Third increment Report

Objectives:

The objective of this increment is to develop the employee services and extend the services of REST API implemented in the previous iterations which will be useful for developing employee services.

Existing Services/API:

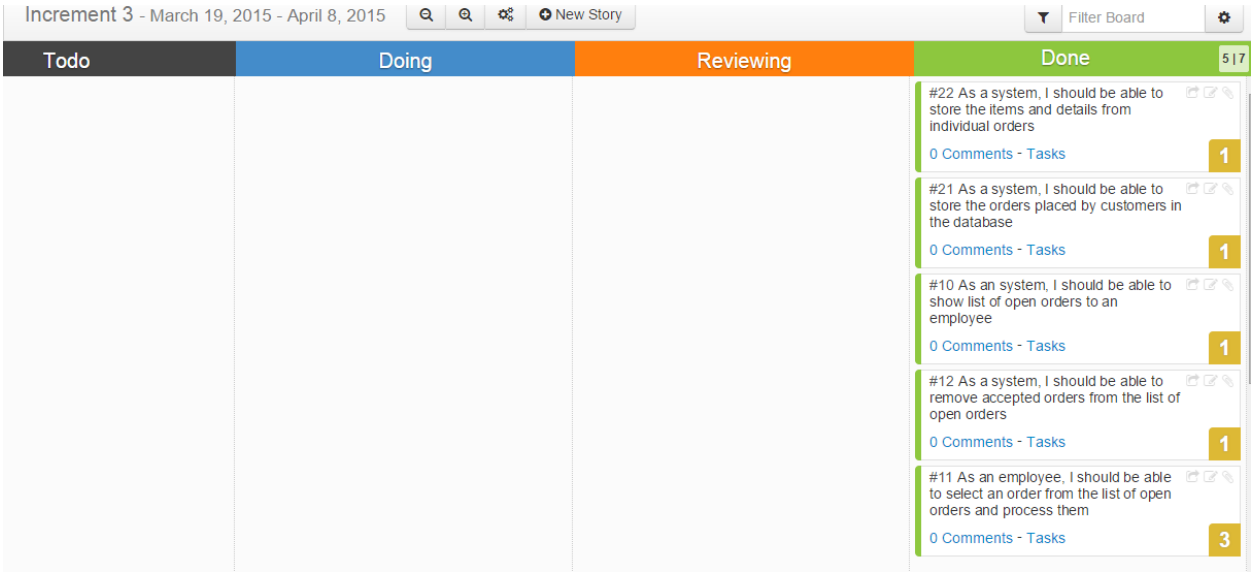
For the front end we haven't used any existing services. But for the development of mobile client we referred the following links and example codes to know how to consume REST services in client application, parsing JSON objects and few small concepts like array adapter, list view, toasts etc.

http://www.tutorialspoint.com/android/android_json_parser.htm
http://www.androidhive.info/2012/01/android-json-parsing-tutorial/
http://developer.android.com/guide/topics/ui/layout/listview.html
http://developer.android.com/reference/android/widget/ArrayAdapter.html
http://developer.android.com/guide/topics/ui/notifiers/toasts.html

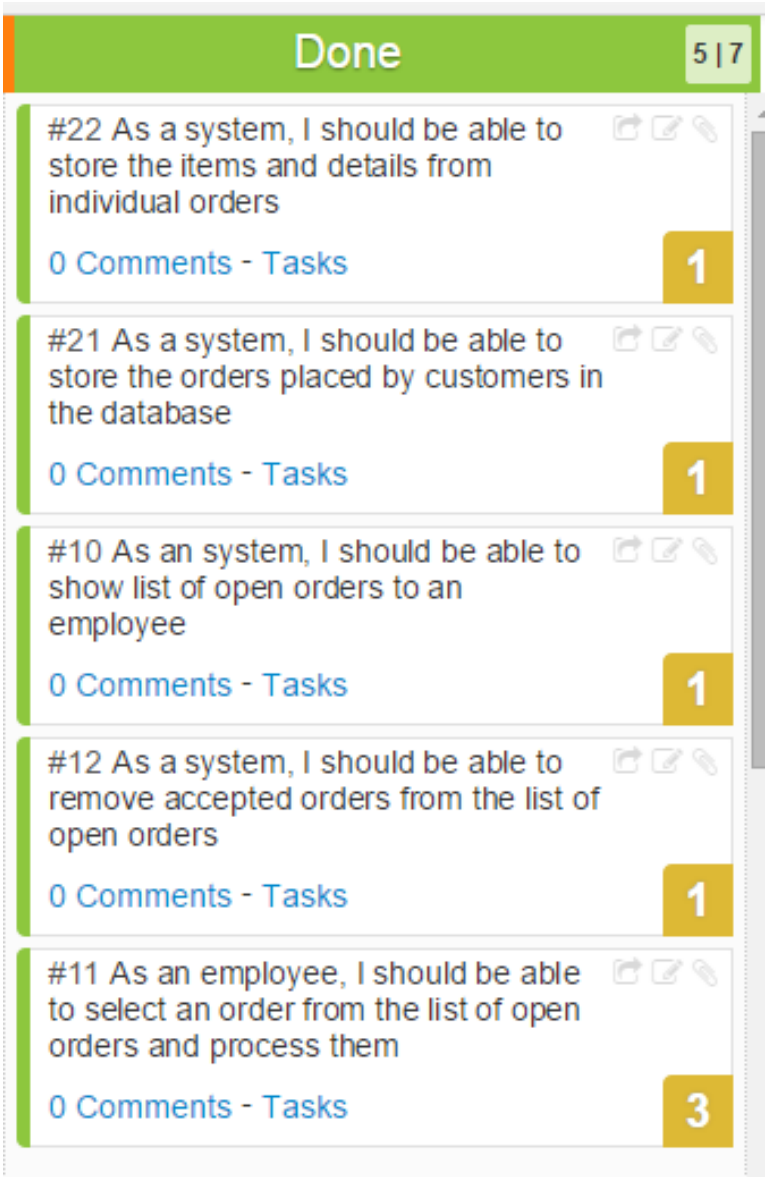
For the backend, we implemented our own API in previous iterations and extended few services in this iteration. For testing and rapid development we used **Jetty server plugin .** It will scan our project periodically for changes and redeploy the web application if any changes are found automatically

Detail design of Services:

As a recap, in the previous iteration a customer can see list of different food items (fruits, vegetables, fast foods and restaurant menu) and can add them to cart. In this iteration, we are storing all the orders placed by different customers. When an employee login to the application, he can see all the list of open orders. He can choose to pick any order and process them. Once an employee selects an order to process, it will automatically be removed from the list of open orders. When another employee logs in to the application, he can't see the processed orders.



User stories for Employee service:



Services:

We have developed the registration, login service in first increment and for second increment, we developed the customer service where customer will be able to select items and add them to cart. In this iteration apart from adding services to REST API, we implemented Employee service.

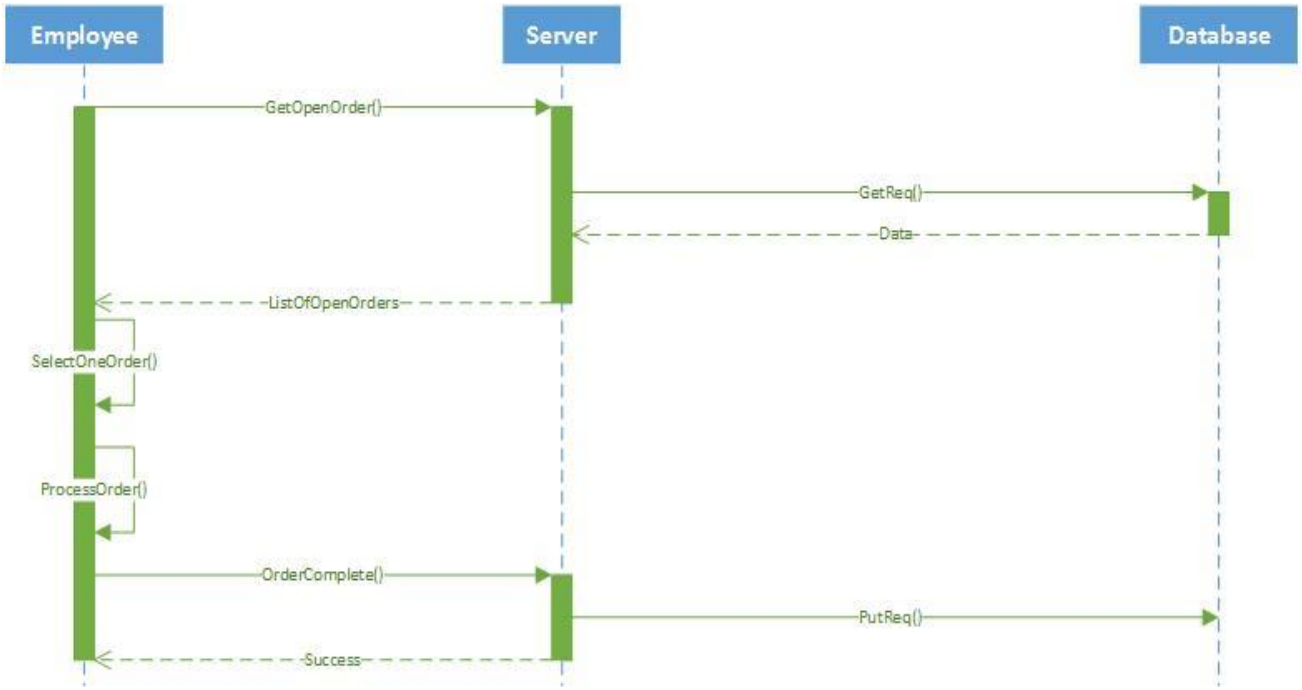
Employee service:

Service description:

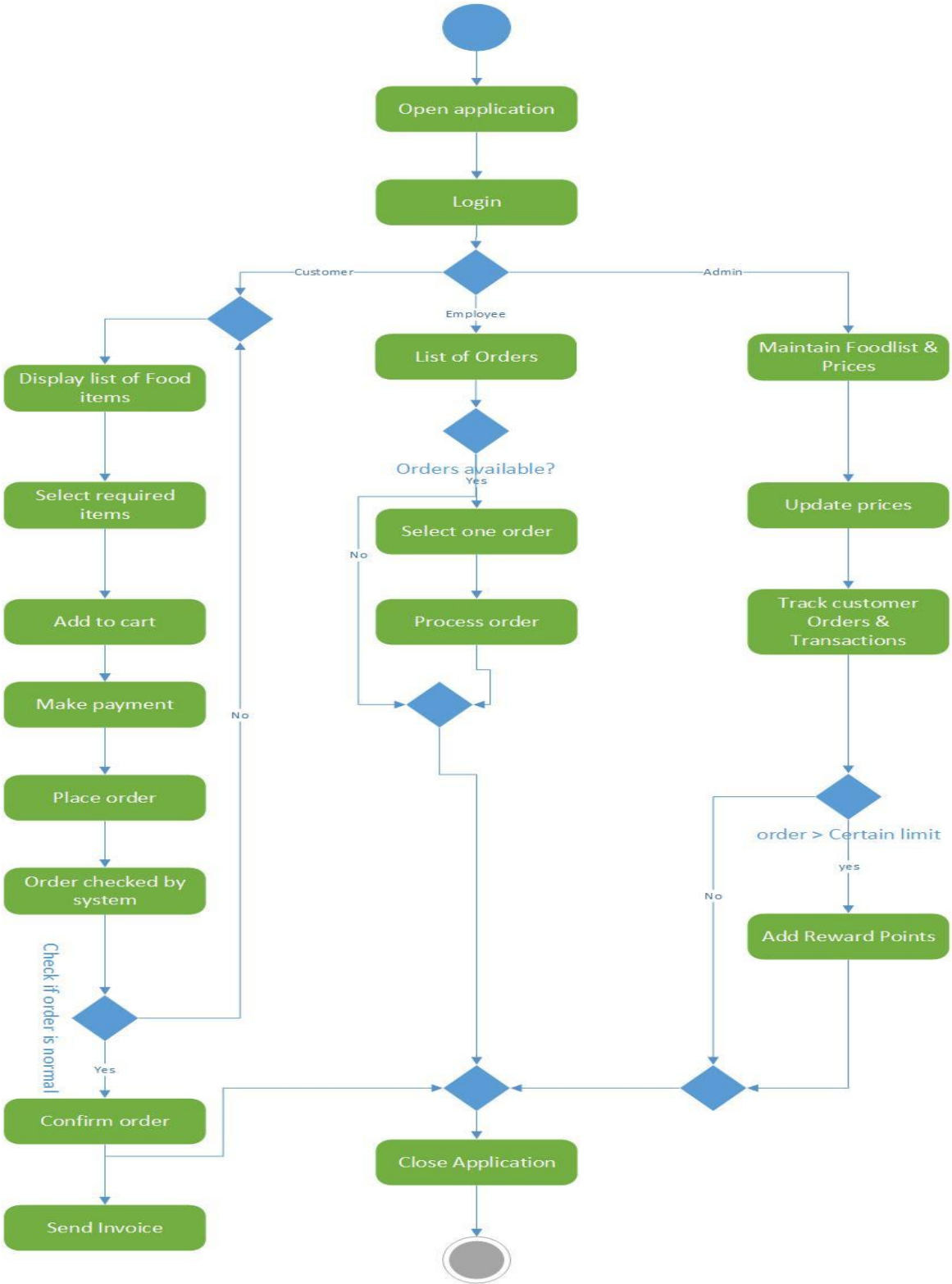
When an employee logs into the application, he can see list of open orders and accepts an order of his interest and process them. Accepted orders will be removed from the list of open orders.

Class diagram:

Sequence diagram:



Activity Diagram:



Implementation:

Implementation of REST services:

We implemented the REST services using Apache CXF. It is open source and multi featured web services framework. It takes care of all the http requests. Every request from the mobile client goes to CXF servlet.

The REST service we developed provides all the data and information about prices and description of fruits, vegetables, fast foods, restaurant menu etc.

We used MySQL as database for our project. We developed and created REST services using Apache CXF and Spring dependency injection. For mapping Java classes to database tables, we used Hibernate.

We added few more services to the API we developed in the previous iterations. Now, it is capable of obtaining and storing the orders placed by customers, provides orders and order details to the employees who logs in to the application and employee can request each order individually.

A detailed info about the implementation of REST services will be discussed later in the testing part.

Implementation of user interface:

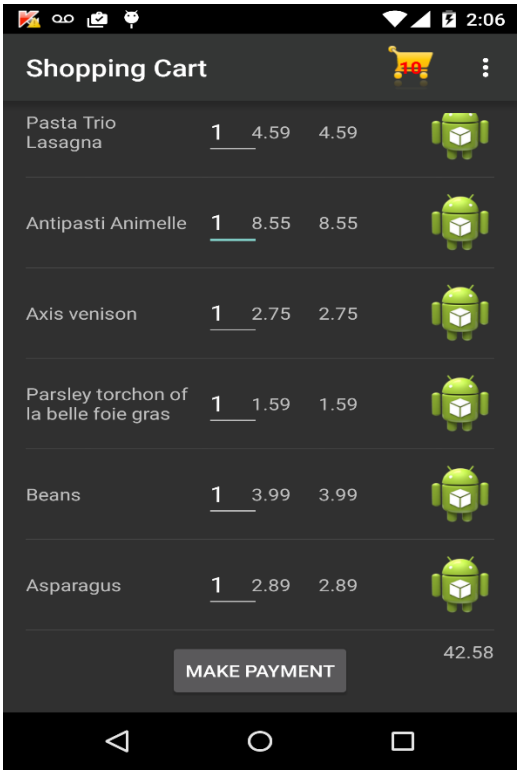
For this iteration we have developed the activity page for employee, activity page for displaying list of open orders and activity page for displaying each order (items in order and other details).

When a person logs in as an employee to the application, s/He will see list of open orders. When an employee selects an order from the list, order details activity page will appear and an employee can choose to accept and process the order.

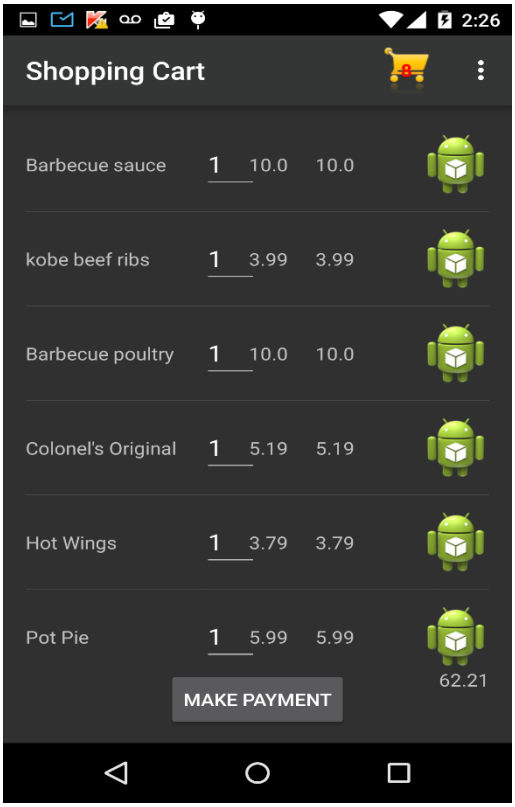
Once he selects an order for processing, no other employee logged into the application can see that order again.

Following are the screen shots for the activities discussed above

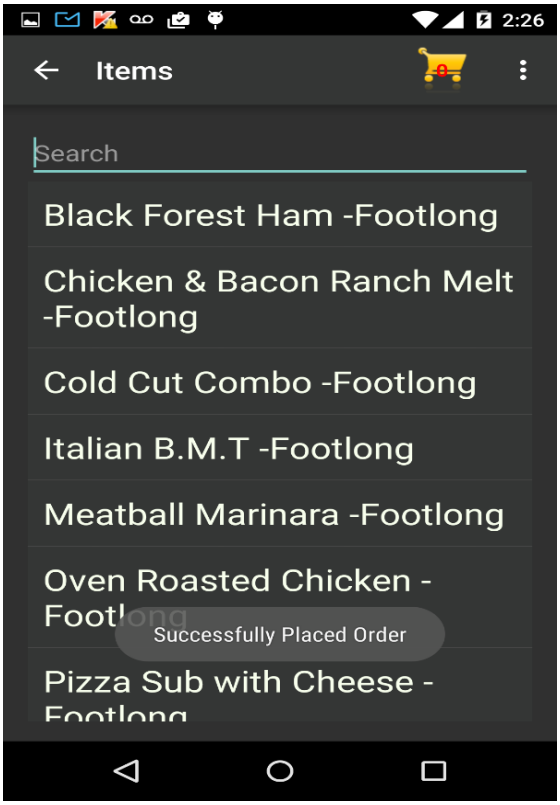
Items added to the cart by the customer 1:

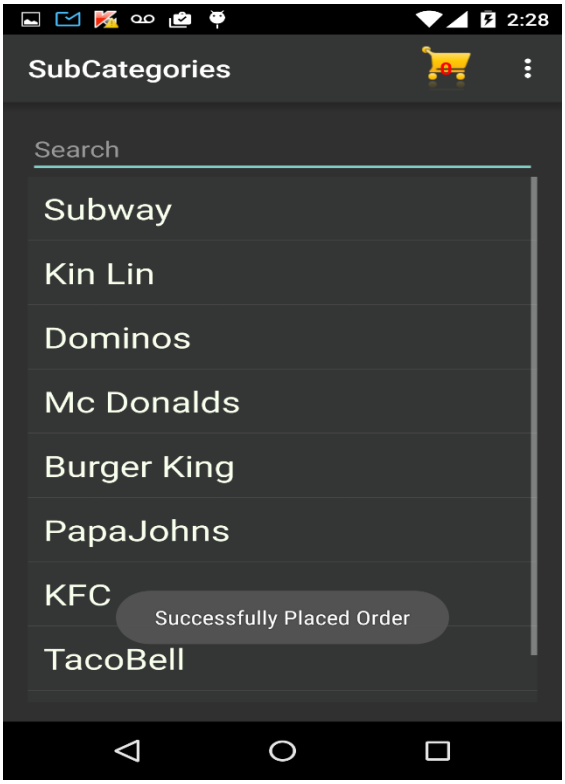


Items added to the cart by the customer 2:

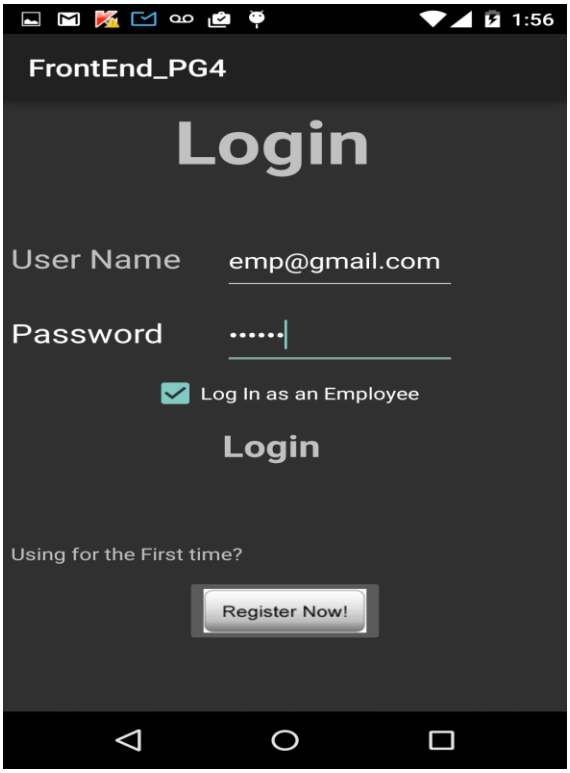


Toast showing that the order was placed successfully:

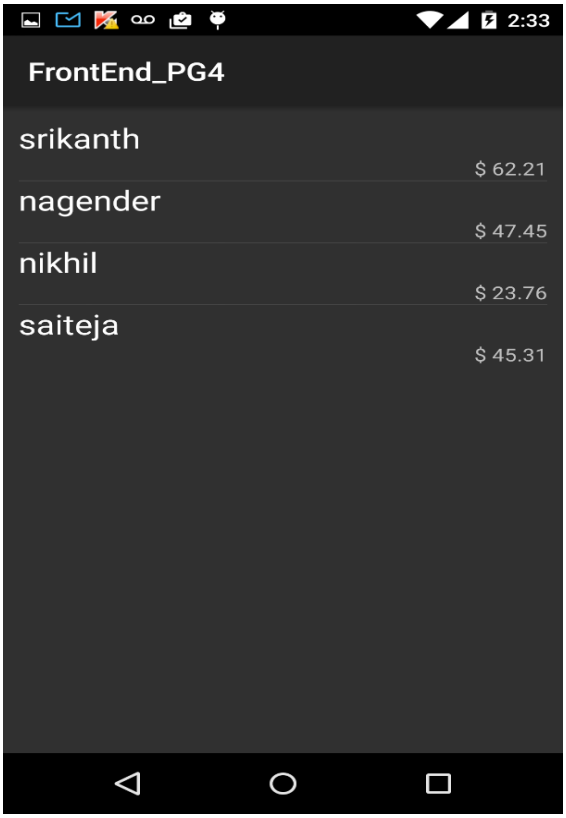




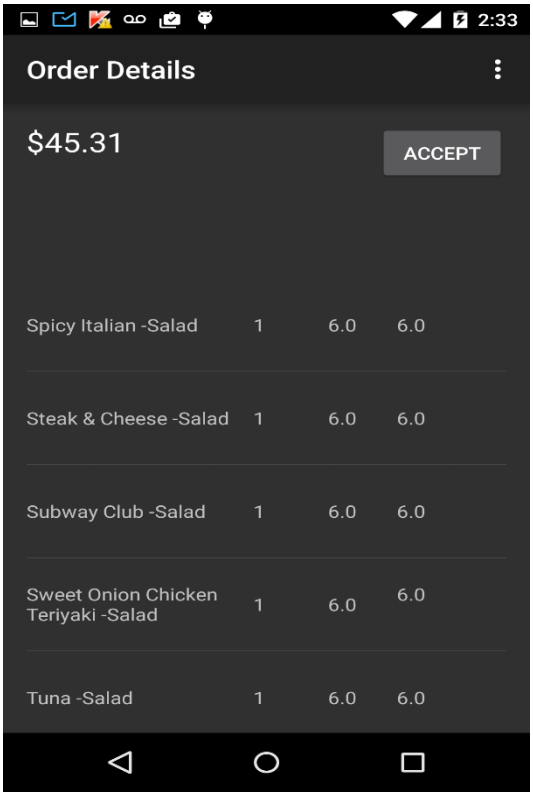
Employee logging into the application:

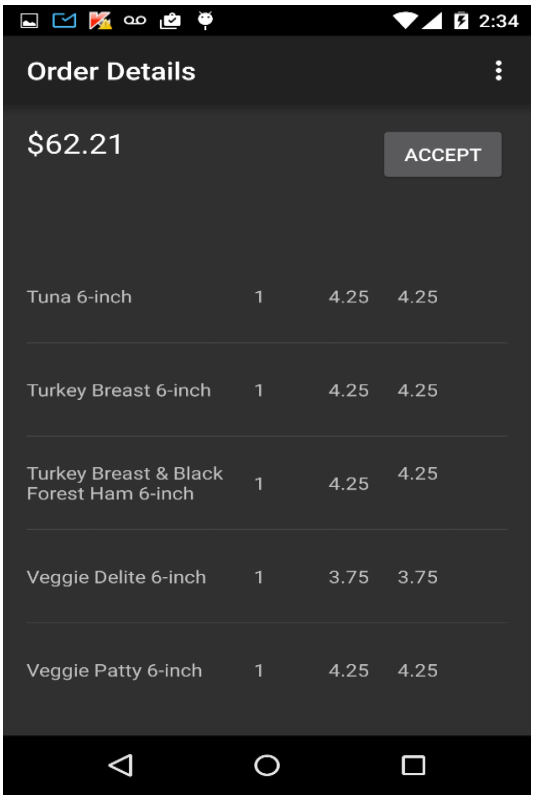


Activity page when employee logs in successfully:

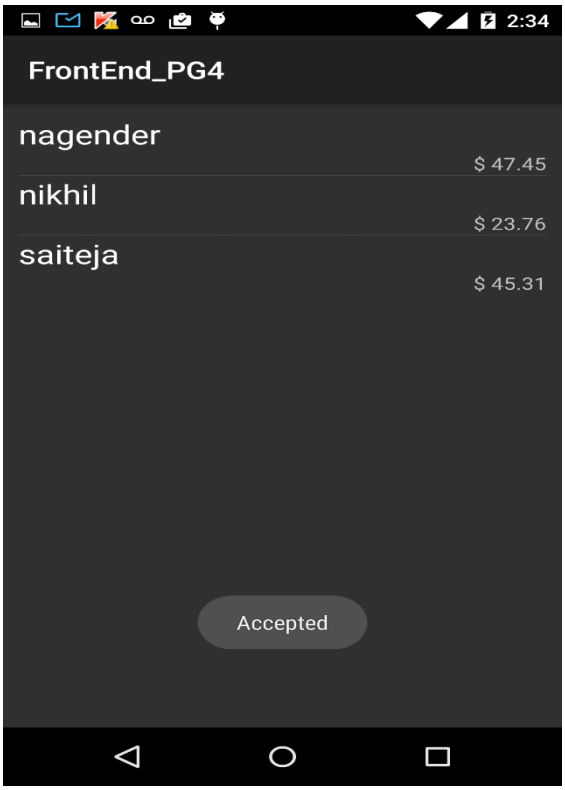


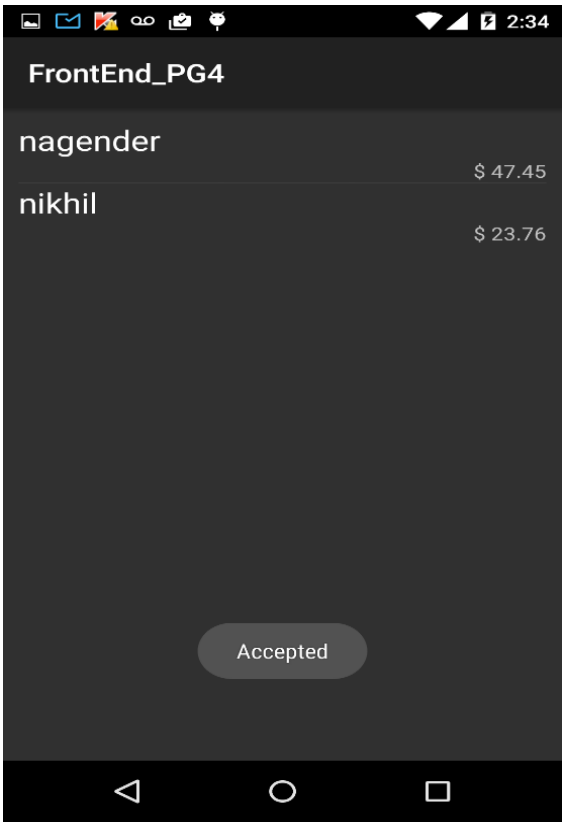
Activity page when employee selects an order from the list:

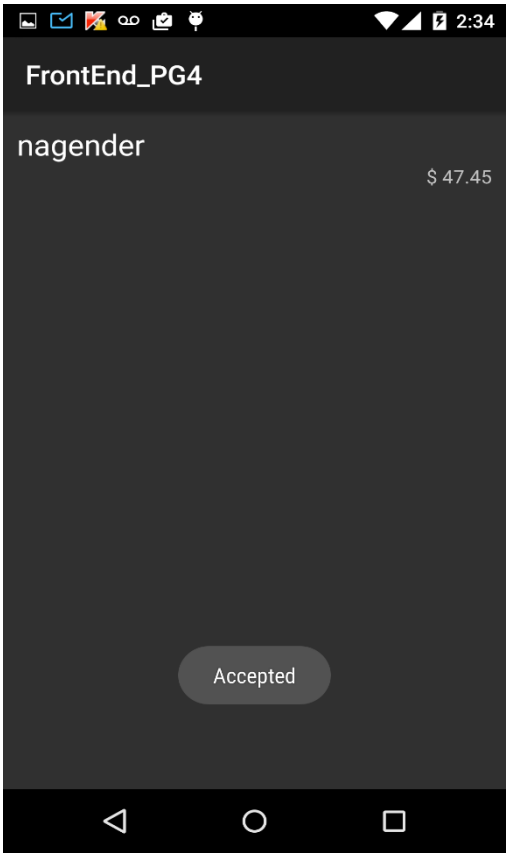




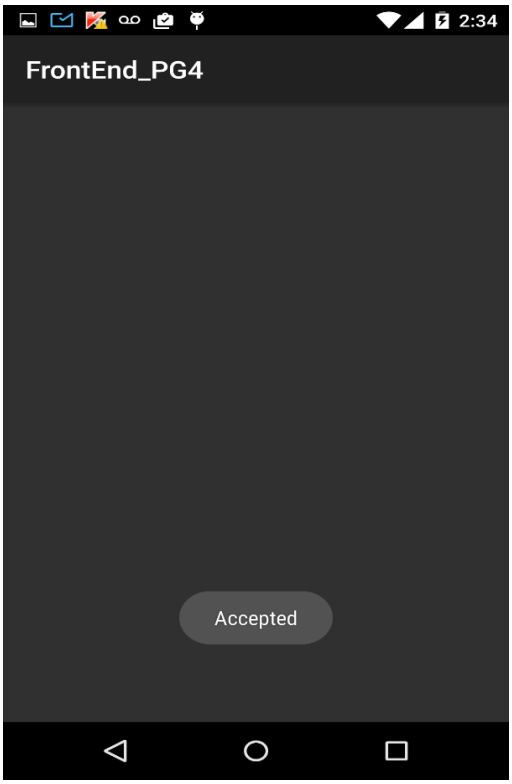
Activity page shown after employee accepts a particular order: (Order removed from the list of open orders):







Activity page showing no more open orders when all of them were accepted:



Testing:

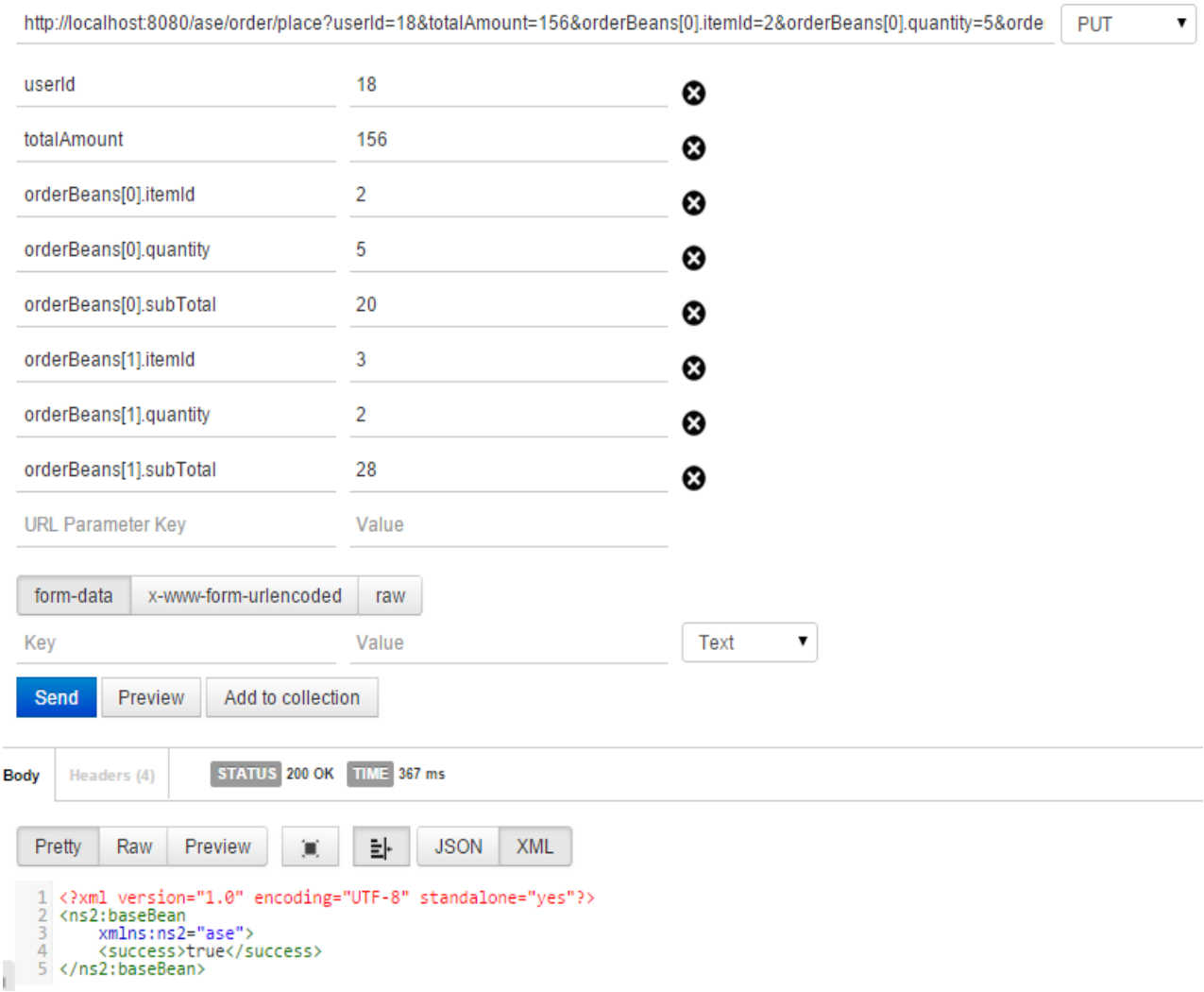
We added few services to the API we developed in the previous iterations. Now, it is capable of obtaining and storing the orders placed by customers, provides orders and order details to the employees who logs in to the application and employee can request each order individually.

We are testing these services provided by API using POSTMAN an extension of google chrome.

Placing order:

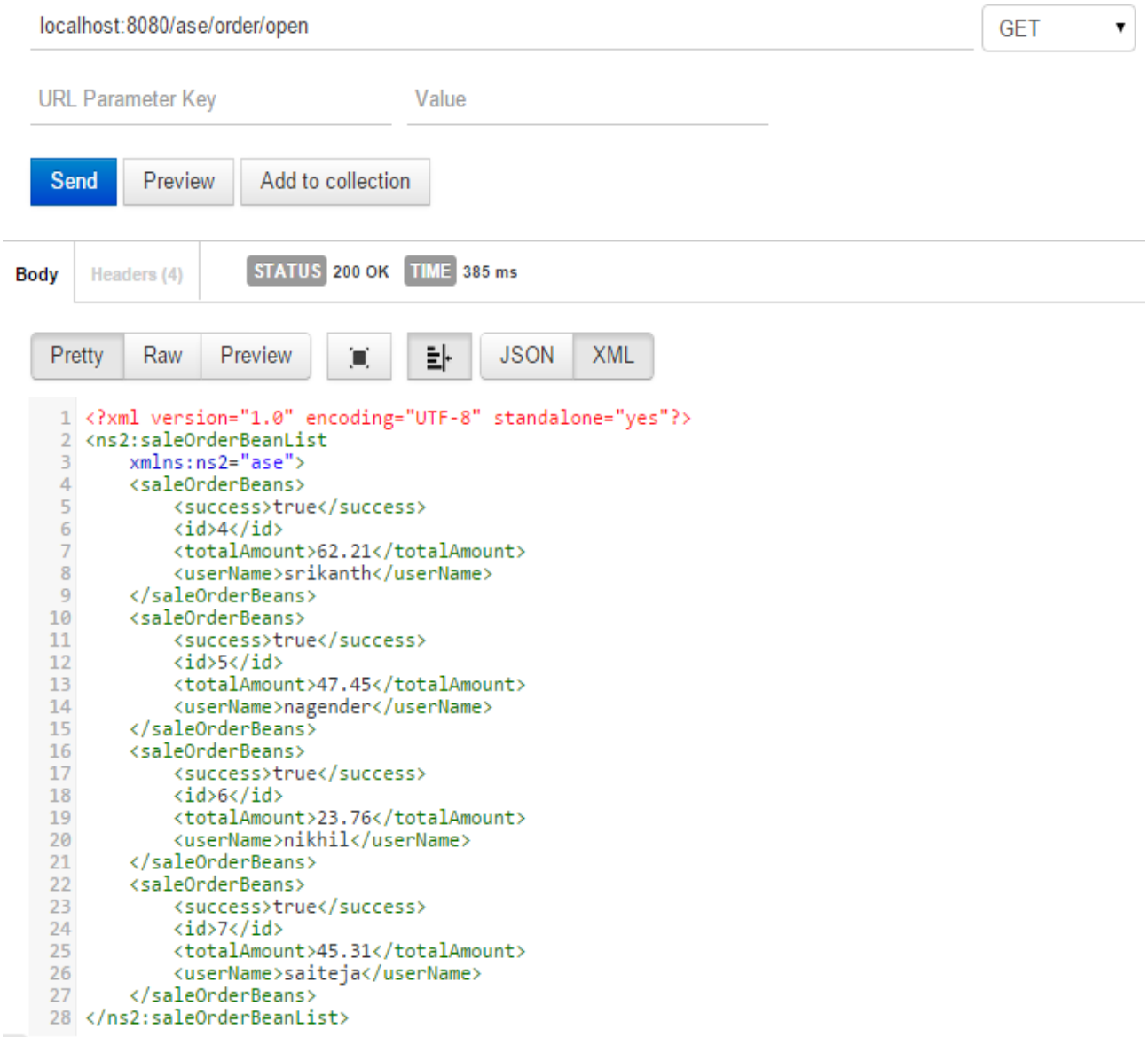
http://localhost:8080/ase/order/place

http://localhost:8080/ase/order/place?userId=18&totalAmount=156&orderBeans[0].itemId=2&orderBeans[0].quantity=5&orderBeans[0].subTotal=20&orderBeans[1].itemId=3&orderBeans[1].quantity=2&orderBeans[1].subTotal=28



