# *FOOD FRENZY*
# *Online food-ordering system*

## Test Plan Document

Prepared by:
Group 8
Yug Patel (AU1741017)
Yesha Shastri (AU1741035)
Ratnam Parikh (AU1741036)
Devshree Patel (AU1741075)
Param Raval (AU1741083)
Sajil Vohra (AU1741094)

# TABLE OF CONTENTS

# 1. Introduction

FoodFrenzy is an online food-ordering software which is made for customers to order food from their desired place and time at their ease. This software can be used by anyone who wants to order food. The software allows the restaurant owners to register their restaurants in the system so that customers can place orders from them. The system also possesses delivery agents to deliver orders from restaurants to the customers. Basic privileges for customers will be able to browse restaurants, place orders and give feedback while the restaurant owners will be able to register/remove their restaurants from the system, accept/reject orders and add or update items on the menu. Delivery agent is equipped with the functionality of changing the status of order.

This Software Test plan gives a summary of the project by describing the scope and objectives of the project, testing strategy of the components, set of items to be tested, features to be tested, types of testing performed, various requirements needed for the testing, roles and responsibilities of the team members and the risks associated with the plan.

# 2. Objectives and Tasks

## 2.1 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.

## 2.2 Tasks

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

# 3. Testing Strategy

## 3.1 Test process

### 3.1.1 Requirements Understanding

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

### 3.1.2 Preparing 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.

### 3.1.3 Reviewing Test cases and Matrix

- Conducting peer reviews for test cases and matrices.
- Suggestions given by the reviewers will be included in the further improvements made by the team.
- Review for the improved test cases and matrix will also be done.

### 3.1.4 Executing Test Cases

- Test cases will be performed for various scenarios
- Status(Pass/Fail) of execution for different test cases will be updated in the document.

### 3.1.5 Regression Testing and Retesting

- Regression testing will be performed in cases like: if there is some problem with the database or when the web app is not responding. Other than that retesting for the fixed bugs will be done, once it is resolved, it will be updated in the document.

### 3.1.6 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.

## 4. 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. User stories are created which will have various requirement and acceptance criteria so the testing should be done in order to fulfill those. The acceptance criteria will help to drive the definition of done. If a user story passes the acceptance criteria, it is considered to be done. Therefore a test will verify that you have completed a user story.
We begin by creating unit tests for the story.

### 4.1 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.
- Eg: In our system, the function which computes the total amount for an order should display the correct value of the amount in the payment gateway.

### 4.2 Integration testing

- This testing considers the testing of all the components of the system as a whole. It verifies the interactions of various modules of a system. The appropriateness of transitions from one module to another is tested.
- It is performed by the remaining team members excluding the peer who developed the particular module.
- The secondary level of testing is done by **black box testing method**. This integration testing is done by a bottom-up approach for our system. In integration testing, each module of the software is tested individually and then other modules are appended to the existing ones.
- Eg: The interfaces and functionalities for customer side, restaurant side and delivery agent side were built independently and were later integrated to form a part of the system.

### 4.3 System Testing

- It is the third level of software testing. It validates the complete and fully integrated software product. End-to end system specifications will be tested here.
- Testing is done independently by the other peers who were not a part of the development of the product.
- Validation of the system is done by checking that it meets the pre-defined requirements mentioned during the development phase of the product.
- Eg: The basic requirements of the system - placing an order, giving reviews, successful payment were tested after the integration of the complete system.

### 4.4 Regression Testing

- This testing is done to ensure that a recent change in the code or program has not adversely affected the existing features. It is basically a selection of already executed test cases which are re-executed to ensure existing functionalities work fine.
- It is performed by the peers of the development team who are also testers.
- Eg: When the payment for an order is unsuccessful, the items which are already in the cart should remain and the amount should not be deducted from the respective payment credentials provided.

### 4.5 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.
- The tester 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 the user interface of the software which makes the user believe about the utility of that application.
- Eg: The testing will check all the pages, menus, buttons, dialog boxes, icons and windows.

### 4.6 Acceptance Testing

- Acceptance Testing is a level of software testing where a system is tested for acceptability.The aim of this test is to check if the system satisfies the acceptance criteria mentioned by the users on user stories and whether the system is deliverable or not.
- It is the last phase of software testing so it is performed before making the system available for actual use.
- User Acceptance Testing - This is basically an end-user testing. It is done by picking up specific requirements of the user for testing.
- Eg: One of the customer requirements is to be able to browse restaurants for ordering food, this specification should be fulfilled while delivering the final system.

### 4.7 Smoke Testing

- Smoke testing ensures that a deployed build is stable or not by verifying that the important features of the system are working or not.
- It is done whenever the new functionalities of software are developed and integrated with existing builds.
- The testing is conducted by the development engineers.

