

SHOPCONNECT

- Online Grocery Delivery Service

12/5/2014

**The UNIVERSITY OF TEXAS AT DALLAS**

GROUP 6

Swathi Damacherla

Whitney Dodson

Abhay Satish Joshi

Huidan Xu

Guangi Zhang

**Contents**

Introduction............................................................................................................................

Problem Statement.................................................................................................................

Proposed System Context Diagram........................................................................................

Proposed system Use case diagram......................................................................................

Use case description.............................................................................................................

Class diagram with methods...............................................................................................

Class diagram without methods............................................................................................

Database Design.................................................................................................................

Database Constraints………………………………………………………………………………………………….

Proposed system Sequence Diagram…………………………………………………………………………….

Functional Specification Document...............................................................................

Data Dictionary Notation....................................................................................................

Interface Design…………………………………………………………………………………………………………

Object Design………………………………………………………………………………………………………………

System Control Design......................................................................................................

Weekly Timeline.............................................................................................................

Minutes of Meeting......................................................................................................

References.....................................................................................................................

YouTube Link...............................................................................................................

INTRODUCTION

Currently, there is no convenient method for students living on university campuses without personal transportation to acquire essential living items such as food, toiletries, etc. Many students living on campus and some who live in other areas do not have a vehicle during their first few years of college, particularly international students who are in a country for the first time while attending a university. Many of these students use public transportation when they need to get from point A to point B, but this can be a long process of looking up pickup/drop-off schedules, walking to a particular stop, waiting extra time for busses or trains that only run in long intervals, etc. Furthermore, a student can only carry a limited amount of items on a train or bus. There should be a quicker and more convenient way for student to obtain necessities without having to plan their whole day around the process. The objective of this project is to create an application that serves students who need a more expedient way to purchase their needed items. The application would need to connect students with other users would could purchase items for them and deliver the goods for a specific fee. In order to create a useful application several key factors must be addressed including geo-location mapping, user data, affiliations with local retailers and logistical requirements.

PROBLEM STATEMENT

What if you could connect to friends and even unknown people from the city to help you with your daily needs? Make money while you are shopping for yourself!! One can save fuel and time and contribute to the environment.

ShopConnect provides the ultimate tool in your palm as an application on your phones and tablets to connect to friends and strangers which help facilitate your needs to your doorstep.

As a user, ShopConnect protects your bank details and manages cash flows through secure money transactions between user and deliverer.

As a paid user, you will be eligible to receive orders on your account and make money while shopping for yourself as well as order goods to your doorstep.

As a free user, you will be able to order products online and have them delivered at your doorstep for a meager amount saving you time and fuel.

**Strengths:**

* First of its kind application to connect people to help each other.
* Helps save time and fuel.
* Extra earning potential from everyday chores.
* Promotes sales in stores which do not have outreach and delivery service.

**Weakness:**

* Do not know market potential.
* Dependent on phone networks for GPS Tracking.
* Need tie ups with major stores to allow for transactions.
* Money transactions need to be handled well.

