CMPT 276 Group Project Iteration 2

Test accounts are provided, but you would need to sign up with your own phone number to see the actual SMS sent when a driver has accepted your order.

When checking out, you may fill required fields with test data and proceed using this credit card.

Card Number: 4242 4242 4242 4242
Expiry: 10/22
CVV: 222

Driver Partition: Please accept the location request when prompted.

If there are errors on the driver partition, please clear cache and browsing history and visit with HTTPS protocol.

Project Abstract

BC Liquor Stores are currently the largest in-person alcohol stores in British Columbia. However, with new technology coming out, the stores are missing out on a lot of revenue which could be made by creating a web application that delivers their products. Our application is designed to solves such problems:
1. Helping BC Liquor Store expand an online store to increase revenue
2. Bringing convenience to our users: Users can order their alcohol with a touch of a button
3. Benefiting our society for decreased DUI rate: Using our App may help reduce the likelihood of people driving to get refills

Customer

Client:
Our target customers are ages 19+ liquor lovers who enjoy drinking at home, sensitive to price, and prefer board product selections.
Driver:
The drivers who take the order and courier this alcohol to the customers' place.
Admin:
The store owners want to manger their products and orders online.

Competitive Analysis

Our competitors such as Uber Eats or DoorDash only provide restaurant liquor delivery service. Usually, these products are overpriced due to the profit earned by both liquor providers and restaurant owners. Our app helps customers reduce their expenses by directly ordering their favorite product from one of the largest liquor retailers in BC. Moreover, BC Liquor Stores' product is more varied than any restaurant in town. From world-renown brands to local popular brands, no matter wine, beer, spirit, coolers, or ciders, all purchases can be easily done with a touch of a button.

User Stories

Name and Actor:
Regular User
Title:
Search product and purchase products
Triggers:
As a regular user, I need to search first and then purchase products on the website.
Actions:
After opening the homepage, I input the name of the alcohol that I want to purchase and click on the search button. I can see the alcohol that I want to purchase on the list the website provided. I click on the alcohol the information of the alcohol popped up and I check the size and the flavor of the alcohol. Then after I make sure that the product is the one I want, I add a number of the products to my shopping cart by clicking the button below the product. After I add all the products that I need into my shopping cart, I click on the cart button at the top corner to double-check the number and product that I want to buy. Then I click on the checkout button below to

pay for my order. After clicking the button “Pay Now” window pop up to ask me to input my email address, my name, and my address to receive the delivery. I input all the information and click the Payment Info button. The website continues to ask me for the credit card and other information to finish the payment. I put in my card information and click pay now, the website checks the information, and a window pop up and says payment successful.
Thus, I finish payment and can wait at home for the delivery service.

Improvements:

1. Users can put items into their shopping cart even if they are not logged in, but users will redirect to the login page if they are going to check out for these items.
2. Provides purchase history will be saved for users to save the time for uses looking up these items they buy regularly.
3. A close button is provided to users after checking the order summary, in case a user forgets to add some of the items into the shopping cart.
4. On the address input window, if a user input wrong information into email address the box becomes red to let users know that this field is wrong and please check the information.
5. Users can use different addresses each time they purchase or use a saved address in the system.

Regular user story ToDoList:

- Users can comment on products and courier
- Users can check the status of their orders

Name and Actor:

drivers/courier

Title:

Get order and delivery to its address

Triggers:

As a driver, get one order and delivery to the customer’s address.

Actions:

When I start working, I log into my account, the system recognizes my account as a driver, so the system shows me a map and the list of order which are already paid by customers and waiting for drivers to delivery. I find one from the order list and hit accept the order, the system issue the job to me to deliver the alcohol to the customer. When the customer receives the alcohol that he ordered, he sends a code that provides by the system so the liquor store knows that I finish the order and I get paid for that.

Improvements:

1. A driver can use the previous order at the user panel to check the history to ensure he gets the right pay.

ToDoList:

- the main page can provide navigation for drivers
- Sort and recommend orders for drivers
- Tips systems
- Rate and comment system

Name and Actor:
administrator
Title:
Check all the user in the system
Triggers:
As an administrator, I need to know how many customers and drivers are using the system during
Actions:
After I log into my account, I can click the all users button at the user panel, all different types of users and their contact information will be shown to me.
ToDoList:

- To delete or change the information of a user.
- To know how many drivers are currently using the system

Test accounts

type	account	password
admin	2368870654	admin111
General user	7785982990	123456
driver	6047263371	test123

These accounts are provided, but you would need to sign up with your own phone number to see the actual SMS sent when a driver has accepted your order.

