Test Plan for Food Delivery Service Management System

Authors:

Sujan Jain (22CS10075)
Ashish Kumar (22CS30014)
Shshank Mittal (22CS30051)

Instructors:

- Prof. Sourangshu Bhattacharya
- Prof. Debasis Samanta

Indian Institute Of Technology Kharagpur

Date: 9th April 2024

1. **REFERENCES**:

- SRS Document
- Use-Case Diagram
- Class Diagram

2. **INTRODUCTION**:

The primary objective of this test plan is to ensure the robustness and reliability of the Food Delivery Service Management System (FDSMS). This plan outlines the testing procedures to be followed to validate the system's functionality, covering various user interactions and system components.

3. **TEST ITEMS**:

Home Page(To be tested using frontend)

Available Features

- **Buttons for Different Functions**
  - Verify the presence of three buttons for sign-up, login, and dashboard access.
  - Test redirection to the correct pages based on user actions.
  - Ensure the header symbol redirects users to the homepage.

Activities

- **Redirection**
  - Test redirection functionality from buttons without server or debug errors.

Test Plan for Sign-Up Process

Redirection to the Right Pages

- **Buttons**
  - Verify the presence of buttons for customers, restaurants, and delivery agents.
  - Test redirection to respective sign-up pages based on user selection.
- **Redirection**
  - Confirm that pages display appropriate headings over input forms.

Working Input Fields

Required Input Fields

- Test if required fields gives error when clicked submit but left empty.

Redirection to Login Page After Signup

- Confirm redirection to the login page upon successful sign-up.

Test Plan for Login Process

Appropriate Message When Wrong Credentials are Given

- Test display of correct error message for wrong login credentials.
- Verify behavior of input fields regarding login query creation.

Redirection to Correct Dashboard Based on Login Credentials

- Confirm redirection to the correct dashboard with a welcome message displaying user name and type.

Logout Process

- Test removal of session details upon logout.
- Verify redirection to the login page and prevention of unauthorized dashboard access.

Test Plan for Customer Features

Personal Details

- **Button Redirection**
    i. Verify that clicking on the personal details button redirects the user to the page displaying their information.
- **Details Display**
    i. Ensure that personal details are displayed in a clear and organized list format.

Dashboard Button Functionality

- Test each button on the dashboard to ensure they redirect users to the correct pages with the expected functionality.
- Verify that redirections align with the labels on the buttons.

Dashboard Redirection

- Verify that clicking on the dashboard button in the top bar redirects the user to the dashboard page, displaying the appropriate buttons.

Order Now

- **Button Action**
    i. Confirm that clicking on the "Order Now" button displays a list of restaurants with essential details and an option to view menus and place orders.
- **Complete Restaurant List**
    i. Ensure that clicking on a restaurant from the list displays the complete menu with selectors for item quantities.
- **Order Creation**
    i. Verify that users can create orders from the menu by selecting items and quantities.
    ii. Confirm that clicking the order button redirects users to a page showing order details and options to apply promotional offers.
- **Apply Promotional Offer**
    i. Test functionality to apply promotional offers to the order.
    ii. Verify that only one offer can be applied at a time and that the order total reflects the applied discount.

        iii. Confirm removal of the applied offer from the list after placing the order.
- **Correct Amount Display**
    1. Ensure that the total amount displayed for the order is accurate and includes any applied promotional offer discounts.

Recommended Restaurants List
- Verify that the recommended restaurants list displays only those restaurants recommended by the admin.
- Confirm that the features and functionality of this list match those of the complete restaurant list.

Present Orders
- **List of Orders**
    1. Test functionality to display all current running orders.
    2. Verify that clicking on order details redirects users to a page showing detailed information about the order.
- **Order Details**
    1. Confirm that order details include items, restaurant name, and order timeline/status.
    2. Ensure the timeline accurately reflects the order's progress, including preparation, out for delivery, and delivered stages.
- **Order Status**
    1. Verify that the order status/timeline provides approximate times of updates and relevant information about food preparation and delivery.
- **Feedback**
    1. Test functionality to provide ratings and feedback to the restaurant and delivery agent.
    2. Verify that users can submit ratings and feedback only once.