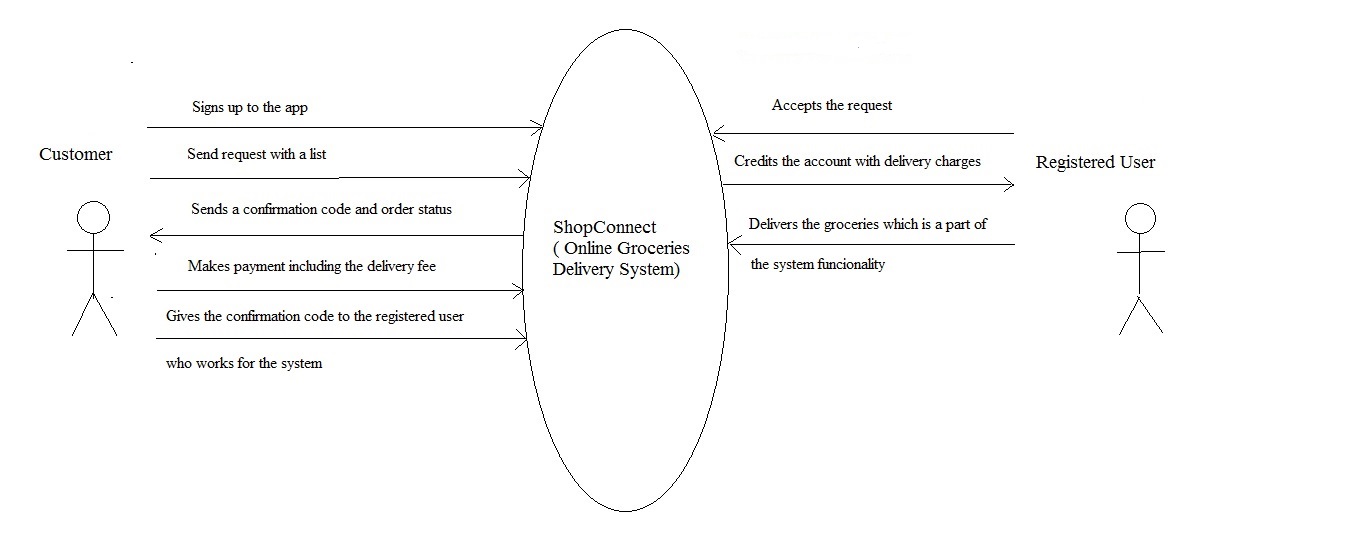
**Others:**

No Existing direct Competition

**Trends:**

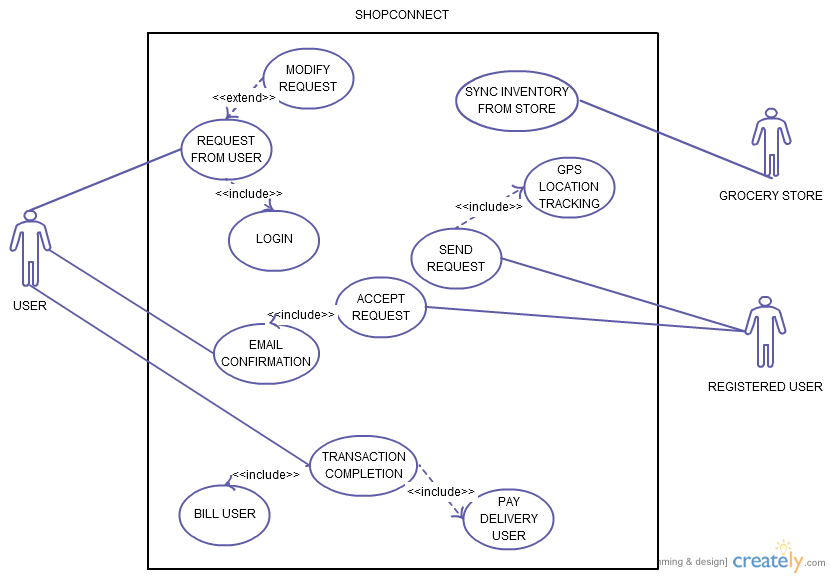
Have to see how the application gains trust in the market.

PROPOSED SYSTEM CONTEXT DIAGRAM

****

**Figure 1: Context Diagram**

PROPOSED SYSTEM USE CASE DIAGRAM



**Figure 2: Use Case Diagram**

USE CASE DESCRIPTION

**Use Case 1: Login**

|  |
| --- |
| **Use Case Name**: Login |
| **Primary Actor**: User |
| **Stakeholders**: ShopConnect System |
| **Trigger**: A user requires online delivery of groceries and opens the app |
| **Relationships**:  Includes: None  Extends: None |
| **Normal flow of events**:   1. The user requires online delivery of groceries and opens the app. 2. The user enters the required information like name, address, phone number, e-mail address and payment details. 3. The user accepts the terms and conditions and logs into the app. 4. The user has to use the same login ID and password everytime he wants the online delivery. |
| **Exceptional Flow**:  None |

**Use Case 2: Request from user**

|  |
| --- |
| **Use Case Name**: Request from user |
| **Primary Actor**: User |
| **Stakeholders**: ShopConnect System |
| **Trigger**: The user chooses groceries from the inventory. |
| **Relationships**:  Includes: Login  Extends: Modify Request |
| **Normal flow of events**:   1. After the user logs into the system, there is a screen which shows all the grocery stores. 2. The user selects a store of his choice from the list which takes him to the inventory page. 3. The user selects the items and quantity from the inventory and proceeds to checkout. 4. The user then puts a request to the system. |
| **Exceptional Flow**:   1. The user can go back and modify his request within a particular time. |

**Use Case 3: Modify Request**

|  |
| --- |
| **Use Case Name**: Modify Request |
| **Primary Actor**: User |
| **Stakeholders**: ShopConnect System |
| **Trigger**: User needs to change his list of items after he puts in a request. |
| **Relationships**:  Includes: none  Extends: none |
| **Normal flow of events**:   1. The user clicks modify button to revise the request within a particular time 2. App allows the user to pick a grocery store of choices again. This takes the user to inventory page 3. The user chooses the item and quality and proceeds to checkout. 4. The user continues to send the request to the system. |
| **Exceptional Flow**:  none |