List of open orders:

localhost:8080/ase/order/open

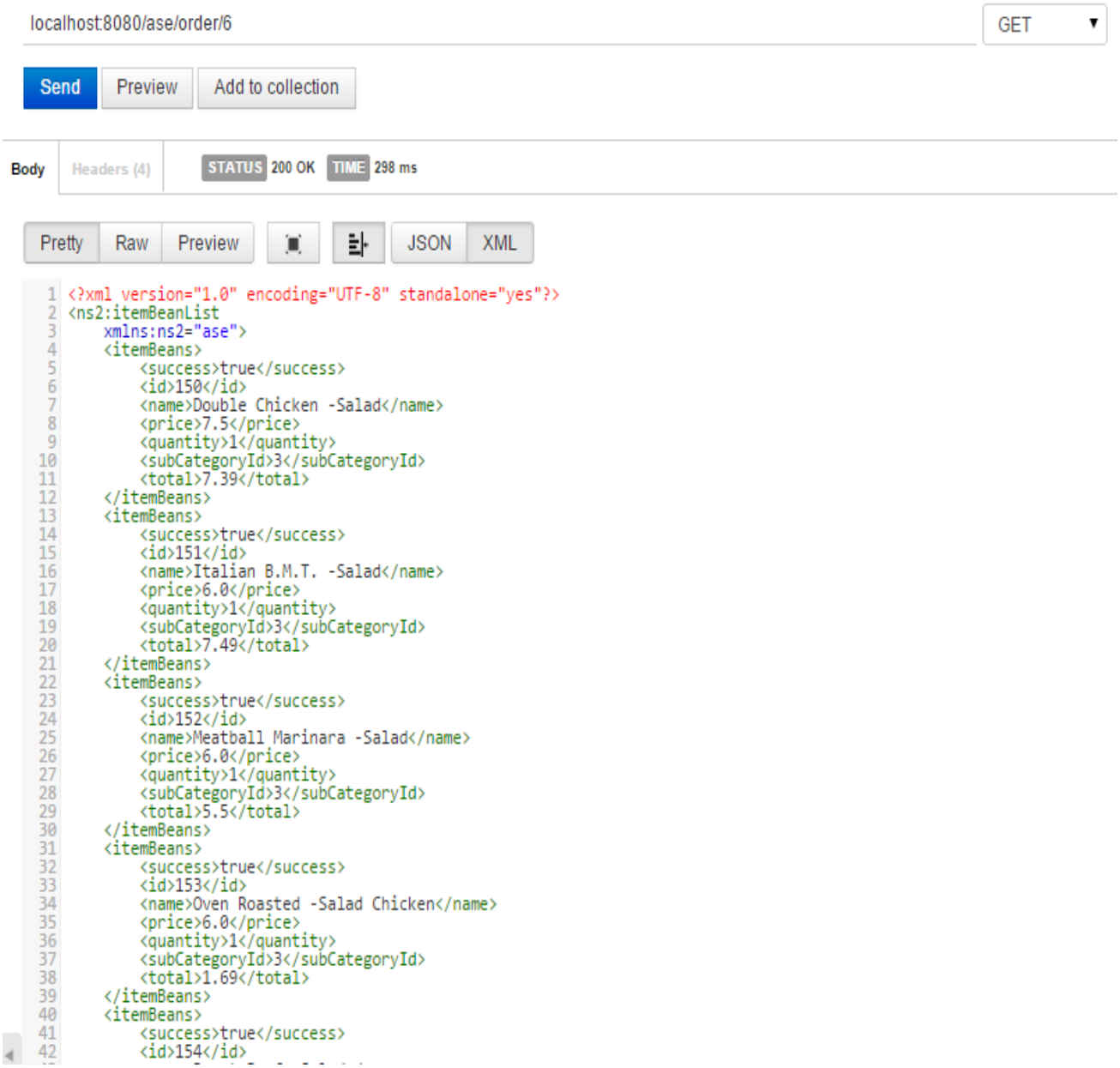



```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:saleOrderBeanList
    xmlns:ns2="ase">
    <saleOrderBeans>
        <success>true</success>
        <id>4</id>
        <totalAmount>62.21</totalAmount>
        <userName>srikanth</userName>
    </saleOrderBeans>
    <saleOrderBeans>
        <success>true</success>
        <id>5</id>
        <totalAmount>47.45</totalAmount>
        <userName>nagender</userName>
    </saleOrderBeans>
    <saleOrderBeans>
        <success>true</success>
        <id>6</id>
        <totalAmount>23.76</totalAmount>
        <userName>nikhil</userName>
    </saleOrderBeans>
    <saleOrderBeans>
        <success>true</success>
        <id>7</id>
        <totalAmount>45.31</totalAmount>
        <userName>saiteja</userName>
    </saleOrderBeans>
</ns2:saleOrderBeanList>
```

Retrieving each order individually:

localhost:8080/ase/order/{saleOrderId}

example: localhost:8080/ase/order/6




```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:itemBeanList
    xmlns:ns2="ase">
    <itemBeans>
        <success>true</success>
        <id>150</id>
        <name>Double Chicken -Salad</name>
        <price>7.5</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>7.39</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>151</id>
        <name>Italian B.M.T. -Salad</name>
        <price>6.0</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>7.49</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>152</id>
        <name>Meatball Marinara -Salad</name>
        <price>6.0</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>5.5</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>153</id>
        <name>Oven Roasted -Salad Chicken</name>
        <price>6.0</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>1.69</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>154</id>
```

localhost:8080/ase/order/7 GET

Send Preview Add to collection

Body Headers (4) STATUS 200 OK TIME 417 ms

Pretty Raw Preview JSON XML

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:itemBeanList
    xmlns:ns2="ase">
    <itemBeans>
        <success>true</success>
        <id>155</id>
        <name>Spicy Italian -Salad</name>
        <price>6.0</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>10.0</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>156</id>
        <name>Steak &amp;amp; Cheese -Salad</name>
        <price>6.0</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>6.55</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>157</id>
        <name>Subway Club -Salad</name>
        <price>6.0</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>4.99</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>158</id>
        <name>Sweet Onion Chicken Teriyaki -Salad</name>
        <price>6.0</price>
        <quantity>1</quantity>
        <subCategoryId>3</subCategoryId>
        <total>14.0</total>
    </itemBeans>
```

localhost:8080/ase/order/2 GET

Send Preview Add to collection

Body Headers (4) STATUS 200 OK TIME 178 ms

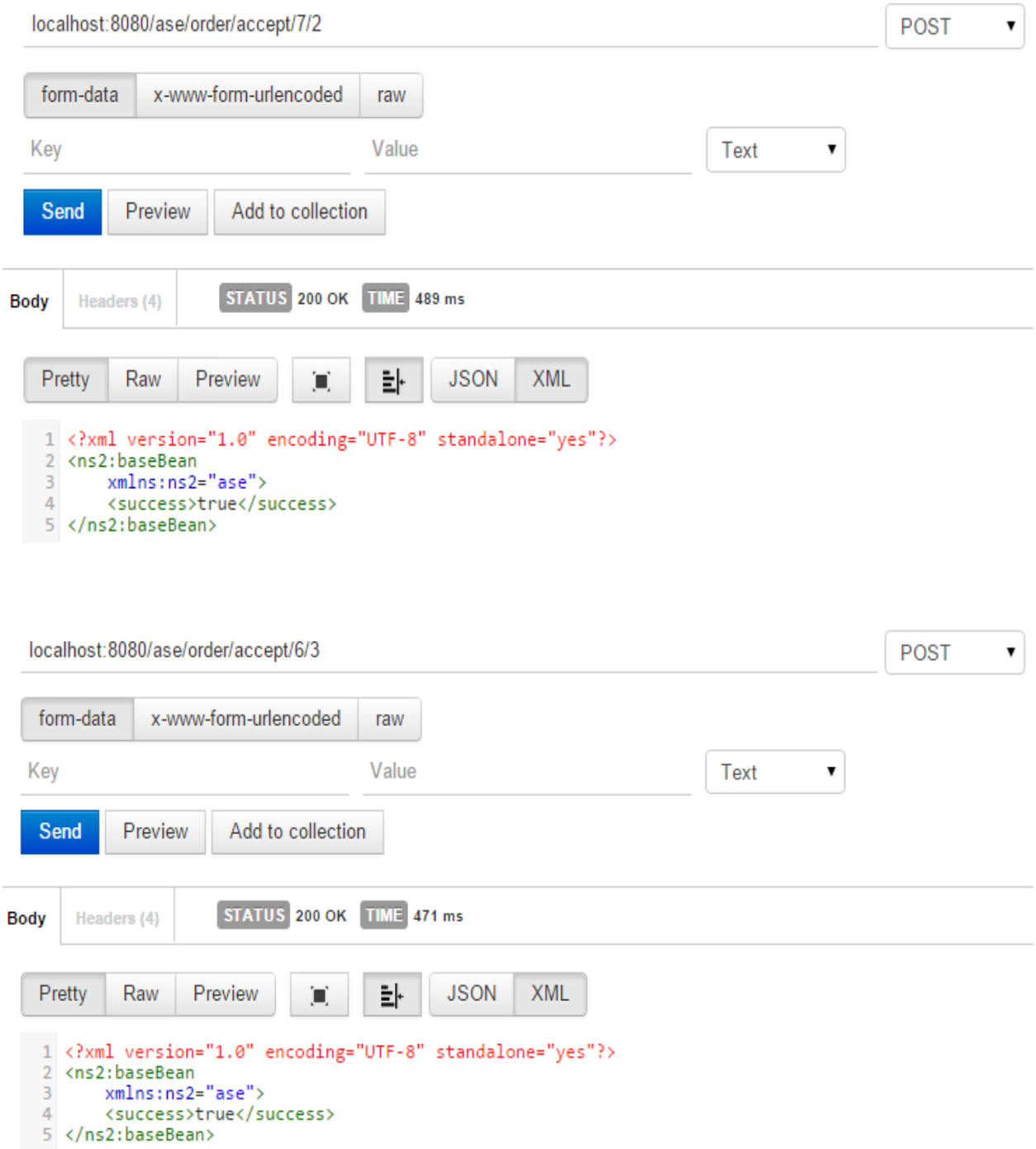
Pretty Raw Preview JSON XML

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:itemBeanList
    xmlns:ns2="ase">
    <itemBeans>
        <success>true</success>
        <id>104</id>
        <name>Sweet Potato</name>
        <price>1.59</price>
        <quantity>1</quantity>
        <subCategoryId>2</subCategoryId>
        <total>3.99</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>105</id>
        <name>Taro</name>
        <price>3.99</price>
        <quantity>1</quantity>
        <subCategoryId>2</subCategoryId>
        <total>2.89</total>
    </itemBeans>
    <itemBeans>
        <success>true</success>
        <id>106</id>
        <name>Tomatillo</name>
        <price>5.99</price>
        <quantity>1</quantity>
        <subCategoryId>2</subCategoryId>
        <total>3.99</total>
    </itemBeans>
</ns2:itemBeanList>
```