Past Order List
- **Details of Orders**
    1. Test functionality to display details of past orders, including restaurant name, delivery agent name, date of placing, total bill, and an option to view complete order details.
- **Elements in the List**
    1. Confirm that the list excludes present orders currently in progress.
    2. Verify that delivered or rejected orders are added to the list accurately.

Test Plan for Restaurant Features

Personal Details
- **Button Redirection**
    - Verify that clicking on the personal details button redirects the restaurant to the page displaying their information.
- **Details Display**
    - Ensure that all essential details of the restaurant are fetched from the database and displayed in a clear list format.

Dashboard Button Functionality

- Test each button on the dashboard to ensure they redirect restaurants to the correct pages with the expected functionality.
- Verify that redirections align with the labels on the buttons.

Dashboard Redirection

- Verify that clicking on the dashboard button in the top bar redirects the restaurant to the dashboard page, displaying the appropriate buttons.

Create Menu

- **Add Food Item**
  - Test functionality to add new food items to the menu.
  - Verify that clicking the add food item button leads to a page with input fields for item name, price, and that data is correctly inserted into the database.
- **Display Menu**
  - Ensure that the menu is displayed below the add food item button in a tabular form.
  - Confirm that the menu is vertically scalable as new food items are added, and that data is fetched from the database using the restaurant ID.
- **Delete Food Item**
  - Test functionality to delete food items from the menu.
  - Verify that clicking the delete button removes the food item from the database and updates the menu accordingly.

Pending Orders List

- **Accept/Reject Orders**
  - Test functionality to accept or reject orders from customers.
  - Verify that pending orders are displayed in a tabular form, and clicking accept successfully accepts the order while clicking remove removes the order from the list.
- **Estimated Time**
  - Test functionality to provide an estimated preparation time.
  - Confirm that the estimated time input field functions correctly.
- **Generate Delivery Request**
  - Test functionality to generate delivery requests for accepted orders.
  - Verify that accepted orders become visible for delivery agents to accept upon generating a delivery request.
- **Mark Out for Delivery**
  - Test functionality to mark orders as out for delivery.
  - Verify that clicking the "Out for Delivery" button marks the order accordingly and is visible to the customer.

Past Orders List

- Test functionality to display past orders served by the restaurant to customers.
- Verify that past orders are shown in a tabular form order details button.

Test Plan for Delivery Agent Features

Personal Details

- **Button Redirection**

        - Verify that clicking on the personal details button redirects the delivery agent to the page displaying their personal information.
    - **Details Display**
        - Ensure that all essential details of the delivery agent are fetched from the database.

Mark Location

- **Current Areas List**
    - Test functionality to view and select from a list of available areas.
    - Verify that the list of areas is populated correctly and allows the delivery agent to choose or mark their current location.
- **Update Location in Database**
    - Test functionality to update the current location of the delivery agent in the database.
    - Verify that clicking the update location button successfully pushes the marked location to the database.

Delivery Request

- **See Delivery Requests**
    - Test functionality to view orders in the delivery agent's area that haven’t been accepted yet.
    - Verify that orders are displayed in a tabular form on the webpage.
- **Accept Request**
    - Test functionality to accept delivery requests.
    - Verify that clicking the accept button marks the order as accepted, removes it from the delivery request page, and assigns it to the delivery agent.