**Use Case 4: Sync Inventory From Stores**

|  |
| --- |
| **Use Case Name**: Sync Inventory From Stores |
| **Primary Actor**: ShopConnect |
| **Stakeholders**: ShopConnect, User, deliverer  **Description**: ShopConnect App syncs all inventory list with ID, name and prices from affiliate stores databases. |
| **Trigger**: Automation of daily sync request |
| **Relationships**:  Includes: None  Extends: None |
| **Normal flow of events**:  1.Check standard time is equal to 6 AM  a. If standard time is 6 AM proceed to next step  2.Connect to stores database  a. If connection successful, proceed to next step  3.Check if inventory with ID, name is available  a. If inventory ID available, proceed to next step  4.Check price for inventory ID  a.If price is available, proceed to next step  5.update Inventory ID, name, price in Application |
| **Exceptional Flow**:  1.a. If time is not 6 AM, continue polling to check time  2. a. If connection is not successful, retry after 5 minutes for 30Sec and update Empty in application for all fields from store if still unsuccessful  3. a. If Inventory ID not available, update NA for ID in application  4.a If Price not available, update price column as NA of Inventory table in application |

**Use Case 5: GPS Location tracking**

|  |
| --- |
| **Use Case Name**: GPS Location Tracking |
| **Primary Actor**: ShopConnect |
| **Stakeholders**: ShopConnect, deliverer  Description : ShopConnect Application finds out nearest available deliverer(registered user) from store through GPS Location tracking to send request |
| **Trigger**: Completion of 'Request from user' use case |
| **Relationships**:  Includes: None  Extends: None |
| **Normal flow of events**:   1. Take StoreID from 'request from user' use case. 2. Map storeID with Store Address. 3. Find nearest possible registered userID from StoreAddress using Google map API within 5 miles.   a. If resistered userID found, End use case. |
| **Exceptional Flow**:  1. a. If no registered userID found, wait for 5 minutes and repeat step 3 for 5 minutes every 10 s and if still unsuccessful, send e-mail to user. |

**Use Case 6: Send Request**

|  |
| --- |
| **Use Case Name**: Send Request |
| **Primary Actor**: ShopConnect |
| **Stakeholders**: ShopConnect, deliverer  **Description** : After finding nearest possible registered user ID, send a request to ID via e-mail and notify on App |
| **Trigger**: Delivery user ID found by 'GPS Location Tracking' use case |
| **Relationships**:  Includes: GPS Location Tracking use case  Extends: None |
| **Normal flow of events**:   1. Fetch delivery user ID. 2. Fetch Store ID, StoreName, StoreAddress, InventoryID,InventoryName,Price, Quantity 3. Send e-mail to delivery user with request details 4. Notify on App |
| **Exceptional Flow**: |

**Use Case 7: Accept Request**

|  |
| --- |
| **Use Case Name**: Accept Request |
| **Primary Actor**: Registered User |
| **Stakeholders**: User, Shopconnect System |
| **Trigger**: Registered user accepts the request and presses the “accept request” button. |
| **Relationships**:  Includes: Email Confirmation.  Extends: None |
| **Normal flow of events**:  1. Press the “accept” button.   1. If accept it, proceed to next step.   2.Shop the listed groceries and deliver goods to user |
| **Exceptional Flow**:   1. a) If not accept it, implement “gps location tracking” use case to find another user. |

**Use Case 8: Email Confirmation**

|  |
| --- |
| **Use Case Name**: E-mail Confirmation |
| **Primary Actor**: ShopConnect System |
| **Stakeholders**: ShopConnect System, Users |
| **Trigger**: System acknowledges that user has pressed “request accept” button. |
| **Relationships**:  Includes: Accept Request  Extends: None |
| **Normal flow of events**:   1. ShopConnect System acknowledges that user has pressed “request accept” button. 2. ShopConnect generates email message with order ID code and order details . 3. ShopConnect sends email message to user who placed the order *with* ID code. 4. ShopConnect sends email message to requested user (delivery person) *without* ID code. |
| **Exceptional Flow**:  None |

