();
```

```
Build started at 5:07 AM...
========== Build: 0 succeeded, 0 failed, 1 up-to-date, 0 skipped ==========
---------- Build completed at 5:07 AM and took 00.137 seconds ----------
```

Fig4: Add Menu item test



Test run finished: 1 Tests (1 Passed, 0 Failed, 0 Skipped) run in 6.2 sec

0 Warnings 0 Errors

Test	Duration	Traits	Error Message
TestSnackSnap (5)	44.2 sec		
TestSnackSnap (5)	44.2 sec		
Tests (5)	44.2 sec		
TestAddCategory	6 sec		
TestAddMenuItem	7.8 sec		
TestAddRestaurant	8.1 sec		
TestUserLogin	16.5 sec		
TestUserRegistration	5.8 sec		

Test Detail Summary

TestAddCategory

Source: UnitTes

Duration: 6 sec

```
added.");
        }
        {
            Assert.Fail("TestAddMenuItem failed: " + ex.Message);
        }
        finally
        {
            driver.Quit();
```

```
Build started at 5:08 AM...
---------- Build: 0 succeeded, 0 failed, 1 up-to-date, 0 skipped ----------
========== Build completed at 5:08 AM and took 00.107 seconds ==========
```

Fig4: Add Restaurant category test

6.3 Meetings

Distributing tasks effectively to the right individuals is crucial for the success of the testing team. To ensure efficient progress and collaboration, it is essential to hold regular team meetings. Therefore, the testing team met once a week to review the progress of each member, assess whether they had completed their assigned tasks, and identify any challenges or issues they were facing during testing. If any team member encounters difficulties, the entire team tried collaborate to find solutions and resolve the issue promptly.

7. TEST CASES/TEST ITEMS

Test Case 1:

Project Name: SnackSnap	Test Designed by: Nahid Hasan Nobil
Test Case ID: SS_TC_01	Test Designed date: 16/06/25
Test Priority (Low, Medium, High): High	Test Executed by: Nahid Hasan Nobil
Module Name: Login	Test Execution date: 19/06/25
Test Title: Login as Customer	
Description: Test user login functionality with valid credentials	
Precondition (If any): User must be registered.	

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Go to the website 2. Enter username 3. Enter password 4. Click submit	Username: Nahid Hasan Nobil Password: Nahid@1234	User should login into the application	As expected,	Pass
Post Condition: User is authenticated and redirected to homepage with active session.				

Test Case 2:

Project Name: SanckSnap	Test Designed by: Nahid Hasan Nobil
Test Case ID: SS_TC_02	Test Designed date: 16/06/25
Test Priority: High	Test Executed by: Talha Hossain Sifat
Module Name: Registration	Test Execution date: 19/06/25
Test Title: Register new customer	
Description: Valid registration form submission	
Precondition: Email must be unique and format valid.	

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Enter First Name 2. Enter last Name 3. Enter Email 4. Enter Password 4. Enter Confirm Password 4. Enter Address Details 4. Click Register	First Name: Jayed Last Name: Hasan Email: jayed@gmail.com Password: jayed@234 Confirm password: jayed@234 Address Details: 1224-dhaka, Khilkhet	Registration Successful	As expected	Pass
Post Condition: Account is created and Customer details are logged in the database.				