Current Order Details

- **Display**
    - Test functionality to display details of the current order assigned to the delivery agent.
    - Verify that all relevant order details are visible.
- **Estimated Time for Pickup**
    - Test functionality to provide an estimated time for pickup visible to the customer.
    - Confirm that the estimated pickup time input field functions correctly.
- **Estimated Time for Delivery**
    - Test functionality to provide an estimated time for delivery visible to the customer.
    - Confirm that the estimated delivery time input field functions correctly.
- **Mark as Delivered**
    - Test functionality to mark the order as delivered.
    - Verify that clicking the "Delivered" button updates the status of the order to delivered and removes it from the delivery agent's screen.
- **Rate Customer**
    - Test functionality to rate the customer.
    - Verify that the delivery agent can rate the customer, and upon submission, the rating data is fetched from the database and updated correctly.

Test Plan for Admin Features

List of Customers

- **Button Redirection**
  - Verify that clicking on the dashboard button redirects the admin to a page displaying the list of all customers in the database.
- **Table of Details**
  - Test that the list shows basic customer details with buttons to give an offer and delete the user.
  - Ensure that customer details align with the information provided during signup, and ratings are correctly displayed.
- **Give an Offer**
  - Test functionality to give offers to customers.
  - Verify that selecting an offer from the list successfully adds it to the customer's offers in the database.
- **Delete User**
  - Test the ability to delete a customer from the list.
  - Ensure that clicking the delete button removes the user's data from the database and refreshes the page with the updated list.

List of Restaurants

- **Button Redirection**
  - Verify that clicking on the dashboard button redirects the admin to a page displaying the list of all restaurants in the database.
- **Table of Details**
  - Test that the list shows basic restaurant details with buttons to recommend, open menu, and delete the restaurant.
  - Ensure that restaurant details align with the information provided during signup, and ratings are correctly displayed.
- **Open Menu**
  - Test functionality to view the menu of a restaurant.
  - Verify that clicking the open menu button displays the restaurant's menu in a tabular form.
- **Recommend**
  - Test functionality to recommend or remove recommendation status from a restaurant.
  - Verify that clicking the recommend button changes the recommendation status accordingly.
- **Delete User**
  - Test the ability to delete a restaurant from the list.
  - Ensure that clicking the delete button removes the restaurant's data from the database and refreshes the page with the updated list.

List of Delivery Agents

- **Button Redirection**
  - Verify that clicking on the dashboard button redirects the admin to a page displaying the list of all delivery agents in the database.
- **Table of Details**

        - Test that the list shows basic delivery agent details with a button to delete the user.
        - Ensure that delivery agent details align with the information provided during signup, and ratings are correctly displayed.
    - **Delete User**
        - Test the ability to delete a delivery agent from the list.
        - Ensure that clicking the delete button removes the delivery agent's data from the database and refreshes the page with the updated list.

Change of Recommendation of Restaurants and Food Items

- Test functionality to change the recommendation status of restaurants and food items.
- Verify that clicking the change recommendation button updates the recommendation status accordingly.

Create Offer

- **Offer List**
    - Test functionality to view the list of previously created offers.
    - Ensure that basic details of each offer are correctly displayed.
- **New Offer**
    - Test functionality to create a new offer.
    - Verify that input fields for offer details function correctly and adding a new offer updates the list.

Give Offers to Customers

- **Giving an Offer**
    - Test functionality to give offers to customers.
    - Verify that selecting an offer adds it to the customer's list of offers and redirects back to the all customers list.

4. SOFTWARE RISK ISSUES:

Data Security Risks: MongoDB, being a NoSQL database, might present security risks if not properly configured. Insecure database configurations, such as weak authentication or authorization settings, could lead to data breaches.

5. FEATURES TO BE TESTED:

The following is a list of the areas to be focused on during testing of the application.