**Use Case 9: Transaction Completion**

|  |
| --- |
| **Use Case Name**: Transaction Completion |
| **Primary Actor**: Delivery User |
| **Stakeholders**: ShopConnect System, Users, Delivery Users |
| **Trigger**: Ordering user gives delivery user the Order ID Code given in email message |
| **Relationships**:  Includes: Pay Delivery User, Bill User  Extends: None |
| **Normal flow of events**:   1. Ordering user gives delivery user the Order ID Code given in email message 2. IF delivery user does not deliver items 3. Delivery user enters Order ID Code into ShopConnect system and presses “finish” button 4. ShopConnect system verifies Order ID Code 5. ShopConnect pays delivery user set fee for services via credit to account 6. ShopConnect bills ordering user for items purchased and delivery fee |
| **Exceptional Flow**:   1. a. THEN order is canceled and ShopConnect sends cancellation notice to both ordering user and delivery user |

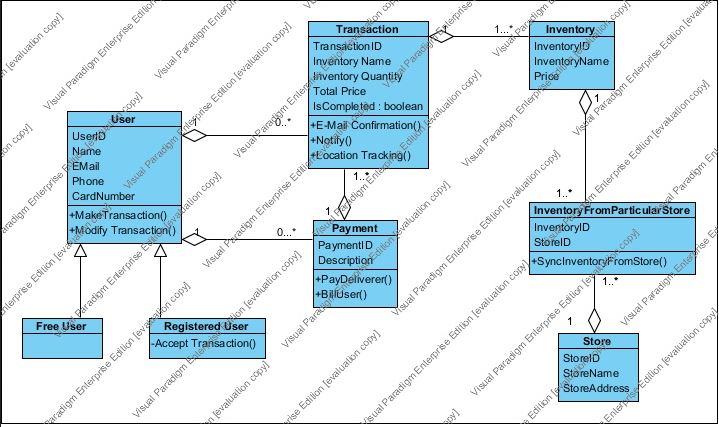
**Use Case 10: Bill user**

|  |
| --- |
| **Use Case Name**: Bill user |
| **Primary Actor**: User |
| **Stakeholders**: ShopConnect System |
| **Trigger**: The groceries have been delivered and the confirmation code is exchanged. |
| **Relationships**:  Includes: none  Extends: none |
| **Normal flow of events**:   1. The delivery user shops the list of groceries and delivers them to the user. 2. After receiving the groceries the user shares the confirmation code with him. 3. The delivery user enters the code into the ShopConnect system. 4. If the code does not match the one sent by the system. 5. The user is then charged for the groceries including the delivery fee |
| **Exceptional Flow**   1. The delivery user is not paid and any discrepancy in the service is verified. |

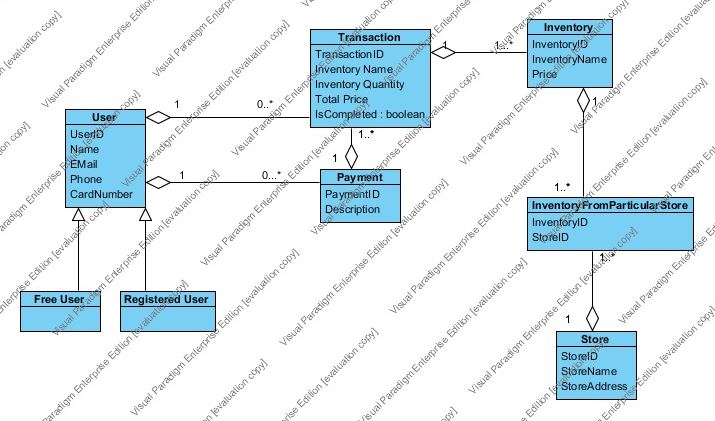
**Use Case 11: Pay Delivery user**

|  |
| --- |
| **Use Case Name**: Pay delivery user |
| **Primary Actor**: ShopConnect system |
| **Stakeholders**: Delivery User |
| **Trigger**: The confirmation code given to the delivery user by free user is entered into the system. |
| **Relationships**:  Includes: none  Extends: none |
| **Normal flow of events**:   1. After receiving the groceries the user shares the confirmation code with him. 2. The delivery user enters the code into the ShopConnect system. 3. If the code does not match the one sent by the system. 4. The user is then charged for the groceries including the delivery fee. 5. The delivery fee is credited to the delivery user’s account. |
| **Exceptional Flow**   1. The delivery user is not paid and any discrepancy in the service is verified. |

CLASS DIAGRAM WITH METHODS

****

CLASS DIAGRAM WITHOUT METHODS

****

DATABASE DESIGN

User( UserID, Name, E-mail, Phone, CardNumber )

Transaction(TransactionID, UserID, PaymentID, Inventory Name, Inventory Quantity, Total Price, IsCompleted)

