Test Plan Document

OpenCrowd - LettuceBuy

1. Introduction

LettuceBuy is a website designed to provide users an easy way to obtain groceries while sitting in the comforts of their homes while drivers that have extra free time can register to make more money on the side. Our aim for this project is to provide core functionalities for both client and driver side with a self-explanatory user interface. The current working product allows new users to register as either client or drivers safely while using a password encryption in the backend to ensure security. The clients are able to submit a list onto an active pool list that is visible to all drivers while drivers can pick a certain list to their liking based on the number of items they see in a particular list or simply by their own location preference. Note: The shopping lists are available to all drivers but a driver chooses what to pick. The client is allowed to delete a list at any time unless the list has been 'fetched' by a driver. At this point, the client is prompted to call the driver instead. Both the driver and client have the ability to finish a transaction which marks the end of the client-driver interaction. Once that button is pressed by the driver, he/she cannot fetch another list until the client confirms the fact that they have received the items requested.

2. Quality Control

Every time a commit is pushed, developer must update test suite to reflect current build. If functions are added new test cases should be added. Our git repository is linked with travis, every commit is tester with all test case automatically by travis. Our developers will debug, modify, update coded based on the exit status reported by Travis-IC. No commit should be pushed to master unless all tests are passed.

2.1 Test Plan Quality

We have selected only the most important functional and critical classes for formal unit testing. In the test environment we used, a unit for unit testing is a class. Here we illustrate the testing methodology by discussing only two classes.

Our main interaction occurs with the database which is located onto a Google Cloud server. Therefore, all of our test cases such as php script for register will be marked correct if and only if the input made by a user is the same as the contents inside the database. Our database contains three tables, one containing all client information, another for all driver information and lastly shopping lists which is being referred to as a pool of lists in this document. We have a set of flags being used for a variety of cases in different pages. For instance, the login page has a flag that will output "You are registered", if and only if a user is being redirected to it after registration. The same flag with a different value is used if a user enters invalid credentials during login phase. We will input false login

information and check the values of the flags to confirm that the connection between the server and database is working the way we intend it to.

The most intricate part of our website is the info inside of our database as we are constantly interacting with it therefore every one of our test will be checking the result inside a database to see if it matches the input. This is how we know our test plan is viable for our project.

2.2 Adequacy Criterion

The code convention that we have used thus far has been such that once a change in the database has been made successfully without error, the user is often redirected to a new page depending upon the situation. Instead of this redirection, we will echo out a statement that says “True” where it succeeds or “False” where it fails. For example, once the login script determines that the input credentials by the user were false, it will redirect the user back to the login page and instead will be seeing a message in red depending upon the situation.

A test is passed successfully if the exit code in Travis for all scripts is 0. Test scripts also act as drivers for codes. These scripts run our methods in a specific order and coverage will be discussed below. After an operation is done, the test scripts test the condition of both, the current state of the program as well as the modification in the database. If any one of these failed, the corresponding exit code will be sent to Travis which will indicate whether a test is passing or failing and also where. A command on the failed build is printed when a test case fails to help developer find out where the mistake resides.

3. Test Strategy

Here we will list the types of tests we will be performing in order to make sure that our project’s integration runs as smoothly as possible. The test cases have been selected using the functional approach. The goal is to test the different functional requirements, as specified in the requirements document.

We are using a continuous integration tool called Travic-IC for testing. Every time a piece of code no matter how trivial is pushed to github, Travis automatically executes the script that is written inside the travis.yml file. However, developer must manually adjust the test suit in side the testing folder to match current design.

We focus on testing individual block in the cycle from register to finishing a list. The testing cases go from testing individual or a small set of methods to using multiple methods to ensure success during integration. That way, when a test fails, a developer can trace the origin of the bug easily based on our approach and customized exit code as well as comments that travis prints out.

4. Test Cases

Here are all of the test cases performed on Travis using multiple different test cases. All of the test cases passed as expected. The main functions of our website were prioritized when it came to testing in order to ensure that it is as bug-free as possible.

Operation Performed	Purpose	steps	Condition tested	Actual result
Register driver	ensure drivers are able to register with correct input	fill in the required information and use driver register php to register	check driver table for existence and redirecting to login-page.php	test passed
Register Client	ensure clients are able to register with correct input	fill in the required information and use client register php to register	check client table for existence and redirecting to login-page.php	test passed
duplicated username register	ensure that user name are unique	register two users with the same username using the register pages	check/ensure no duplication happened and redirect to register page with warning	test passed
login client	ensure that clients with correct registered info are able to login	logging in using login php for clients with the correct credential	check for redirecting URL and go to corresponding page based on list state	test passed
login driver	ensure that driver with correct registered info are able to login	logging in using login php for driver with the correct credential	same as above	test passed
login with wrong password	only user with correct credential	logging in to account with wrong password	redirect to login page with warning	test passed
submitting a list	ensure list are upload when needed values are provided and users are at the right sate	fill in items, address and username and uploade the list using newlist.php	check that the list is added and is associated with client	test passed

submitting a list with no item	required items must be filled before submit	filling everything required accepts item field being empty	check that no list is added and go back to new list page with corresponding flag	test passed
fetching list	ensure drivers are able to fetch list from the pool	clear database register two new users, create a list, fetch the list. Check database and redirection link	check for status update in list table and drivers table	test passed
deleting a list	deleting a list and right changes happen user and list pool	clear the database register as new client, submitting a list and deleting the list	check for empty database after deletion	test passed
completin g	users are able to finish a list and create or accept new lists	after fetching list as described above driver complete the transaction and clients confirms	check for status update to driver clients and list as well as redirection link	test passed