API used

- Google Maps API

  provide a map feature for drivers.

- BC Liquor Store

  Provide alcohol information to clients

- Twilio API

  Provide phone and messages features for login and passwords reset

- Stripe API

  Provide an online payment for clients

Test code

public/src/__tests__/index.test.js

```
import React from 'react';
import App from
'../pages/App.js';
import { shallow } from
'enzyme';



describe('test app', () => {
  test('test app', () => {
    const wrapper =
shallow(<App/>);
  });
});
```

routes/api/test.js

```
//
Simulate
sending
request
            function getResponse(url,method,data,fun)
            {
                var xmlhttp;
                   if (window.XMLHttpRequest)
                   {
                          xmlhttp=new XMLHttpRequest();
                   }
                   else
                   {
                          xmlhttp=new
            ActiveXObject("Microsoft.XMLHTTP");
                   }
                   xmlhttp.onreadystatechange=function()
                   {
```

```
             if (xmlhttp.readyState==4 &&
xmlhttp.status==200)
              {
                fun(xmlhttp.responseText);
         }
         else
         {
             fun("Error");
         }
       }
       xmlhttp.open(method,url,true);
       xmlhttp.send(data);
}


//Convert JSON string to object
function stringToJson(text)
{
    return JSON.parse(text);
}




//Product information test
function T_products()
{
    var data;
    var fun=function(info){data=info};
    getResponse("/products","POST",NULL,fun);
    console.log(data);
}


//Order test
function T_checkout(phone,products,name,address,latLng,)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.stripeToken=generateCode();
    data.phone=phone;
    data.products=products;
```

```
    data.name=name;
    data.address=address;
    data.latLng=latLng;
    getResponse("/checkout","POST",data,fun);
    console.log(res);


}


//Generate random order number
function generateCode() {
       return Math.floor(Math.random() *
Math.floor(1000000));
}


//Get order
function T_getOrders()
{
    var data;
    var fun=function(info){data=info};
    getResponse("/getOrders","POST",NULL,fun);
    console.log(data);
}


function T_getPrevious()
{
    var data;
    var fun=function(info){data=info};
    getResponse("/getPrevious","POST",NULL,fun);
    console.log(data);
}


function T_getOngoing()
{
    var data;
    var fun=function(info){data=info};
    getResponse("/getOngoing","POST",NULL,fun);
    console.log(data);
}
```

```
function T_acceptOrder(stripeToken,phone)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.stripeToken=stripeToken;
    data.phone=phone;
    getResponse("/acceptOrder","POST",data,fun);
    console.log(res);
}


function T_sms(msg,num)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.msg=msg;
    data.num=num;
    getResponse("/sms","POST",data,fun);
    console.log(res);
}


function T_allUsers()
{
    var res;
    var fun=function(info){res=info};
    getResponse("api/users/allUsers","POST",NULL,fun);
    console.log(res);
}


function T_forgotpw(phone)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.phone=phone;
    getResponse("api/users/forgotpw","POST",data,fun);
    console.log(res);
```

```
}


function T_forgotpw2(phone,code,password)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.phone=phone;
    data.code=code;
    data.password=password;
    getResponse("api/users/forgotpw2","POST",data,fun);
    console.log(res);
}


function T_login(phone,password)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.phone=phone;
    data.password=password;
    getResponse("api/users/login","POST",data,fun);
    console.log(res);
}


function T_getOngoing(phone)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.phone=phone;
    getResponse("api/users/getOngoing","POST",data,fun);
    console.log(res);
}


function T_getPrevious(phone)
{
    var res;
    var fun=function(info){res=info};
```

```
    var data=new Object();
    data.phone=phone;
    getResponse("api/users/getPrevious","POST",data,fun);
    console.log(res);
}


function T_register(name,phone,password,password2)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.phone=phone;
    data.name=name;
    data.password=password;
    data.password2=password2;
    getResponse("api/users/register","POST",data,fun);
    console.log(res);
}


function T_register2(code,phone)
{
    var res;
    var fun=function(info){res=info};
    var data=new Object();
    data.phone=phone;
    data.code=code;
    getResponse("api/users/register2","POST",data,fun);
    console.log(res);
}
```