Transaction.UserID references User.UserID

Transaction.PaymentID references Payment.PaymentID

Payment(PaymentID, UserID, Description)

Payment.UserID references User.UserID

Inventory(InventoryID, TransactionID, InventoryName, Price)

Inventory.TransactionID references Transaction.TransactionID

InventoryFromParticularStore( InventoryID, StoreID )

InventoryFromParticularStore.InventoryID references Inventory.InventoryID

InventoryFromParticularStore.StoreID references Store.StoreID

Store(StoreID, StoreName, StoreAddress)

**Constraints:**

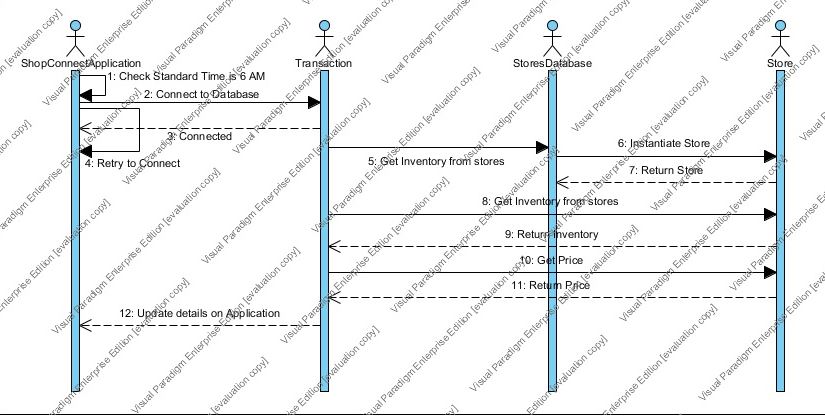
Not NULL constraint: Ensures that a column cannot have NULL values.

Primary Key should not have null values in any part

Foreign Key: The value of foreign key should be equal to primary key of source table

Primary Key: Uniquely identifies each rows in the below table

SEQUENCE DIAGRAM

****

FUNCTIONAL SPECIFICATION DOCUMENT

1. The mobile app lets the free user sign up into the mobile app **SHOPCONNECT** and enters all the required details such as name, phone number, email address, credit card details, etc.
2. People can register to the app for delivering the groceries for which they will be paid for.
3. The mobile app allows the user to order, modify and get the groceries delivered.
4. The App lets the user pick a store of his choice from the list of stores available.
5. The user can then send a request for online delivery of groceries with a list of items needed from the inventory in the app.
6. “ShopConnect” takes up the order and searches for registered users nearby grocery store and sends a notification to all the registered users.
7. The app gives the order with the customer details to that registered user who accepts the request first.
8. The customer is informed that the order has been accepted with a confirmation code and timeframe.
9. The registered user shops the items in the list and makes a payment.
10. The registered user then delivers the groceries to the customer where the actual physical meeting occurs.
11. The Customer exchanges the confirmation code to the registered user who delivered the groceries.
12. The registered user then sends the confirmation code to the ShopConnect who then verifies and credits his account with the delivery charge.
13. The Customer is charged with an extra amount for delivery as mentioned before.
14. During the whole process, the customer can keep track of the person’s location who is delivering the groceries through Global Positioning System.

DATA DICTIONARY NOTATION

**Login** = Username + Password

**Request from user** = [Walmart|Target|Kroger|Costco|Sam’s Club] + 1{Item +Quantity}n +Timeframe given

**SyncInventory** = ItemID + Item name + Price + Store Name + StoreID

**GPS Location Tracking** = Registered User’s Location + Store Location

**Send Request** = Timeframe + Free User’s Details + List of Items + Store name

**Free User’s Details** = Name + Address + Phone number

**E-mail Confirmation to User** = E-mail ID of free User + Confirmation Code

**Bill User** = Free User Name + Card Number + Card Expiry Date + (Security Code)

**Pay Delivery User** = Registered User Name + Account number + Zipcode

**Modify Request** = (Store Name) + (Item) + (Quantity) + (Timeframe)

INTERFACE DESIGN

**LOGIN SCREEN**

****

The User can utilize ShopConnect’s online grocery delivery service after entering the username and password which the user himself created while downloading and installing the app.

**SELECT STORE SCREEN**

****

The above figure is the select store screen. This is the first step in the process where the user gets to choose a store of his choice.

**INVENTORY SCREEN**

****

The above figure is the inventory screen. After choosing the store the user can pick the list of items from the inventory.