- Eg: New registration button is added in the login window and the build is deployed with the new code. We perform smoke testing on a new build.

## 5. Features to be tested

### 5.1 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

### 5.2 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

### 5.3 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

### 5.4 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 orders

## 6. Quality Objectives

### 6.1 Primary Objectives

The primary objective for conducting testing is to ensure that all the requirements - functional and non functional have been met by the project. Additionally, it has to be checked whether the quality metric for each individual requirement has been satisfied or not. All the requirements and acceptance criteria mentioned by the use-case stories should be accomplished. The ultimate aim at the end of the project development cycle is to ensure that the user identifies all of the requirements mentioned initially have been provided with much quality and are up to the expectations. Furthermore, any updates made to the requirements, functions or the design of the project will be appropriately documented and tested to provide the deliverables of highest quality within the available time frame.

### 6.2 Secondary Objectives

The secondary set of objectives will be to recognise any errors/bugs associated with the working of the current system and communicate about the same to the team members. After the system has been examined completely and the issues identified, a proper methodology should be employed to resolve the same in the given time frame. In all, a complete analysis of the system will be done followed by fixing the identified bugs.

# 7. Entry and Exit Criteria

## 7.1 Entry criteria

- Software and Hardware platforms required for testing must be installed. Testing environments must function properly.
- Requirements understanding documentation as well as design documents should be ready in order to allow the testers to test the system properly.
- Test data scenarios must be well known to the testing team.
- Sound understanding of functionality and requirements is expected from all the group members.
- Proper review and proofreading of test cases and requirements is done beforehand.

## 7.2 Exit criteria

- Rudimentary levels of requirements are satisfied.
- No bugs/defects marked as medium or high priority are left to be resolved.
- The plan has been followed.

# 8. Resource and Environment Needs

## 8.1 Testing Tools

| Process | Tools |
|---|---|
| Test Case Creation | Microsoft Excel |
| Test Case Tracking | Microsoft Excel |
| Test Case Execution | Manual |
| Test Case Management | Microsoft Excel |
| Test Reporting | PDF |

### 8.2 Test Environment

- **Hardware -** Computer
- **Software -** Python editor, wamp / mamp / lamp (for database), Google Chrome / Mozilla Firefox / Safari
- **Language -** Python, HTML, CSS
- **Backend Framework -** Django
- **Tools -** Idle (Python), Sublime Text, wamp/mamp/lamp

## 9. Pass / Fail Criteria

This criteria is to determine whether the execution of a particular test functionality passes or fails. If the guidelines are followed appropriately by the user and no mistake is made while executing the test functionality then the test is considered ‘Pass’ else it is considered ‘Fail’. Normally, there are three situations while executing the test cases - normal, suspension, resumption.

### 9.1 Suspension Criteria

- These are the criteria to suspend all execution until a specific requirement of the functionality is met.
- It can occur in case of unavailability of external dependent systems during execution, when a defect is introduced which cannot allow further testing or a critical path deadline is missed so that client will not accept delivery.
- For example, the system will go to a suspended state when the user enters incorrect username, password or type.

### 9.2 Resumption Criteria

- These are the criteria to resume the suspended execution after the specific requirement of the functionality is met.
- It can occur when the dependencies become available, a solution is introduced for the caused defect or when the client agrees on extending the time for delivery.
- For instance, the system will ask for a correct username, password or type when the user enters an incorrect one.

### 9.3 Approval Criteria

- These are the criteria which when fulfilled will result in the approval of the test cases.
- When all the requirements of a particular test case is fulfilled, it is tested appropriately by the testers and the check for quality is done with a positive outcome then the approval criteria is said to be satisfied.
- For example, the system user has to enter correct login credentials which will then be approved by the system in order to access the respective functionalities.

## 10. Dependencies, Assumptions and Risks

### 10.1 Dependencies

When the tasks or the modules of a project are dependent on certain requirements then those requirements are called dependencies. Some of the dependencies are listed below :

- SQL servers will be required for the database.
- Database is required to store and fetch data for the functioning.
- Payment gateway is needed to fulfill the procedure of payment.
- Internet based servers are required for the working of the website.

### 10.2 Assumptions

- Every user will have a paytm account and balance, since currently payment is possible only via paytm.
- Proper internet connectivity will be available with users in order to accomplish an order.
- The restaurants will be in the periphery of 1-2 km from the location of the user.
- Restaurants will be open 24/7 for accepting the orders.
- All the items provided in the menu by the restaurant will be available at any time.