Test Case 3:

<table>
<tr><td colspan="2">Project Name: SanckSnap</td><td colspan="3">Test Designed by: Nahid Hasan Nobil</td></tr>
<tr><td colspan="2">Test Case ID: SS_TC_03</td><td colspan="3">Test Designed date: 16/06/25</td></tr>
<tr><td colspan="2">Test Priority: Medium</td><td colspan="3">Test Executed by: Talha Hossain Sifat</td></tr>
<tr><td colspan="2">Module Name: Cart</td><td colspan="3">Test Execution date: 19/06/25</td></tr>
<tr><td colspan="2">Test Title: Add food item to cart</td><td colspan="3"></td></tr>
<tr><td colspan="2">Description: Test adding item to cart from menu page</td><td colspan="3"></td></tr>
<tr><td colspan="5">Precondition: User must be logged in</td></tr>
<tr><td>Test Steps</td><td>Test Data</td><td>Expected Results</td><td>Actual Results</td><td>Status (Pass/Fail)</td></tr>
<tr><td>1. Select item
2. Click “Add to Cart</td><td>Item: Burger</td><td>Item added to cart</td><td>As expected</td><td>Pass</td></tr>
<tr><td colspan="5">Post Condition: Cart reflects the newly added item. Cart details are logged in the database.</td></tr>
</table>

Test Case 4:

<table>
<tr><td colspan="2">Project Name: SanckSnap</td><td colspan="3">Test Designed by: Nahid Hasan Nobil</td></tr>
<tr><td colspan="2">Test Case ID: SS_TC_04</td><td colspan="3">Test Designed date: 16/06/25</td></tr>
<tr><td colspan="2">Test Priority: Medium</td><td colspan="3">Test Executed by: Talha Hossain Sifat</td></tr>
<tr><td colspan="2">Module Name: Order</td><td colspan="3">Test Execution date: 19/06/25</td></tr>
<tr><td colspan="2">Test Title: Complete order checkout</td><td colspan="3"></td></tr>
<tr><td colspan="2">Description: Test successful order placement</td><td colspan="3"></td></tr>
<tr><td colspan="5">Precondition: User must have items in cart and be logged in</td></tr>
<tr><td>Test Steps</td><td>Test Data</td><td>Expected Results</td><td>Actual Results</td><td>Status (Pass/Fail)</td></tr>
<tr><td>1. Go to Cart
2. Click Checkout
3. Select Payment Method
4. Confirm Order</td><td>Payment: Card</td><td>Order placed successfully</td><td>As expected</td><td>Pass</td></tr>
<tr><td colspan="5">Post Condition: Order is recorded and confirmation is sent to user.</td></tr>
</table>

Test Case 5:

Project Name: SanckSnap	Test Designed by: Nahid Hasan Nobil
Test Case ID: SS_TC_05	Test Designed date: 16/06/25
Test Priority: Medium	Test Executed by: Talha Hossain Sifat
Module Name: Admin Dashboard	Test Execution date: 19/06/25
Test Title: Add New Restaurant	
Description: Test the ability of admin to add a new restaurant profile to the system	
Precondition: Admin must be logged in	

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Go to "Add Restaurant" Page 2. restaurant name 3. Business email 4. Phone 5. website url 6. opening hour 7.close hour 8.opening days 9. select category 10. Restaurant Address 11. Click Save	restaurant name: Biryani Palace Businesss email: biriyani123@gmail.com phone: 01700000000 website url: www.briryani.com opening hour: 6am closing hour: 10pm opening days: sat- wed category: deshi address: Dhanmondi	Restaurant is added and listed under available vendors	As expected	Pass
Post Condition: Restaurant is stored in the database and visible on user-facing browse page.				

Test Case 6:

Project Name: SanckSnap	Test Designed by: Nahid Hasan Nobil
Test Case ID: SS_TC_06	Test Designed date: 16/06/25
Test Priority: Medium	Test Executed by: Talha Hossain Sifat
Module Name: Menu Management	Test Execution date: 19/06/25
Test Title: Admin adds new food item	
Description: Add a new dish from admin dashboard	

Precondition: Admin must be logged in

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Go to Add Menu 2. Name 3. price 4.description 5. Click Save	Dish: coke Description: soda drink with sugar Price: 250	Item added to menu	As expected	Pass

Post Condition: Item appears in user-facing menu.

Test Case 7:

Project Name: SanckSnap	Test Designed by: Nahid Hasan Nobil
Test Case ID: SS_TC_07	Test Designed date: 16/06/25
Test Priority: Medium	Test Executed by: Talha Hossain Sifat
Module Name: Order Management	Test Execution date: 19/06/25
Test Title: Delete an Order	
Description: Admin deletes an unprocessed order from the system	

Precondition: Admin must be logged in and the order must be in a deletable state.

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Go to “Manage Orders” 2. Select a pending order 3. Click Delete	Order ID: ORD00987	Order is removed from database and order list is updated	As expected	Pass

Post Condition: Order no longer appears in order list and is deleted from backend records.

Test Case 8:

Project Name: SanckSnap		Test Designed by: Nahid Hasan Nobil		
Test Case ID: SS_TC_08		Test Designed date: 16/06/25		
Test Priority: Medium		Test Executed by: Talha Hossain Sifat		
Module Name: Admin Panel		Test Execution date: 19/06/25		
Test Title: Admin login				
Description: Test login with admin credentials				
Precondition: Admin account must exist				
Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Enter Admin username 2. Enter Password 3. Click Login	username: admin Password: Admin@123	Redirected to admin dashboard	As expected	Pass
Post Condition: Admin session initiated and dashboard accessible.				

8. ITEM PASS/FAIL CRITERIA

The testing team for the SnackSnap project will consist of the following roles:

- Unit Test Manager
- Test Analyst
- Project Sponsor
- Developer

The team is responsible for performing the following tasks:

- Ensuring that the codebase is free from critical errors and compiles within the expected timeframe across all modules.
- A test will only be considered successful if 100% of test cases pass as per the defined requirements.