**SELECT TIME SCREEN**

****

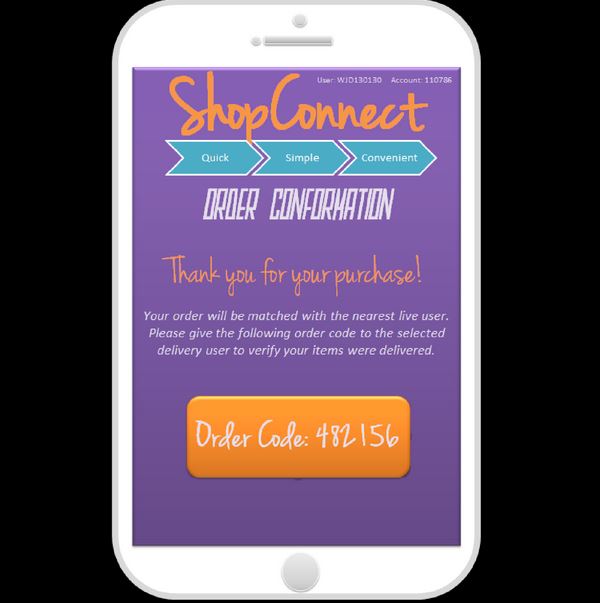
The above figure is the select time screen which is the next step where the user can choose a timeframe.

**FINALIZED ORDER SCREEN**

****

The above screen is the summary of the order.

**FINAL SCREEN**

****

The confirmation code is generated in this step which is also sent to the user through e-mail.

OBJECT DESIGN

1. **Method Name: acceptRequest()**

**Class Name**: Accept request

**Clients(Consumer):** User/Members registered in APP

**Associated Use Cases**: Confirm request

**Description of Responsibilities**: Based on the availability of requests, we need to make sure that there is only one-person take the certain request. We hope no duplicated delivery to same request user. We need to track the request Status to send alerts to people around in case there is no person accept the request.

**Arguments received**: There might be more than one people try to click the Accept button.

**Type of Value Returned**: void

**Pre-conditions**: Requests were sent and listed as Available

**Post-conditions**: Requests were removed from public list to users’ private account.

**Pseudo Code**:

**For User**:

Verify the login information entered by the user

Create a request

IF request is still available then

Delivery user click accept button

Else request will not be seen in the list

IF delivery user successfully accept the request then

System will send confirmation email to users

Else the delivery user will be notified the request is not available

Then exit

1. **Method Name: PickDeliveryUsers()**

**Class Name**: Send request

**Client (Consumer):** User/Members registered in APP

**Associated Use Cases**: Send request/GPS location tracking

**Description of Responsibilities**: To select the proper delivery users by tracking their location. Make sure that request is sent to users that are located less than 5 miles away from requested store.

**Type of Value Returned**: void

**Pre-conditions**: user created request and saved in system

**Post-conditions**: requests are sent to eligible users

**Pseudo Code:**

**For user**:

Verify the login information entered by the user

Choose a store in request

Read store address

Read users address by GPS location

Read driving distance from store to user’s address

IF distance < =10 miles

Send request

Else send no notice

Then exist

1. **Method Name: SyncStoreInfo()**

**Class Name**: Sync Store Info

**Client(Consumer):** Grocery Store

**Associated Use Case**: Sync Inventory from grocery store

**Description of responsibility**: Update inventory information from grocery store in real time. Including item price and stock status. To make sure that the prices is accurate and same in both store and system. To make sure that the items are in stock and requested quality won’t exceed.

**Type of value returned**: void

**Pre-condition**: App inventory info are not updated

**Post-condition**: App inventory info should be sync to store info

**Pseudo Code**:

Read Store inventory info

Read App inventory info

IF Store item price = App Item Price

Keep records on App system

Else change App item price to Store item price

IF Store stock status = App Stock status

Keep records on App system

Else change App stock status to store stock status

1. **Method Name: MakePayment()**

**Class Name**: Make Payment from request user to delivery user

**Client (Consumer):** User/Member registered in App

**Associated Use case**: Bill user/ Pay delivery user

**Description of responsibility**: After transaction completes, we will get confirmation from delivery user and collect paypal information from delivery user. After that, we need bill request

user and collect paypal information from him. The system will transfer exactly billed amount to delivery user from request user

**Type of value returned**: void

**Pre-condition**: Items are delivered but delivery users are not paid

**Post-condition**: Money will transfer from request user’s account to delivery user’s

**Pseudo Code**:

**For user**:

Verify the login information entered by the user

Transaction is completed

IF delivery user does not confirms complement

Keep waiting and track location

Else IF Calculate Item price + tax +extra delivery fee to delivery

And delivery user confirm the amount

Create Invoice and collect paypal infomation

Else recalculate total amount

IF request user does not confirm complement

Keep waiting and track location

Else IF send bill to request user and request user agreed with it

Collect paypal information and transfer money to delivery user’s account

Else recalculate total amount

1. **Method Name: ModifyRequest()**

**Class Name**: Modify request

**Clients(Consumer):** User/Members registered in APP

**Associated Use Cases**: Request from user

**Description of Responsibilities**: In the condition that when the delivery person go to a grocery store but find the goods requestor need are out of stock. He will inform the requestor and the requestor needs to modify the request.

