Individual assignment

Introduction

This document will describe the functional requirements of the shop-navigation system as well as a documentation for the electronic prototype and a proposed testing methodology.

Use case (copy)

Follow Shopping Route

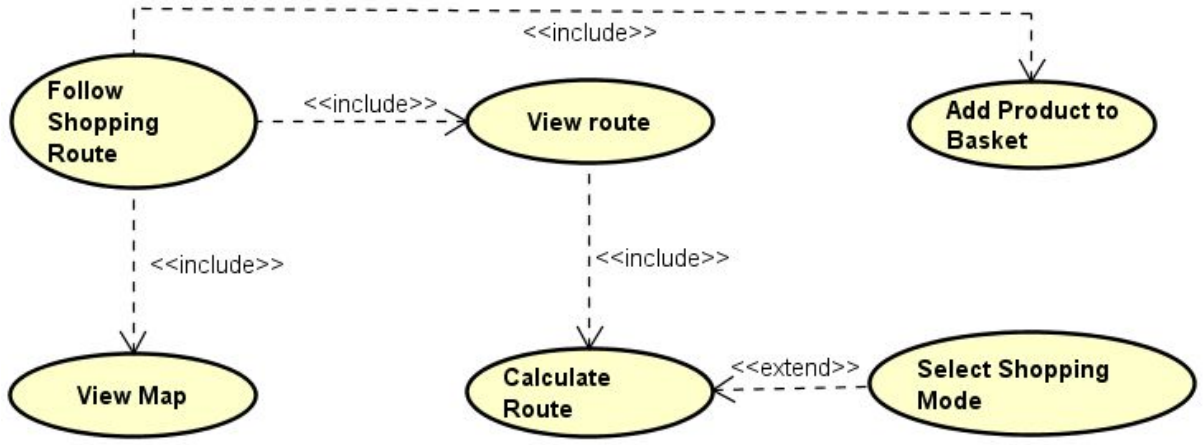

Figure 3 *- Lower level Use Case Diagram (Follow Shopping Route)*

ID and Name:	UC-2 Follow Shopping Route
Created By:	Jacob, Peter, Thomas
Primary Actor:	Customer
Secondary Actors:	N/A
Description:	The customer follows the shopping route based on the previously created shopping list. The customer picks up desired items.
Trigger:	Customer decides to start a shopping trip with the route-planner.
Preconditions:	PRE-1: Customer has a registered account. PRE-2: Customer has a smartphone with internet access. PRE-3: Customer has installed the route-planner app. PRE-4: External services are available.
Postconditions:	POST-1: Customer has seen the route and followed it.

	POST-2: Customer has obtained the desired products on the shopping list. POST-3: Customer ends up at the payment area.
Normal Flow:	**Following route** 1. The customer enters the supermarket. 2. Customer selects shopping list and tells system to start navigation for this list (in *Follow Shopping Route*). 3. System provides a route (in *Calculate Route*). 4. Customer follows the provided route and adds products to basket (in *Add Product to Basket*).
Alternative Flow:	**1.0 Following route with reaction to advertisements** 1. Customer enters the supermarket. 2. Customer selects shopping list and tells system to start navigation for this list (in *Follow Shopping Route*). 3. System provides route. 4. Customer follows the provided route and adds products to basket (in *Add Product to Basket*) 5. While the Customer is shopping, the system shows advertisements to the Customer. Then the customer: a. Reacts to the advertisement (Use Case in top-level diagram, so not explained here). b. Or: ignores the advertisement and continues regularly. 6. Customer *Checks-out* and leaves the store.
Exceptions:	**E1 Empty Shopping list** 1. The app notifies the user that their shopping list is empty. 2. Customer chooses to: a. Cancel the shopping trip and leave. b. Just view the map.
Priority:	High
Frequency of use:	Very often

Functional requirements

Requirements:

1. Shopping route
    1.1. The system shall display a route as soon as the shopper enters the shop
    1.2. The shopper shall be able to get a shopping route which contains all the products.
    1.3. The shopper shall get a new route if they do a dtour.
    1.4. The system shall navigate the user to the cash registers if there are no more items on the shopping list.
2. Item management
    2.1. The user shall get a popup if an item has been added to their basket
    2.2. The system shall remove the item from the shopping list if it has been added to the basket.
    2.3. The system shall add the promotional items to the shopping list if the user presses yes.
3. Representation
    3.1. The system shall mark the position of the items,on the shopping list, on the route.
    3.2. The system shall mark the facing of the shopper on the map.
    3.3. The user shall get a popup if they're near one of the items on their shopping list.
    3.4. The user shall get a popup if there is a suggested item nearby.