- If any test case fails or if the pass rate drops below 100%, the corresponding bugs or issues will be immediately logged, prioritized, and resolved before the next test cycle.
- All test executions will be documented thoroughly, with clear records of test inputs, expected outputs, actual results, and final status for future traceability.

For this round of testing, we implemented 8 test cases covering the core functionalities of the SnackSnap platform. During the initial run, 6 out of 8 test cases (75%) passed successfully, while 2 failed due to issues related to database write operations and permission validation in the admin panel.

After identifying and resolving the issues, the failed test cases were retested. As a result, all test cases passed successfully in the final test cycle, indicating that the system is stable and functioning as expected.

9. TEST DELIVERABLES

The following deliverables will be produced for the SnackSnap system testing:

- **Test Specification Document:**
  This document will outline all testable scenarios for SnackSnap, including the purpose, scope, and coverage of each test case, particularly focusing on core functionalities such as user login, ordering, cart operations, and admin management.
- **Test Strategy:**
  A high-level overview of the testing approach for SnackSnap. It will define the testing methodology, levels of testing (unit, integration, system, UAT), test environments, roles, tools (e.g., Selenium IDE), and schedules to ensure systematic validation.
- **Test Scenario:**
  A set of detailed test cases that verify both functional and non-functional aspects of the SnackSnap system. This includes customer ordering flow, cart updates, checkout process, admin operations, and real-time Ajax-based interactions.
- **Test Data:**
  Input values and data sets used during testing, such as user credentials, restaurant names, dish details, and payment options. These data points are essential for validating correct behavior under various scenarios and edge cases.
- **Defect Reports:**
  Bug and issue tracking logs created during the testing process. Each defect report will include the bug description, severity, affected modules, steps to reproduce, and final resolution status.
- **Test Design:**
  A structured framework for designing and writing test cases, ensuring thorough coverage of all system modules (frontend and backend). This design process will support both manual and automated testing workflows.
- **Test Status Report:**
  A progress report that summarizes ongoing test case executions, pass/fail ratios, resolved issues, and pending tasks. This report will help stakeholders monitor the readiness and stability of the system.
- **Summary of All Reports:**
  A consolidated summary capturing the overall testing efforts, including final outcomes of test cases, critical insights, and a readiness verdict for system deployment. This will serve as a final checkpoint before product release.

10. STAFFING AND TRAINING NEEDS

To ensure the success of the SnackSnap online snack ordering system, it is essential to assemble a skilled and well-prepared testing team. Documented certifications and a commitment to continuous learning will play a vital role in maintaining quality throughout the project.

- Encourage team members to undergo training in relevant technologies such as JavaScript, PHP, MySQL, AJAX, and automated testing tools like Selenium IDE.
- Leverage interactive documentation platforms (e.g., internal wikis or shared workbooks) to maintain up-to-date guidance on evolving system features, bug resolution protocols, and test case standards.
- Utilize online resources including video tutorials, live sessions, and recorded webinars for scalable and accessible training delivery.
- Ensure the training provided also builds industry-relevant skills, allowing testers to apply their knowledge beyond this project and improve their professional development.

For SnackSnap, a team of five dedicated testers will be required. The team should include members with experience across multiple levels of software testing, including unit testing, integration testing, system testing, and UAT. These individuals must possess a deep understanding of SnackSnap's functional modules, such as user authentication, order processing, cart management, admin operations, and payment integration.

Additionally, system-specific training will be conducted to ensure the team is proficient in:

- Testing real-time features using Ajax
- Managing and verifying order lifecycles
- Testing mobile responsiveness and UI across devices
- Validating the admin dashboard functionalities

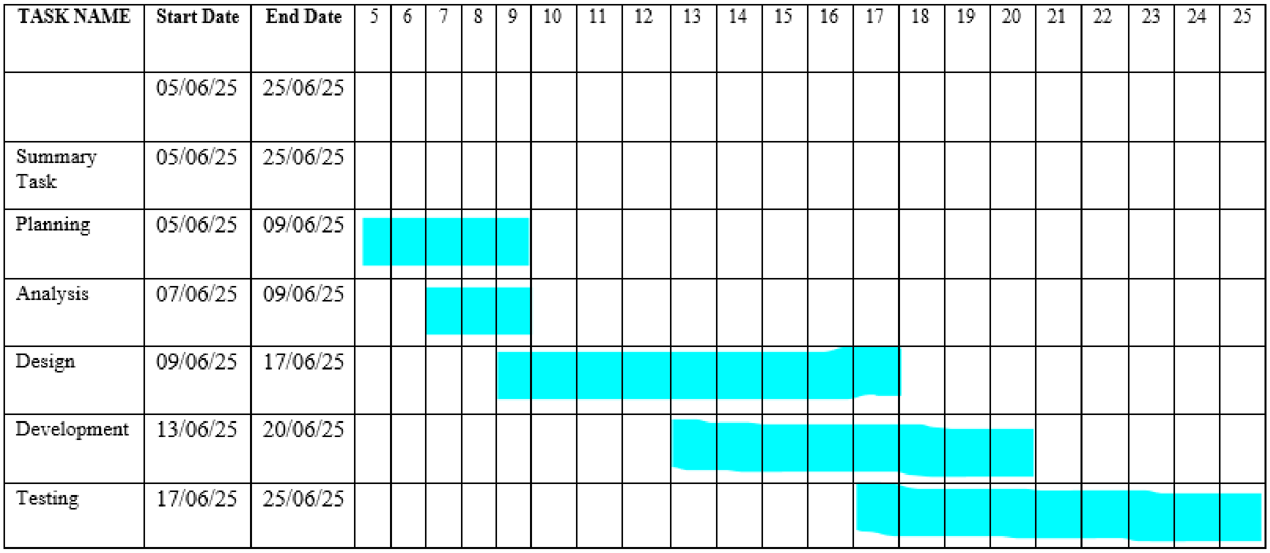
This structured staffing and training strategy will ensure a capable and confident QA team that can execute the testing process effectively, within the project timeline and budget.

