Test Plan Document

Foodigo

Group 5

Yashil Depani AU1841005
Ridham Shah AU1841007
Rushil Patel AU1841008
Harshil Mehta AU1841010
Raj Mehta AU1841018
Vidish Joshi AU1841019
Manav Patel AU1841037
Harsh Agola AU1841106
Sanket Shah AU1841111

TABLE OF CONTENTS

1. Objectives and Tasks

1.1 Objectives

- To assess the working of various components of the project.
- Getting a fair idea of the improvements needed for the components which do not satisfy the testing criteria.
- To get a clear picture of the actual working implementation of the project.

1.2 Tasks

- To devise a testing strategy for the project.
- To test all the components and features.

2. Testing Strategy

2.1 Test process

3.1.1 Requirements Understanding

- Taking proper requirements specifications from the stakeholders
- Understanding the requirements

3.1.2 Preparing & Reviewing Test cases And Test Matrix

- Formulating test cases by exploring different scenarios.
- Making a test matrix which maps test cases to respective requirements.
- The test matrix will make sure to cover all the requirements.
- Conducting peer reviews for test cases and matrices.
- Suggestions given by the reviewers will be included in the further improvements made by the team.

3.1.3 Deployment

- Report will be delivered after the final testing with no further bugs is done.
- The application is deployed in the green environment and is further tested for its performance and functionality.
- After obtaining successful test results, it becomes suitable for new production.

3. Testing Methods

We are using Scrum methodology in Agile development. In scrum, development and testing happen in parallel and therefore the test comes first in agile development. The acceptance criteria will help to drivethe definition of done. If a user module passes the acceptance criteria, it is considered to be done. Therefore, a test will verify that you have completed a user story.
Methods :

3.1 Unit testing

- It is a first level of software testing in which individual units of a software are tested. Here, a unit refers to an individual program, function or procedures.
- It is performed by the software developers themselves or could also be done by individual testers.
- It is conducted by performing **white box testing method**. It identifies the security holes, poorly structured paths in the coding processes, flow of specific inputs and outputs generated and testing of each unit on an individual basis.
- All the features starting from login of the users of the system and the functionalities provided to them will be broken into units and testing individually.
- Example: In our system, the function which computes the total amount for anorder should display the correct value of the amount in the payment gateway.

3.2 GUI Testing

- It focuses on testing of Graphical User Interface which is created in order to interact with the users. This testing focuses on design of the screen and ease of interaction with graphical elements on the screen.
- A peer who is not a part of the development team conducts the test.
- We will keep in mind the look and responsiveness of the screen and at the same time one will have to think like an end user since it is theuser interface of the software which makes the user believe about the utility of that application.
- Example: The testing will check all the pages, menus, buttons, dialog boxes,icons and windows.

4. Features to be tested

4.1 Login

- Create account option for customer and restaurant owner
- Enter details for the account
- Change password option for both
- View account details option
- Edit details option
- Sign-in option if account has already been created

4.2 Customer Order food

- Select order food option for customer
- Option to browse restaurants
- Choosing a restaurant from the listed options
- Selecting items from the menu
- Adding the selected items to the cart
- Updating the items in the cart
- Option to select payment method
- Receive live order updates
- Place order
- View order history

4.3 Restaurant Owner

- Register for new restaurants
- Enter details for the restaurant
- Create menu for the restaurant
- Enter items and their prices in the menu
- Update items of the menu
- View incoming orders to the restaurant
- View past orders of the restaurant
- Remove registration of the restaurant

4.4 Delivery Agent

- Sign-in to the account
- View account details
- Edit account details
- Receive order from the restaurant
- Collect payment for cash-on-delivery type order

5. Functional Test Review - Findings

Testing Type	Test Cases	Input Data	Output Data	Test case success
Unit Testing	Sign up and login (Restaurant owner, customer, delivery agent)	User details	Successful sign up (details entered in db) and login (details fetched from db)	pass
Unit Testing	Browse restaurants	Restaurant name	Nearby restaurant name with icon displayed	pass
Unit Testing	Edit account details (Restaurant owner, customer, delivery agent)	Details to update	New details successfully saved	pass
Unit Testing	Provide feedback	Ratings and reviews	Input is successfully saved to db and gets displayed on the feedback page of the restaurant	pass
Unit Testing	View order history (Restaurant owner, customer, delivery agent)	Click on order history page	Display order history page	pass
Smoke Testing	Add items to cart (Customer)	Item name	Cart filled with added items	pass
GUI Testing	Display the ‘About’ section	Click on website icon	‘About’ section displayed	pass
GUI Testing	Open home page upon login for (Restaurant, customer, delivery agent)	Log in to the account	Display respective home page	pass
Integration Testing	Check integration of modules	Customer places an order	Restaurant receives the order	pass
System Testing	Check working of whole system	Place an order	Receive an order (Customer side)	pass

6. Unit Test Cases

- Input Data Validation

Test case	Test case description (Instances)	Checklist
1. Mandatory Fields testing	All fields are mandatory while creating an account.	✓
2. Ensure error notification	An email field should pop up an error when '@' is not present.	✓
3. Null value testing	The value of 'amount' should not be null.	✓
4. Garbage Value testing	Values like null or negative should be checked.	x
5. Unique Field Values testing	Values for a username and password should not be the same.	x

- System Interfaces

Test case	Test case description (Instances)	Checklist
1. Check if all fields/parameters on an interface are exercised	All the buttons and functionalities present on an interface should work	✓

properly.	correctly.	
2. All data fields need to work properly as per the validation list.	Data field for an ‘amount’ should display the correct value by summing the prices of ordered items.	✓

- Usability

Test case	Test case description	Checklist
1. Check if the layout is consistent with the design criteria.	Layout of the pages should be done to facilitate and display all the details corresponding to the particular functionality in a clear manner.	✓
2. Check for the Fonts, Colors, Sizes, etc.	Appropriate font size and colors should be set which makes the site easy to read and attractive.	✓
3. Check for alignment.	The details on a page should be aligned by grouping similar content.	✓
4. Mandatory fields need to be highlighted with an asterisk symbol.	The fields required for signing up for an account should be highlighted with an asterisk when left empty.	✓

- Security

Test Case	Test case description	Checklist
1. Negative testing- Password is not visible	When a password is entered, it should be shown in the form of an asterisk.	✓
2. Check if the password is saved in clear or encrypted?	To check the form in which the password is stored in the database.	✓
3. Verify the application with valid userId and invalid userIds	Check the userId field for correctness while logging in to the system.	✓

4. Verify the application with valid password and various invalid passwords	Check the password field for correctness while logging in to the system.	✓