### 10.3 Risks

Risk identification and management are the main concerns for a software project. Risks are identified, classified and managed before the actual execution of the project. Categories of risk are :

| Risk Type | Reason for Risk | Mitigating action |
|---|---|---|
| Schedule Risk | Wrong estimation of time because some resources may not be tracked properly. | Gather all the requirements and track all the resources beforehand. |
| Operational Risk | Insufficient planning and availability of resources, no communication in the team. | Make sure all required resources are available and discuss in the team the individual operational roles. |
| Technical Risk | Advanced technology is not available or the integration of project modules is difficult. | Prepare alternatives for the existing needs; a system modular approach will help ease the process of integration. |
| Programmatic Risk | Some flaw is generated in the database, an incorrect logic is written in code. | After writing code, an overview of it should be done by different team peers in order to recognise any fallacy if it exists. |
| Budget Risk | Cost overruns due to expansion of project scope. | A buffer should be kept for the cost in case the requirement changes. |

## 11. Limitations

- Redundancy of passwords is not checked for whenever a new account is created.
- Payment mode is restricted only to paytm.
- Dynamic calculation of delivery fee is not done as per varying circumstances under which order is placed.
- The working mechanism of the delivery agent is independent from the restaurant as the restaurant will not be able to take any actions against the delivery agent.

- Process of assignment of a delivery agent for picking an order does not take any mechanism based on distance or traffic into consideration.

## 12. 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 |
| Smoke Testing | Add items to menu (Restaurant) | Item name | Menu filled with added items | pass |
| Regression Testing | Items in order cart to remain same when payment is unsuccessful | Wrong credentials | Display payment unsuccessful message and redirect to previous page | pass |
| Regression Testing | Check for routes having minimum traffic using | Restaurant location | Possible routes should be displayed | fail |

| | | | | |
|---|---|---|---|---|
| | geolocation | | based on traffic density | |
| 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 |
| GUI Testing | Direct to payment gateway | Click on pay button | Display the payment portal | 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 |
| Acceptance Testing | Check fulfilment of user requirements | | | pass |

## 13. Meeting details

| Meeting Number | Agenda | Duration | Date |
|---|---|---|---|
| 1 | Discuss and decide the Software Development Methodologies. | 50 mins | 24/01/20 |
| 2 | Discuss the basic requirements understanding. | 1 hour | 29/01/20 |
| 3 | Discuss the features of software | 1 hour | 03/02/20 |
| 4 | Database Design | 50 mins | 12/02/20 |
| 5 | Design of Use Case Diagram and web pages | 1.5 hours | 20/02/20 |
| 6 | Designing State and Activity | 1.5 hours | 04/03/20 |

| | | | |
|---|---|---|---|
| | diagram, User Analysis | | |
| 7 | Designing of Data Flow Diagram, Sequence Diagram and building test cases. | 2 hours | 13/03/20 |
| 8 | Building of sprint backlog and assigning work for each team member. | 45 mins | 20/03/20 |
| 9 | Sprint-1 review and discussion for sprint-2. | 1 hour | 28/03/20 |
| 10 | Sprint-2 review and discussion for sprint-3. | 1 hour | 05/04/20 |
| 11 | Sprint-3 review and testing of final software. | 3 hours | 20/04/20 |

## 14. 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. | ✓ |
| 3. Security testing across automated Interfaces. | The data integrity measures followed by paytm will be used on the user credentials. | ✓ |

- 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. | ✓ |

- Environment

| Test Case | Test case description | Checklist |
|---|---|---|
| 1. Testing with all the browsers | The web application is hosted online which works on any browser | ✓ |
| 2. Testing by enabling and disabling Java scripts | The web application works fine in presence/absence of JS. | ✓ |

- Search Criteria

| Test Case | Test case description | Checklist |
|---|---|---|
| 1. Verify that scroll bar is implemented | Scroll bar should be implemented to scroll up and down the page. | ✓ |
| 2. Verify that the alignment of the search results is proper. | The search result for the restaurant should display the restaurants in an aligned manner. | ✓ |

## 15. Recommendations

An analyst would suggest following recommendations for the system :

➔ Security measures should be employed into the system for safe payment transactions without theft
➔ Troubleshooting and updating of the system should be done to maintain the standards of the system.
➔ Latest versions of antivirus should be installed and updated frequently to detect the presence of any virus.
➔ Each customer, delivery agent and the restaurant owner should be identified uniquely with a unique id and password
➔ Google Maps API (Paid) should be used for fetching accurate geolocation and complete address of any user
➔ Geolocation and GPS hardware should be used to track customer and delivery agent
➔ Portability to mobile application
➔ Add more payment options such as PayPal, Google Pay, and net banking
➔ Improve summary statistics display for restaurant to see their reviews
➔ Functionality to remove items directly from cart