11. RESPONSIBILITIES

- Project Team Leader: Responsible for verifying all test plans and ensuring they align with project goals.
- Test Lead: Responsible for writing the test plan, outlining the approach and scope of testing activities.
- Test Managers: Responsible for writing detailed test cases based on the test plan, covering all critical functionalities of the system.
- Project Team: All members will participate in the system review, providing feedback and ensuring that all aspects of the system are tested.
- Developer: Responsible for fixing any bugs identified during testing. The tester’s role is to detect bugs, not resolve them.
- Administration: Responsible for verifying the test results, ensuring they are accurate and meet the expected outcomes.

Name	Role	Responsibilities
Nahid Hasan Nobil	Project Manager	1. Requirement analysis and make plans for designing the project. 2. The whole Designing of the Project. 3. Execute all the test cases and report defects. 4. Control the whole project as designed.
Talha Hossain Sifat	Quality Analyst	1. Creation of test plans, test forms, test cases and test information. 2. Carry out testing as per the characterized methods. 3. Prepare all reports related to program testing carried out.
Muntasir Maruf	Developer	Researching, designing, ementing, and managing software rams. Writing and implementing ient methods to meet system irements. Deploying software tools, esses, and metrics.

11. TESTING SCHEDULE



TASK NAME	Start Date	End Date	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
	05/06/25	25/06/25																					
Summary Task	05/06/25	25/06/25																					
Planning	05/06/25	09/06/25	🟦	🟦	🟦	🟦	🟦																
Analysis	07/06/25	09/06/25			🟦	🟦	🟦																
Design	09/06/25	17/06/25					🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦								
Development	13/06/25	20/06/25									🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦					
Testing	17/06/25	25/06/25													🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦

12. PLANNING RISKS AND CONTINGENCIES

Limited Development Resources

- Risk: The SnackSnap development team may face resource constraints if one or more developers become unavailable, which could delay implementation of core features or fixing of critical bugs.
- Contingency: If development capacity becomes a bottleneck, the project timeline will be revised accordingly. High-priority modules—such as user login, order placement, and payment integration—will be prioritized for completion, while non-critical features (e.g., user reviews, wishlist) may be deferred to future updates.

Client/Restaurant Admin Training Delays

- Risk: Restaurant owners and admin users might face difficulties adopting the SnackSnap admin dashboard due to unfamiliarity with digital systems or insufficient training, which could affect system adoption and data entry accuracy.
- Contingency: Additional training sessions will be organized in both virtual and in-person formats. A phased onboarding strategy will be used, allowing restaurants to manually track orders alongside digital operations during the transition period.

User Acceptance Testing (UAT) Delays

- Risk: Customer volunteers, restaurant admins, or QA stakeholders might not be available during the UAT window, which may lead to delays in the testing schedule and feedback collection.
- Contingency**:** If UAT testers are unavailable, backup users or internal team members will be enlisted to simulate real-world usage. If no alternatives are available, the UAT phase will be rescheduled without compromising test coverage, ensuring that all planned scenarios are executed thoroughly.

13. APROVALS

Project Sponsor	AIUB
Development Management	Muntasir Maruf
EDI Project Manager	Nahid Hasan Nobil
RS Test Manager	Dale Tester
RS Development Team Manager	Talha Hossain Sifat
Reassigned Sales	Cathy Stales
Order Entry EDI Team Manager	Julie Order