CSE 445 598 Project 3

Group 5 Submission – Assignment 5 Question 1

Members: Nichole Courtney, Matt Beebe, Chengxu Liang
Contributions: Evenly Split (each member contributed 33.33% effort)

Question 1.1 Description of Service-Oriented Computing System

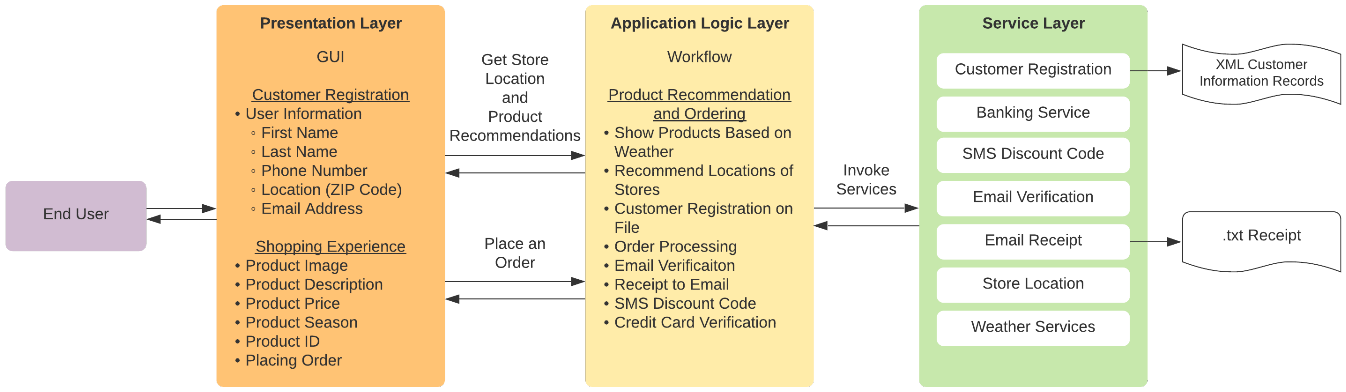
We have decided to create a store front for a camping store. A user will register with our system to create an identity. Our system will recommend products to match their location's weather. For example, if the user lives in a cold climate, our service will recommend products such as jackets that the user may want to purchase. The system will also recommend locations of our stores that the user can visit. The user can select a product for purchasing. Our system will utilize services that check the identity and transaction information provided by the customer (such as validating credit card information) before placing the order. Once placed, our system will email a receipt to the customer, confirming the purchase.

Question 1.2 Diagram of Overall System Design

The diagram of our camping store front system is displayed below. The presentation layer will utilize a GUI for the end user to interact with. The presentation layer has the customer registration and shopping experience components visualized. Two main functionalities are provided: recommending products and locations and placing an order for a product. Both functions gather info from the presentation layer and communicate it through the application logic layer to the service layer, invoking developed services. The results of processing done by the services and the application logic layer will be returned to the end user in the appropriate form (e.g., product recommendations will be visualized on the presentation layer after location and weather services are invoked; .txt receipt is emailed to the customer after order is successfully placed). The application logic layer's workflow diagram is also shown.

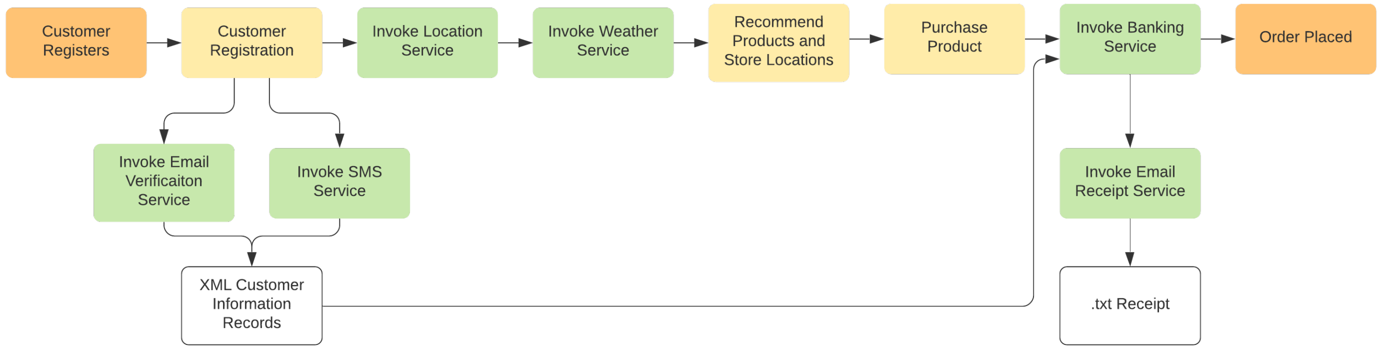
Question 1.2 Service-Oriented Computing System Diagram

