System and Unit Test Report

March 2021

1 Sprint 1

User Story 1: As a customer, I want to be able to log in so that I may manage my account.
User Story 2: As a customer, I want to be able to view all the products I can request so I may request desired products.

Scenario A (Back-end): 1. Using "product/add-product" route, add following product with fields:

- itemName = 'bread cake'
- price = '10.00'
- businessID: 12345
- reqCheckoutDuration: 64
- returnOpt: true
- itemDescription = 'cake is bread though'

2. Using "product/all-products", verify that the fields are correct and retrieve mongodb productID
3. Using "product/:productID", and the productID from step two, verify once again that all fields are correct

Scenario B (Front-end):

1. Execute npm run dev; access website via Chrome; log-in using google API
2. Request the cake product from the request products page
3. Go to the database (mongodb) and verify that the request was made with all fields filled

- itemName = 'bread cake'
- price = '10.00'
- businessID: 12345
- reqCheckoutDuration: 64
- returnOpt: true
- itemDescription = 'cake is bread though'

4. Drop request to avoid duplicates

2 Sprint 2

User Story 1: As a customer, I want to be able to log in so that I may manage my account.
User Story 2: As a customer, I want to be able to view all the products I can request so I may request desired products.

Scenario A: 1. Execute npm run dev; access website via Chrome; log-in using below details

- UserName = "kazhumDev"
- password = "arbok115"

2. Use EditAccount page to change Customer Name from "Orangutan" to "Monkey"
3. Log out, verify that Customer Name has changed in MongoDB
4. Repeat Step one and two, and change Customer Name back to "Orangutan"
5. Repeat Step 3

Scenario B: 1. Execute npm run server 2. Using "product/add-product" route, add multiple fields according to the pattern, with x staring at 1 to 10 (do it manually or write a script):

- itemName = 'bread x'
- price = 'x+1'
- businessID: 1"x"2345
- reqCheckoutDuration: x*10
- returnOpt: true
- itemDescription = 'Item x'

2. Close server; execute npm run dev; 3. Verify that all 10 products appear on requestProducts page 4. Close server; drop products collection

3 Sprint 3

User Story 1: As a business, I want to be able to login and create products so customers can request them.
User Story 2: As a customer, I want to be able to see the list of products I requested.

Scenario A: 1. Execute npm run dev; access website via Chrome; log-in on business side using Google OAuth
2. Add 10 products via create products according to below fields:

- itemName = 'cake x'
- price = 'x+5'
- businessID: 5432'x'
- reqCheckoutDuration: x*5
- returnOpt: false
- itemDescription = 'Cake x'

3. Close server; execute npm run dev;
4. Verify that all 10 products appear on requestProducts page
5. Close server; drop products collection

Scenario B: 1. Execute npm run dev
2. Using "request/add-request" route, add 5 requests based on fields below:

- itemName = 'bread x'
- price = 'x+1'
- businessID: 1"x"2345
- date: new Date()
- returnOpt: true
- itemDescription = 'Item x'