Description of the functional requirements

Because the Follow Shopping route has mostly features related to itself and only including retrieving the shopping list at the beginning from another use case. Most features are only part of follow shopping route namely: 1.1, 1.2, 1.3, 1.4, 2.1, 3.1, 3.2, 3.3, 3.4. The only two that are also included in other use cases are 2.2 and 2.3. These include the shopping list use case as well.

On an even lower level they implement these use cases:

1.1 - Calculate route & View route
1.2 - View route & Calculate route
1.3 - Calculate route
1.4 - Calculate route
2.1 - Add Products to Basket
2.2 -Add Products to Basket & Calculate route
2.3 Calculate route
3.1 View map & View route
3.2 View map & View route
3.3 View route
3.3 View route

The functional requirements can be clearly be divided into three categories: Shopping route, Item management, Representation.
The shopping route category includes everything logistical based on the behaviour and how the system displays the route.
The item management category includes everything that the system has to do adding and removing items during the shopping. Adding suggested items and removing them when not necessary.
The representation category describes how items and the positional markers are placed and modified on the screen.

The prototype

I chose to implement a simple two dimensional graphics based prototype that demonstrates who the prototype works but not how it should look. The functional requirements are implemented for a randomized but limited shopping list. The pop ups are for a few items only but demonstrate how they'll work. I had to choose an interactive non-static prototype because taking a dtour adding promotional items cannot be fully demonstrated in a static prototype.

This implementation was made using C++17 using standard libraries and the system library.

The libraries used are:

1. iostream
2. conio.h
3. locale
4. vector
5. tuple
6. fstream
7. algorithm
8. iterator
9. random
10. stdlib.h
11. time.h
12. io.h
13. fcntl.h
14. windows

The classes Map stores every static element and also helps generate the route. It's functions are:

1. Placebox
2. Placemarker
3. Placeitem
4. Placeroute
5. Calcroute
6. Placedoor
7. Getscreen
8. Printscreen
9. Clearroute
10. Drawitem
11. place list
12. Draw_route
13. Dist
14. Check_route
15. Clearroutes

Placebox
This function places a box onto the map representing walls or the aisles in the shop.
Placemarker
This function places the marker representing the person onto the map.
Placeitem
This function places the characters that represent the item that needs to be picked up by the shopper.
Place route
This is a helper function that places a line onto the map starting at a given point and having a certain length.
Clacroute
This function draws a route between two points.
Place door
This function places a door at a given point.
Drawitem
This function draws out the pop ups for the suggested items.
Placelist
This function takes a list of items and places them onto the map
Draw_route
This function calculates and draws the route for a given shopping list.
Dist

Both dist function calculate the manhattan distance between two points one takes as arguments items from a shopping list.
Ckeckroute
This function checks whether a route has obstacles on it or not.
Clearroutes
This function clear all the routes from the map.

The main part of the program controls the movement and and pop up mechanisms. The Box class is how the rectangles are represented.

Controls

In the application you can move with W,A,S,D or UP, LEFT,DOWN,RIGHT and accept the popups with Y and deny them with N and Exit with ESC. Because it was confirmed that it’s not a problem the application only works on Windows. Compiled with Windows SDK 10.0.17134.0.

Test plan

Testing the influence of this system without setting it completely up is tricky that is why I chose a controlled environment. In a controlled environment the application won’t be using the positioning system, instead another person will simulate the person's movement. This way the test subject won’t be affected, but there is no need to install the system for testing purposes. The test will include an almost complete application without the positioning system.

The purpose of the experiment is to measure how easy it is for a shopper to use the system and how much easier it is to do the shopping with the application. The easiness of usability will be measured with a questionnaire which will also include customer satisfaction and suggestion fields. The improvement in easiness of the shopping will be measured in average shopping time. To be able to compare average shopping times there has to be a control group to compare to with the same diversity as the test subjects.

The usability testing will include a given task which is to collect the items on the given shopping list and also a questionnaire. The observational part of the experiment includes recording the movements of the user and also the fluidity of it

to be able to determine the level of comfort/discomfort of the user whilst using the application and also how much help do people require, to find certain “hidden” (hard to find) items, from the staff.

The group of choice for this test are people who are likely to have own a smart device and who is old enough to do a non-trivial shopping,more than 5 items. This group in my opinion are people between 10 and 65.

The measurable outcomes of this test are a customer satisfaction index obtained by the questionnaire, an improvement quotient between the app users and the controlled group obtained from observational data. For these to be measurable we require a controlled environment otherwise the data can be inconsistent. The experiment can be affected by other shoppers not participating in the experiment.