ECE 6400 Integration Testing

Test case 1 for integration:

Category: Preparing an order

Description: Notifying restaurants about order

Pre-requisite: Customer paid for order

Test Step(s): 1) Fetch restaurant ID from database, 2) fetch order ID from database, 3) notify restaurant about order through messaging method, 4) wait for restaurant to confirm order, 5) if order is rejected, then pop up an error that no restaurant is busy, please try again later, refund, and redirect to main page, 6) if order is confirmed, then raise flag that order is confirmed by restaurant

Expected Result(s): 1) restaurants are notified about order, 2) order is either confirmed by restaurant or refunded to user if restaurant did not confirm

Actual Result: Tester's findings (could be visual or console output as flags or text)

PASS/FAIL: automated result of test (all tests should PASS by default)

Test case 2 for integration:

Category: Preparing an order

Description: Getting available drivers

Pre-requisite: Customer paid for order and restaurant is notified

Test Step(s): 1) call getAvailableDrivers algorithms method, 2) get value from the getAvailableDrivers method, 3) fetch Available Drivers from database, 4) compare with value from getAvailableDrivers, 5) if no driver is free, then pop up an error that no driver is free, please try again later, refund, and redirect to main page

Expected Result(s): 1) Only available drivers should be returned and verified through the algorithms' method, 2) if no drivers are free, notify user, refund, and re-direct to main page

Actual Result: Tester's findings (could be visual or console output as flags or text)

PASS/FAIL: automated result of test (all tests should PASS by default)

Test case 3 for integration:

Category: Delivering an order

Description: Getting free driver

Pre-requisite: Customer paid for order and there are free drivers

Test Step(s): 1) Get free drivers' IDs, 2) notify all drivers of the order, 3) wait for first driver to choose order, 4) get driver ID and location

Expected Result(s): 1) Driver chosen to deliver order

Actual Result: Tester's findings (could be visual or console output as flags or text)

PASS/FAIL: automated result of test (all tests should PASS by default)

Test case 4 for integration:

Category: Delivering an order

Description: Getting time estimate for order

Pre-requisite: Customer paid for order and driver is chosen

Test Step(s): 1) Call getTimeEstimate algorithms method, which return time until driver arrives at restaurant, 2) ensure time is reasonable (not negative and not more than X amount of time), 3) call getDeliveryTimeEstimate algorithms method to get total delivery time, 4) ensure time is reasonable (not negative and not more than X amount of time)

Expected Result(s): 1) time estimate is gathered and is reasonable for order

Actual Result: Tester's findings (could be visual or console output as flags or text)

PASS/FAIL: automated result of test (all tests should PASS by default)

Test case 5 for integration:

Category: Delivering an order

Description: Real-time mapping of order

Pre-requisite: Customer paid for order and time is estimated

Test Step(s): 1) fetch position of customer, driver, and restaurant through map methods, 2) Display position of customer, driver, and restaurant through map methods, 3) display driver moving through the map interface, 4) keep notifying customer of time (should keep decreasing)

Expected Result(s): 1) User can see real-time view of themselves, driver moving, and restaurant, 2) User can see time decreasing

Actual Result: Tester's findings (could be visual or console output as flags or text)

PASS/FAIL: automated result of test (all tests should PASS by default)

Test case 6 for integration:

Category: Order delivered

Description: Customer received order

Pre-requisite: Customer paid for order and real-time mapping of order delivery

Test Step(s): 1) driver arrives at customer location on the map, 2) customer is notified that their driver is there, 3) customer receives order and is redirected to main application page

Expected Result(s): 1) Order marked as done and user redirected to main page

Actual Result: Tester's findings (could be visual or console output as flags or text)

PASS/FAIL: automated result of test (all tests should PASS by default)

Test Case 7 for Integration:

Description:Placing an Order

Pre-Requisite: None

Test Steps:

1) Choose a desired restaurant
2) Select an appropriate location
3) Select desired food items
4) Complete order by providing payment information, or by confirming stored payment information from account (if logged in)

Expected Result:

1) Confirmation message is displayed with expected time of arrival
2) Account is updated with the order details (if logged in)

Actual Result:

Tester's findings

PASS/FAIL:

Automated result of test

Test Case 8 for integration:

Description: Checking order status

Pre-Requisite: Having placed an order prior to checking

Test Steps:

1) Navigate to order status feature
2) Provide an order number
3) Confirm order number entry

Expected Result:

1) If order exists, message is displayed with relevant order details, including expected delivery time
2) If order does not exist or has already been completed, separate message is displayed with this information

Actual Result:

Tester's findings

PASS/FAIL:

Automated result of test

Test Case 9 for integration:

Description: Logging in to user account

Pre-Requisite: Having created an account prior to log-in attempt

Test Steps:

1) Navigate to log-in screen
2) Provide necessary details: username and password
3) Confirm detail entry

Expected Result:

1) If account exists and log-in details are correct, log in to the proper account, allowing for order placement and order checking
2) If account exists but log-in details are incorrect, provide error message indicating that the log-in was unsuccessful.
3) If account does not exist, provide message to encourage account creation.

Actual Result:

Tester's findings

PASS/FAIL:

Automated result of test