**Type of Value Returned**: void

**Pre-condition**: Requests were remain the same

**Post-condition**: Requests were modified to a new request.

**Pseudo Code**:

IF request goods are not available then

Delivery user click goods not available button

IF request user successfully modify the request then

System will send the new request to be accepted by another user

Else this request will be marked as unfinished

Else exit

SYSTEM CONTROL DESIGN

**Preventive Control Mechanism:**

1. One request can only be accepted by one user.
2. If one request’s status is accepted, the request will be remove from the system.
3. Make sure the price will update in time from the grocery store or website.
4. Make sure the quantity requestor entered will not exceed the stock in the grocery store.

**Detective Control Mechanism:**

1. Users are allowed to revise their requests.
2. Use “PayPal” to ensure the security of payment.
3. If a request exists 10 minutes and no one accept it, system will relocate the user in range and resent it again.
4. If the code entered by the delivery user does not match the one sent by the system to the free user then the delivery fee is not credited to the delivery user’s account.

**Credit mechanism:**

1. If a user accept a request but does not finish it, he will earn a bad credit. If he fails to accomplish his work three times, then his account will be suspended.

**Weekly Timeline**

|  |  |  |
| --- | --- | --- |
| **Weeks** | | **Tasks** |
| 9/2/2014 | 9/9/2014 | Exchange of Phone numbers and created a discussion board |
| 9/10/2013 | 9/16/2014 | All group members came up with different ideas |
| 9/17/2014 | 9/23/2014 | Best Two Ideas :  Online Grocery Delivery, Improving the database and billing systems for WOODBRIDGE GOLF CLUB |
| 9/24/2014 | 9/30/2014 | Finalized on Online Grocery Delivery ( A Mobile app - ShopConnect) |
| 10/1/2014 | 10/7/2014 | Executive Summary, Problem Statement, Context Diagram |
| 10/8/2014 | 10/14/2014 | Discussing all possible UseCases and Design Usecase Diagram  Milestone #1 |
| 10/15/2014 | 10/21/2014 | Document UseCase Description |
| 10/22/2014 | 10/28/2014 | Design Sequence Diagram, Data Dictionary notation |
| 10/29/2014 | 11/4/2014 | Class Diagram (With and without methods) |
| 11/5/2014 | 11/11/2014 | Functional Specification Document, Design Database (Attributes, constraints, primary key)  Milestone #2 |
| 11/12/2014 | 11/18/2014 | Design Software and controlling methods to avoid errors. |
| 11/19/2014 | 11/25/2014 | Design Interface |
| 11/26/2014 | 12/2/2014 | Documentation of the Project |
| 12/3/2015 | 12/4/2014 | Recording of the presentation and uploading to YouTube  Milestone #3 |

**Minutes of Meeting**

|  |  |  |  |
| --- | --- | --- | --- |
| MEETING FOR | SAPM GROUP 6 - 9/17/2014 5.30 pm – 6.30 pm | | |
| MEETING TYPE | SAPM PROJECT MEETING 1 | | |
| ATTENDEES | Swathi Damacherla, Whitney Dodson, Abhay Satish Joshi, Huidan Xu, Guangi Zhang | | |
| DISCUSSION   1. Discussed on different project ideas and their feasibility. 2. The online delivery system and improving the database and billing systems for WOODBRIDGE GOLF CLUB were among the best discussed. | | | |
| CONCLUSION   1. Discussed how complex is each idea. 2. Next meeting to discuss more about the topic and to divide work among each other | | | |
| TASKS | | PERSON RESPONSIBLE | DEADLINE |
|  | |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| MEETING FOR | SAPM GROUP 6 - 9/30/2014 5.30 pm – 6.30 pm | | |
| MEETING TYPE | SAPM PROJECT MEETING 2 | | |
| ATTENDEES | Swathi Damacherla, Whitney Dodson, Abhay Satish Joshi, Huidan Xu, Guangi Zhang | | |
| DISCUSSION   1. Discussed about the scope of the problem. 2. Started with the context diagram 3. Noted down the possible usecases. | | | |
| CONCLUSION   1. The work is shared among the group members. | | | |
| TASKS | | PERSON RESPONSIBLE | DEADLINE |
| PROBLEM STATEMENT | | ABHAY | 10/7/2014 |
| EXECUTIVE SUMMARY | | WHITNEY | 10/7/2014 |
| CONTEXT DIAGRAM | | SWATHI | 10/7/2014 |