User Sign-Up Module

a. Three Options for Sign-Up
b. Redirection to the Correct Page
c. Proper Message Display on Sign-Up
d. Redirection to Login Page on Successful Sign-Up

User Login Module

a. Entering Information in Fields
b. Redirection to Correct Dashboard
c. Proper Message on Unsuccessful Login

Customer Features

a. View Personal Details
b. Access List of Restaurants
c. Access List of Recommended Restaurants
d. View Restaurant Menus
e. Create and Place Orders
f. View Current Running Order.
g. Access List of Past Orders
h. Provide Feedback
i. Check Order Details

Delivery Agent Features

a. View Personal Details
b. Mark Current Location
c. Access Available Delivery Requests
d. Accept and Update Order Status
e. Rate Customers

Restaurant Features

a. View Personal Details
b. Menu Management
c. Manage Pending Orders
d. Access List of Past Orders
e. View Nearby Delivery Agents
f. Send Delivery Requests

Admin Features

a. View List of Users
b. Recommend Restaurants and Food Items
c. Delete Users
d. Provide Promotional Offers

6. FEATURES NOT TO BE TESTED:

----

7. APPROACH:

Methodology

We adopt the Agile approach using the Scrum Development Methodology for this project. Our focus lies in fostering agility, continual planning, learning, improvement, team collaboration, evolutionary development, and timely completion. We prioritize individual and team interactions over processes and tools, encouraging flexible responses to change. The iterative and incremental nature of Scrum allows us to adapt to evolving requirements and deliver value incrementally

Test Types

1. As the project is relatively small and developed by the internal team, software testing will be performed by developers rather than a third-party. We aim for optimal testing based on risk assessment, prioritizing the most critical areas.
   a. Unit Testing
   We emphasize unit testing, focusing on smaller units of the software design. Each unit or group of interrelated units will be tested individually by the programmer using sample input and observing corresponding outputs. This ensures the correctness of individual components before integration.
   b. System Testing
   Our testing approach ensures that the software functions seamlessly across different operating systems, primarily Windows. We employ black box testing techniques, focusing on required input and output without delving into internal workings.
   c. Regression Testing
   With the addition of new features or components, regression testing becomes crucial. We ensure that the entire component works properly even after adding new components to the complete program. This helps maintain the integrity and stability of the software throughout its development lifecycle.
2. **Meetings**
   Regular test meetings will be conducted at the end of each day to review and identify any errors within the code written during that day's development cycle. These meetings provide an opportunity for developers to collaborate, discuss testing strategies, and address any issues or concerns promptly. Additional meetings can be scheduled as required for handling emergency situations or critical issues that may arise during development. Effective communication and collaboration among team members ensure efficient problem-solving and timely delivery of high-quality software.

8. ITEM PASS/FAIL CRITERIA:

Completion criteria based on successful data comparison and user satisfaction.

9. ENVIRONMENTAL NEEDS:

Hardware Requirements
There are no special hardware requirements for running the application. A standard computer with internet connectivity is sufficient.
Test Data
Test data will be provided to the software through demo orders, user creations, etc. This data will simulate real-world scenarios and interactions with the software.
General Requirements
Apart from a working email ID and internet connection, no specific requirements are needed while testing the software. Users should be able to access the application using any standard web browser.
Data Ranges
While testing, it's preferred to use unique data to avoid errors caused by data

repetition. Additionally, keeping the size of images small will ensure faster image uploading and retrieval.

Testing Strategy

Each part of a multipart feature should be thoroughly tested at least once by creating dummy users, placing dummy orders, etc. This ensures comprehensive coverage and validation of all functionalities.

Power Requirements

For local server installation, a working computer with power is required. However, for clients accessing the application, power requirements will depend on the device being used. Hosting platforms like Heroku will manage the server's power requirements.

Supporting Software

- MongoDB
- requests
- Flask, render_template, url_for, request, redirect, session from Flask
- wraps from functools
- HTTPError from requests.exceptions

System Usage during Testing

There are no restrictions on the use of the system during testing. The web application operates on a session-based model, where each user session is separate and managed simultaneously by the server via MongoDB. Therefore, testing activities will not interfere with other users' sessions.

10. STAFFING AND TRAINING NEEDS:

No specific training is required for any test tool is required.