Accepting open orders:

localhost:8080/ase/order/accept /{saleOrderId}/{employeeId}

example localhost:8080/ase/order/accept/7/2



Deployment:

ScrumDo link:

https://www.scrumdo.com/projects/project/alpha8/iteration/121756/board

UMKC VM:

Since we developed backend in JAVA, we cannot use UMKC VM (only supports .NET)
We are using jetty server plugin in the backend project (maven).

- **http://localhost:8080/ase/order/place**
- **http://localhost:8080/ase/order/place?userId=18&totalAmount=156&orderBeans[0].itemId=2&orderBeans[0].quantity=5&orderBeans[0].subTotal=20&orderBeans[1].itemId=3&orderBeans[1].quantity=2&orderBeans[1].subTotal=28**
- **localhost:8080/ase/order/open**
- localhost:8080/ase/order/accept /{saleOrderId}/{employeeId}
- localhost:8080/ase/order/accept/7/2

GitHub url:

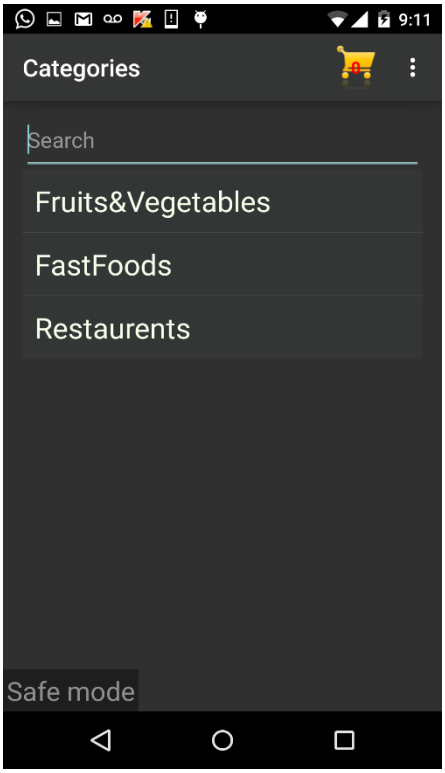
https://github.com/saiteja10/ASE_INCREMENT_3

Report:

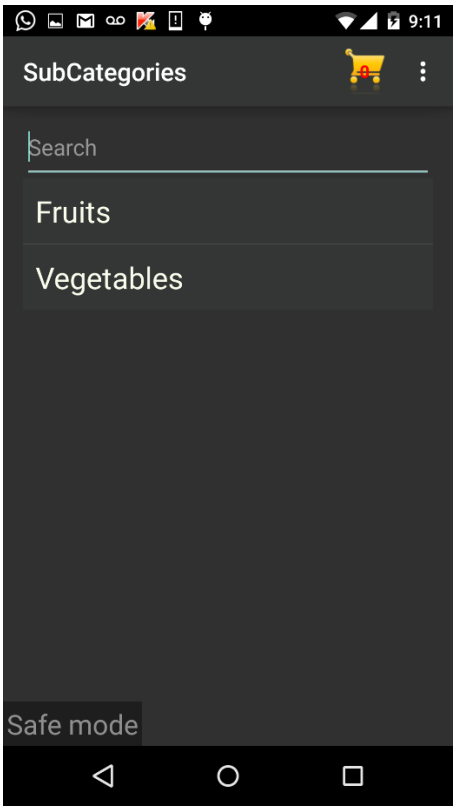
Mobile client Screenshots:

Login page: (customer)

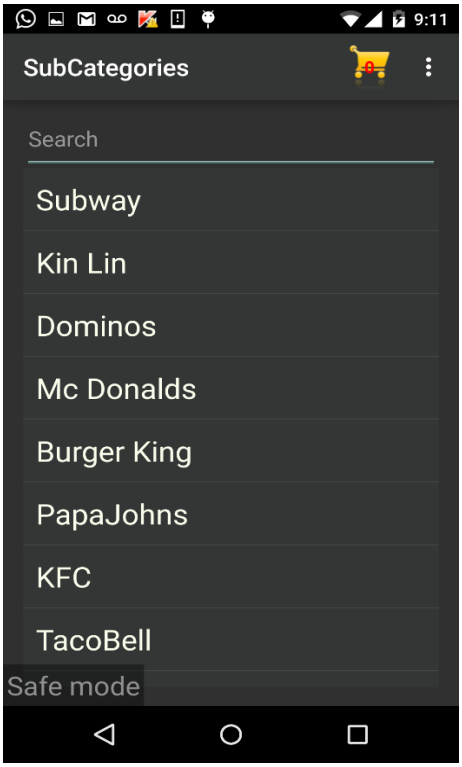
When customer logs in(Categories page):



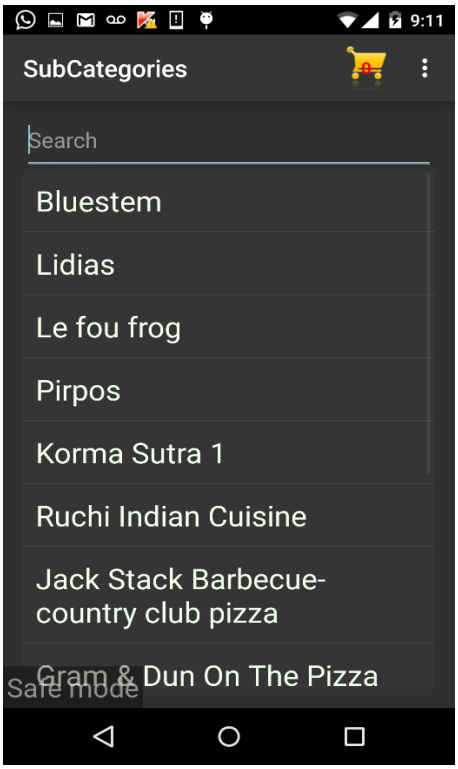
Subcategory page(on tapping fruits and vegetables from category page)



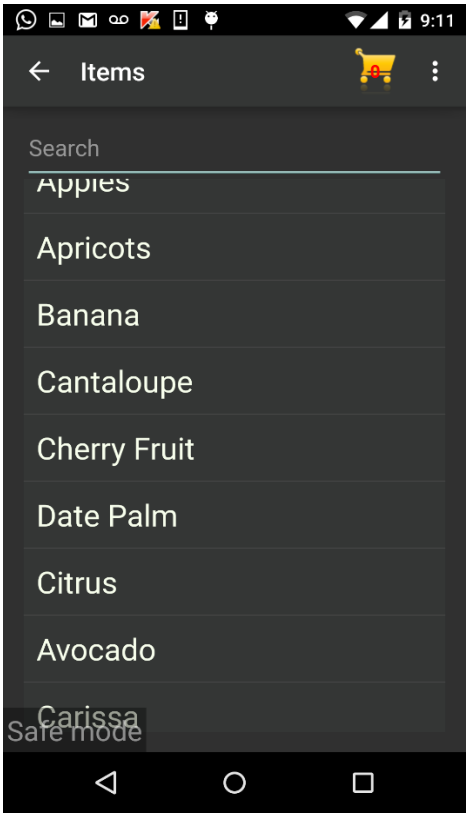
Subcategory page(on tapping Fastfoods from category page)



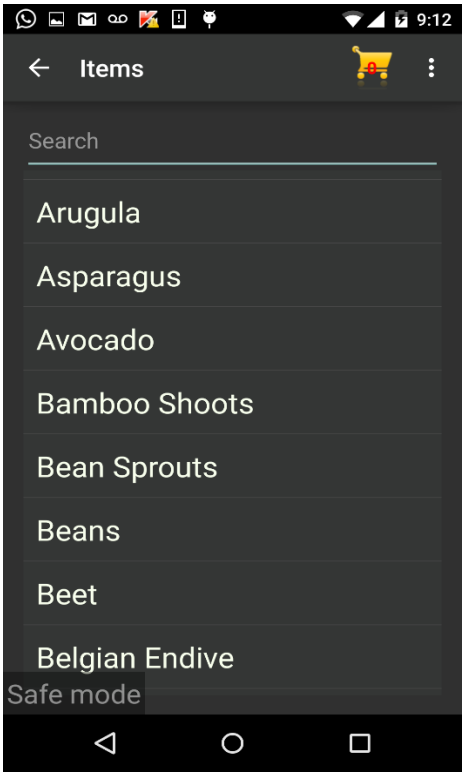
Subcategory page(on tapping Restaurents from category page)