Camping Store Front | Nichole Courtney, Matt Beebe, Chengxu Liang | Group 5



Question 1.2 Application Logic Layer Workflow Diagram

Camping Store Front | Nichole Courtney, Matt Beebe, Chengxu Liang | Group 5



Question 1.3 Proposed Service Directory

<table>
<tr><td colspan="5"><b>Service Directory:</b> The team plan to complete the following services. Changes can be made later in the final project (Project 5)</td></tr>
<tr><td colspan="5">This service directory will be deployed at an address in a later submission (Assignment 6)</td></tr>
<tr><td colspan="5">Team name/Number: Group 5 (Nichole Courtney, Matt Beebe, Chengxu Liang)</td></tr>
<tr><td>Provider Name:</td><td>Service name, with input and output types</td><td>TryIt Link*</td><td>Service Description</td><td>Planned resources need to implement the service</td></tr>
<tr><td>Nichole</td><td>SMS Discount Code for purchase
Input: phoneNo (string)
Output Type: void</td><td>TryIt Link*</td><td>Ability to send SMS with discount code to customer cell phone</td><td>Integrate twilio SMS service into the store front web app (http://www.twilio.com)</td></tr>
<tr><td>Nichole</td><td>Banking Service w/ Credit Card Validation
Input: customerInfo (XML Doc), product (string[])
Output Type: boolean</td><td>TryIt Link*</td><td>Ability for a customer to make purchase with credit card</td><td>Write my own code and use local component to implement the service</td></tr>
<tr><td>Nichole</td><td><b>(REQUIRED SERVICE 1)</b>
Top Ten Words
Input: url (string)
Output Type: string[]</td><td>TryIt Link*</td><td>Download a Webpage as a string and return the top ten words used</td><td>Write my own code utilize an HTML strip library to implement the service</td></tr>
<tr><td>Nichole</td><td><b>(REQUIRED SERVICE 2)</b>
Remove Stop Words Input: str (string)
Output Type: string</td><td>TryIt Link*</td><td>Return a web page as a string and return the string with all the stop words removed</td><td>Write my own code and use local component to implement the service</td></tr>
<tr><td>Matt</td><td>Customer Registration
Input: information (string[])
Output: XML Document</td><td>TryIt Link*</td><td>Create a new customer identity so they can utilize to the store front's features</td><td>Write my own code and use local component to implement the service</td></tr>
<tr><td>Matt</td><td>Email Verification
Input: email (string)
Output: boolean</td><td>TryIt Link*</td><td>Input the new customer's email address to make sure it is a valid, working email</td><td>Integrate abstractAPI email validation services to validate new customer registration emails (https://www.abstractapi.com/email-verification-validation-api)</td></tr>
</table>

Matt	**(REQUIRED SERVICE 3)** Stemming Input: str (string) Output Type: string	TryIt Link*	Find common word amongst word and return string	Write my own code and use local component to implement the service
Matt	**(REQUIRED SERVICE 21)** Number to Words Input: number (string) Output Type: string	TryIt Link*	Convert digit string to an easier-to-remember digit-character string	Write my own code and use local component to implement the service
Chengxu	Email receipt via .txt attachment Input: email (string) Output Type: void	TryIt Link*	Send the receipt of the customer's purchase via email	Integrate Stripe.com API with own code to implement the service
Chengxu	Location Services Input: zipCode (int) Output Type: string	TryIt Link*	Locate the nearest store to the customer via zip code	Use foursquare API to integrate location services into web app
Chengxu	**(REQUIRED SERVICE 16)** Weather Service Input: US Zip Code (String) Output: array/List of Strings	TryIt Link*	Create a 5-day weather forecast service based on the zip code location	Retrieve information from OpenWeather dataset at https://openweathermap.org/current and https://openweathermap.org/forecast16
Chengxu	**(REQUIRED SERVICE 18)** Wind Energy Service Input: Latitude (decimal), Longitude (decimal) Output: decimal	TryIt Link*	Create a service that returns the annual average wind index of a given position (latitude, longitude).	Retrieve information from OpenWeather dataset at https://openweathermap.org/api/statistics-api