|  |  |  |  |
| --- | --- | --- | --- |
| MEETING FOR | SAPM GROUP 6 - 10/7/2014 5.30 pm – 6.30 pm | | |
| MEETING TYPE | SAPM PROJECT MEETING 3 | | |
| ATTENDEES | Swathi Damacherla, Whitney Dodson, Abhay Satish Joshi, Huidan Xu, Guangi Zhang | | |
| DISCUSSION   1. Discussed about the use case diagram and all the persons and the stakeholders involved in the functioning. 2. The work to be done by the deadline is discussed. | | | |
| CONCLUSION   1. The work is shared among the group members. | | | |
| TASKS | | PERSON RESPONSIBLE | DEADLINE |
| USE CASE DIAGRAM | | SWATHI | 10/14/2014 |
| USE CASE DESCRIPTION | | SWATHI, ABHAY, WHITNEY, HUIDAN, GUANGI | 10/21/2014 |

|  |  |  |  |
| --- | --- | --- | --- |
| MEETING FOR | SAPM GROUP 6 - 10/21/2014 5.30 pm – 6.30 pm | | |
| MEETING TYPE | SAPM PROJECT MEETING 4 | | |
| ATTENDEES | Swathi Damacherla, Whitney Dodson, Abhay Satish Joshi, Huidan Xu, Guangi Zhang | | |
| DISCUSSION   1. Discussed about all the classes, methods and relationships for the classes to be included for the class diagram. 2. Discussed about the sequence of steps involved in ordering and delivering the groceries. 3. The data involved in each of these steps. | | | |
| CONCLUSION   1. The work is shared among the group members to meet next week’s deadline. | | | |
| TASKS | | PERSON RESPONSIBLE | DEADLINE |
| SEQUENCE DIAGRAM | | ABHAY | 10/28/2014 |
| DATA DICTIONARY | | SWATHI | 10/28/2014 |
| CLASS DIAGRAM | | ABHAY | 11/4/2014 |

|  |  |  |  |
| --- | --- | --- | --- |
| MEETING FOR | SAPM GROUP 6 - 11/4/2014 5.30 pm – 6.30 pm | | |
| MEETING TYPE | SAPM PROJECT MEETING 5 | | |
| ATTENDEES | Swathi Damacherla, Whitney Dodson, Abhay Satish Joshi, Huidan Xu, Guangi Zhang | | |
| DISCUSSION   1. The attributes, primary keys and foreign keys were discussed. 2. The errors that might occur in the mechanism were discussed. 3. The specifications of the functions were discussed. | | | |
| CONCLUSION   1. The work is shared among the group members to meet next week’s deadline. | | | |
| TASKS | | PERSON RESPONSIBLE | DEADLINE |
| FUNCTIONAL SPECIFICATION DOCUMENT | | SWATHI | 11/11/2014 |
| DATABASE DESIGN | | ABHAY | 11/11/2014 |
| SOFTWARE DESIGN | | HUIDAN | 11/18/2014 |
| CONTROL METHODS | | GUANGI | 11/18/2014 |

|  |  |  |  |
| --- | --- | --- | --- |
| MEETING FOR | SAPM GROUP 6 - 11/18/2014 5.30 pm – 6.30 pm | | |
| MEETING TYPE | SAPM PROJECT MEETING 6 | | |
| ATTENDEES | Swathi Damacherla, Whitney Dodson, Abhay Satish Joshi, Huidan Xu, Guangi Zhang | | |
| DISCUSSION   1. Discussed about how the app looks like including the icon, background and the functions available to the user. 2. The date on which the presentation has to be recorded is decided. | | | |
| CONCLUSION   1. Work is shared among the group members to meet next week’s deadline. | | | |
| TASKS | | PERSON RESPONSIBLE | DEADLINE |
| INTERFACE DESIGN | | WHITNEY | 11/25/2014 |
| PROJECT REPORT | | SWATHI | 11/2/2014 |
| PRESENTATION | | WHITNEY | 11/2/2014 |
| VIDEO OF PRESENTATION | | ABHAY, GUANGI, HUIDAN, SWATHI, WHITNEY | 11/4/2014 |