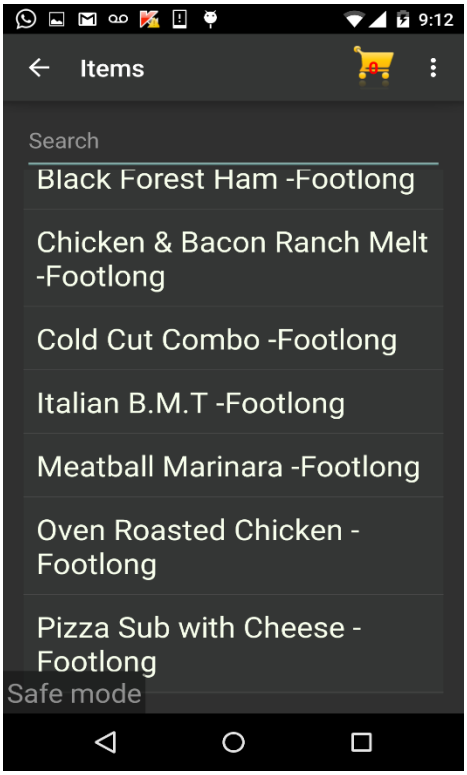
Items page(on tapping fruits from subcategory page)



Items page(Vegetables):



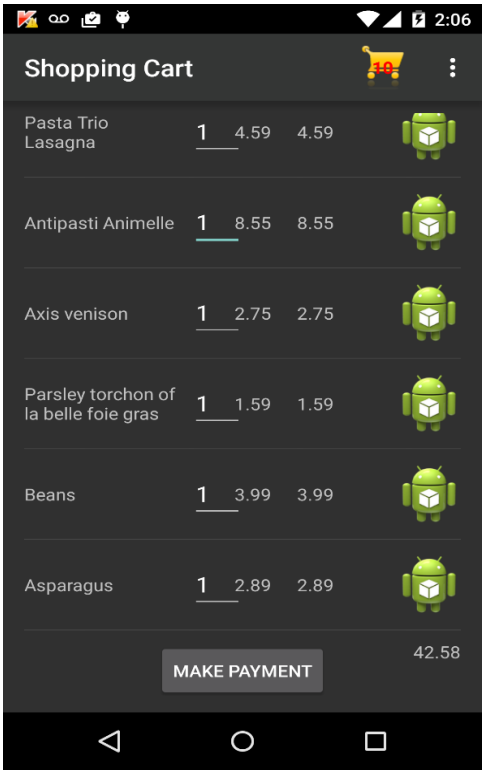
Items page(Subway menu):



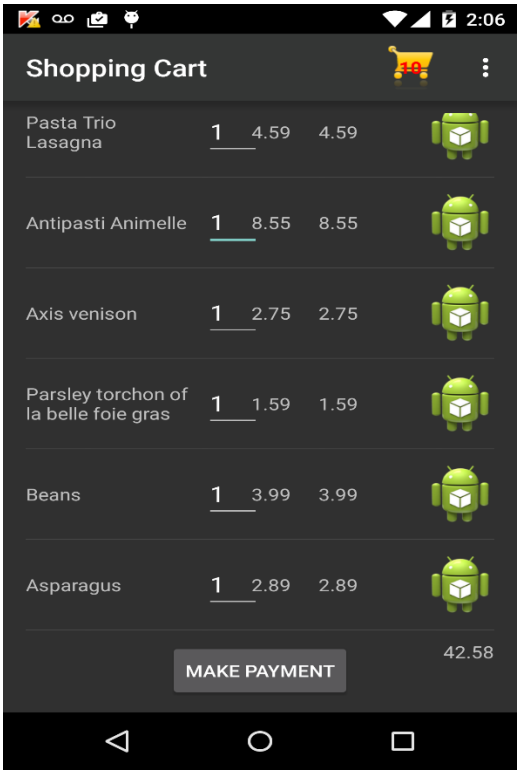
Items page(Burger King menu)

Items page(Ruchi restaurant menu)

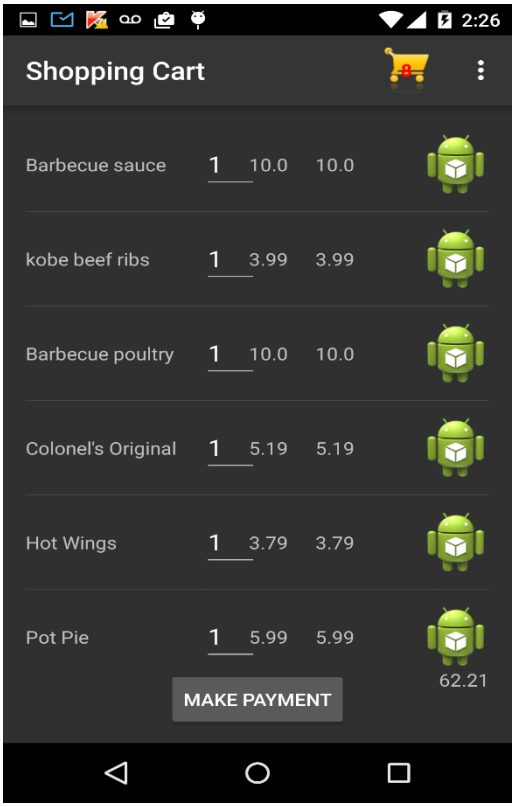
Shopping cart:



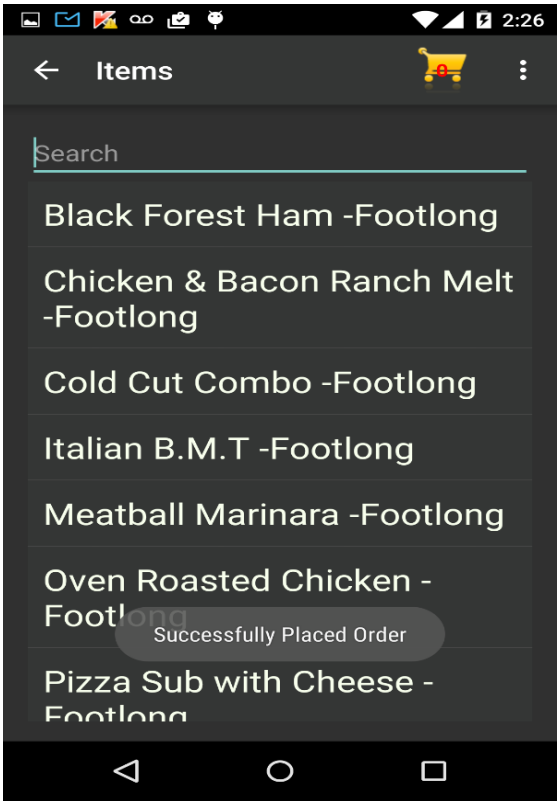
Items added to the cart by the customer 1:

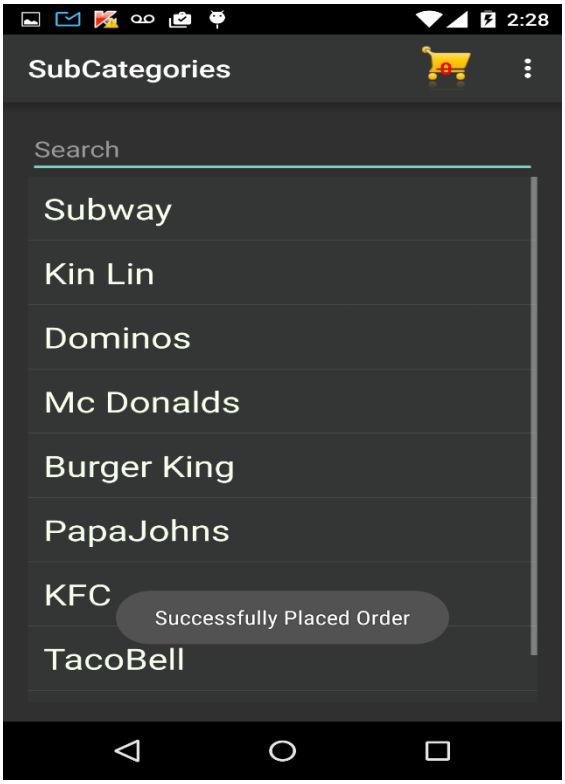


Items added to the cart by the customer 2:

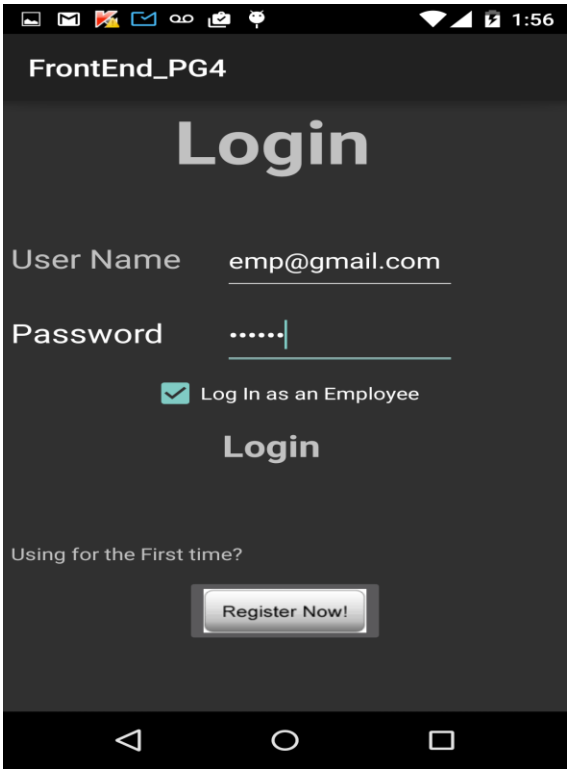


Toast showing that the order was placed successfully:

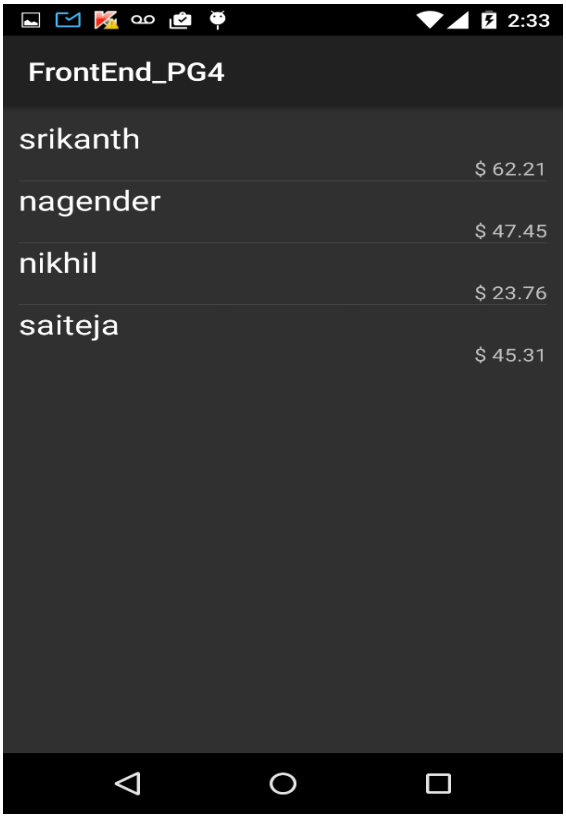




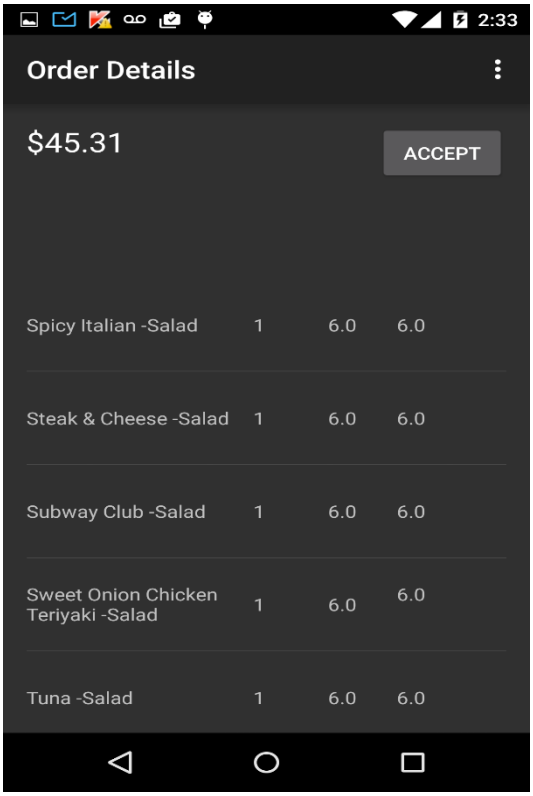
Employee logging into the application:

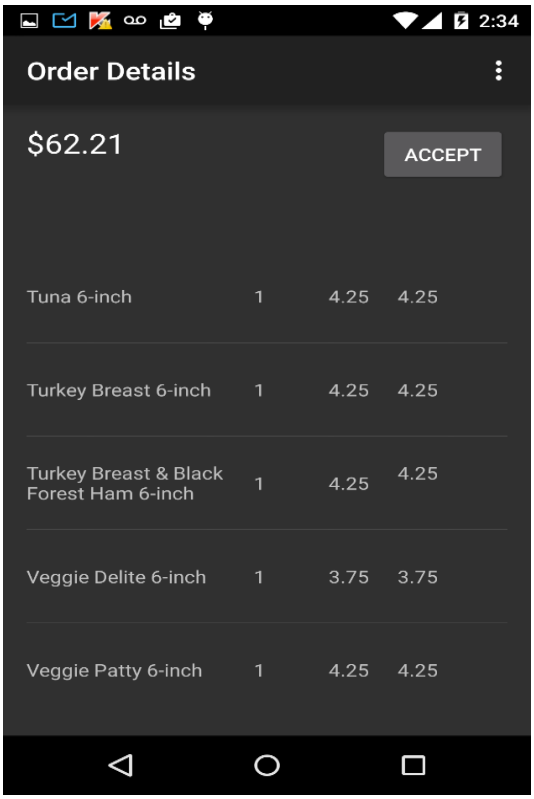


Activity page when employee logs in successfully:

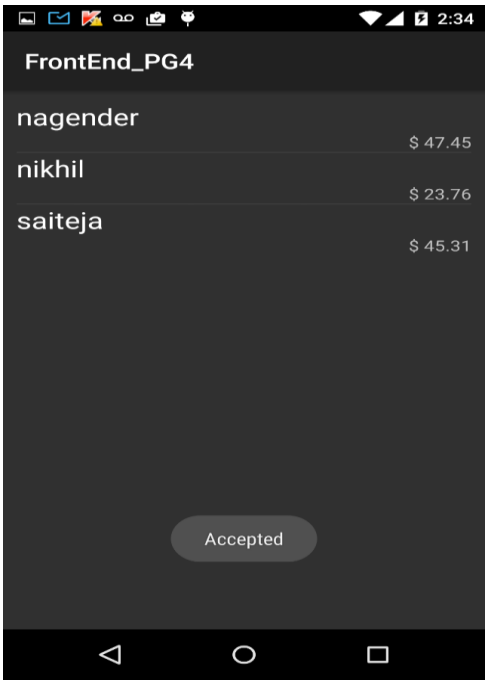


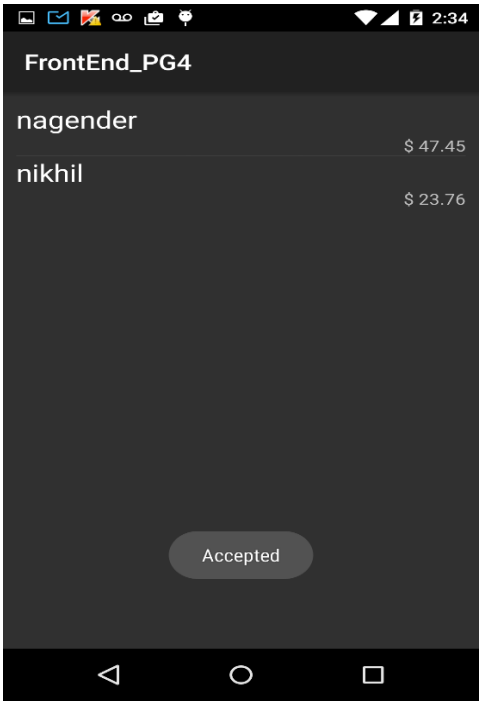
Activity page when employee selects an order from the list:

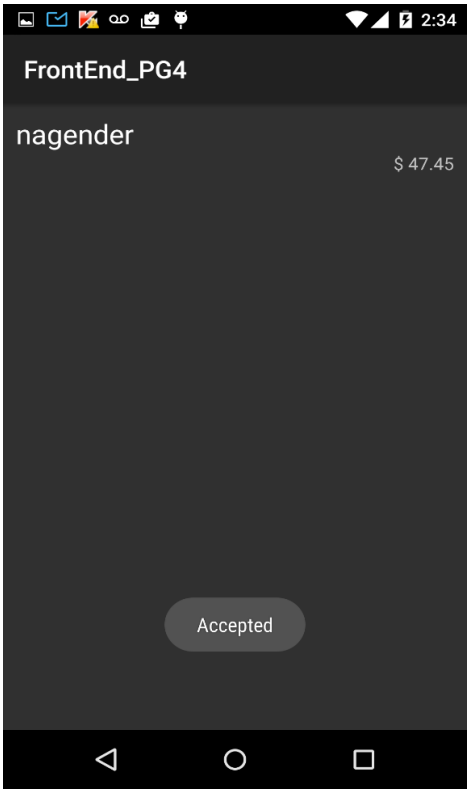




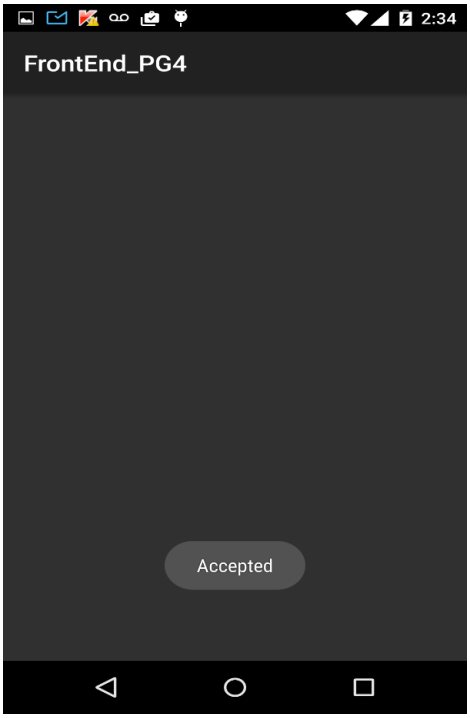
Activity page shown after employee accepts a particular order: (Order removed from the list of open orders):







Activity page showing no more open orders when all of them were accepted:



Project Management:

Scrumdo link:
https://www.scrumdo.com/projects/project/alpha8/iteration/121756/board

Implementation status report:

Work completed:

In this iteration, we added few more services to the API we developed in the previous iterations. Now, it is capable of obtaining and storing the orders placed by customers, provides orders and order details to the employees who logs in to the application and employee can request each order individually.

Also, we have developed the activity page for employee, activity page for displaying list of open orders, activity page for displaying each order (items in order and other details) and functionality for accepting and processing orders.

Task	**Person**
Wrote queries for the newly added tables	Sai Teja Saranam
Functionality of mobile client	Sai Teja Saranam
Call rest services from android	Sai Teja Saranam
Parsing and representing data	Sai Teja Saranam
Layout of mobile client	Srikanth Nadendla
Creation of local database to store items in cart	Srikanth Nadendla, Ram Nikhil
Implementation of REST services (Apache CXF)	Sai Teja Saranam
Hibernate (mapping Java classes to database tables)	Nagender Goud, Srikanth Nadendla
Spring Dependency Injection	Nagender Goud, Ram Nikhil
Testing REST services	Nagender Goud, Ram Nikhil, Srikanth Nadendla
Contributions	
Member	**Percentage**
Sai Teja Saranam	30
Srikanth Nadendla	30
Nagender Goud	20
Ram Nikhil	20