Mobile Software Project

ShopNavigator

Pascal Niedermeyer  
Tobi Reinert  
Rok Virant

Mobile Project work on software engineering

December 2018

table of contents

[1 Describtion 3](#_Toc533039687)

[2 USER GUIDE 4](#_Toc533039688)

[3 LEARNED LESSONS 5](#_Toc533039689)

[4 PROJECT WORK HOURS AND TASK MANAGING 6](#_Toc533039690)

[5 CLASS DIAGRAM 7](#_Toc533039691)

[6 APPLICATION SCREENSHOTS 8](#_Toc533039692)

# DescriPtion

The project work is divided into three main categories on which the applications rely. The first one is application functionality, which includes the basic constructors of the application as well as the algorithm. Within the design and visual representation, we created a layout for the application which represents the actual blueprint of a Lidl shop (in our case we used the Lidl shop in Hervanta). The third categories are based on storage and connecting it to the main application. We created a fine example of 50 items in SQLite, which we then connected to the program. Each file consists of four attributes. The first one describes the name of the item and the other two are two X and Y coordinated each representing the location on the shops' blueprint and the other one representing the location on the shelve. The user has the option to view more precise location (the shelf level) of the item. The application is designed on a two-layer architecture, which means that the first layer implanted as the layout of the shop on which we placed the second layer. This layer is representing the coordinate system that provides the navigational algorithm to find the perfect path between the products. On the second layer, the algorithm places an active line which then connects the user and the selected items.

Algorithm:

All coordinates are relative to the percentage of the height and width of the screen. The map will also adjust itself to the screen resolution.

The routing algorithm will receive a two-dimensional array where x and y coordinates are stored ([x|0] [y|1]). The entrance is fixed as well as the cashiers which have one central point.

The Algorithm will now start at the entrance and seek for the closest product according to the distance of x and y. It will then start from the next product and again seek for the closest product.

The algorithm will continue until all products has been sorted.

At the beginning of the drawing algorithm checkpoints will be created. The entrance and the cashiers are checkpoints too. A checkpoint will be placed at each cross of hallways or according to obstacles.

Next the algorithm will search for the checkpoint which is closest to the next product in the array.

Afterwards it will draw a red line from the entrance up to the first checkpoint which is also a fix checkpoint. Next it will draw a line from this first checkpoint on the y axe to the y coordinate of the checkpoint. Then it will draw a line form the current x position to the x coordinate of the checkpoint. Finally, it will draw a line from that checkpoint tot the x position of the product.

At the end the start will become the checkpoint and the algorithm starts over form that checkpoint. If the user presses the next product in order the algorithm will draw a blue line to it like described above.

|  |  |
| --- | --- |
| Used programs | Used languages |
| -Android Studio  -SQLite  -MySQL  -Blender  -JustinMind | -Java  -SQL |

First, we divided the project in three parts. The surface and basis, the routing algorithm and the database with the connection.

Later in the development process, the different tasks melted together and the work was split to the active workers.

# USER GUIDE

On the first page of the application, you can choose the shop, you want to get navigated through. In the current version of the shop navigator, only one shop is available.

Also, to mention, you can go to the settings (currently only demo, planned to change e.g. the design of the map) and the contact page by clicking on the three dots in the upper right corner.

When going forward the next page, you can insert your shopping list. You do this by writing the needed products in the text field and adding them by clicking on the “Add” Button. It’s suggested to use the proposed results, otherwise a typo can prevent you from adding the item. If this would be the case, a small beacon would show you, that the item is not available. The items for the autocompletion are directly from the database, this way we can be sure, that the proposed items are really available, even after changing the database (in case of multiple shops). You need to add at least one item to your list, otherwise another beacon would show you, that you can’t go forward to the navigation without adding an item.

All the used strings are saved in the “strings.xml” file. This way we can easily add new languages and find misspellings.

On the navigation page, you can see the earlier selected items as a dot on the map and the suggested route in form of red lines. By clicking on them, the way to the next item is getting colored blue. As a small tip: You can use this function for identifying the next item, in the case it isn’t obvious. Therefore, you click on the next possible items and the one that gets blue, is the optimal. The route always starts at the entrance and ends at the middle cashier. You can exit the navigation by clicking the “Exit” button and thereby return to the front page.

# LEARNED LESSONS

Java is a language which is pretty well known by most of the workers though each worker focused on different parts of the application. Some focused on the implementation and creation of the algorithm on which the program is based on, the others did the construction of the main bases of the program and the other created the database. We learned how to use Android Studio and other programs to build this application.

Time management was crucial important for this project. Because of that schedules were made as well as internal and official deadlines. To improve the development of the application several team meetings were arranged, where issues have been discussed, the next steps planned and the main algorithms adjusted to make them work with the other parts of the application.

During the process we faced the problem of not gathering all team members for the meetings or working sessions or even receiving answers from him. The progress was delayed several times because deadlines have not been respected as well as the spirit of working in a group. These issues and delays have been carried and solved in night shifts and long working sessions by the active workers which is displayed in the ‘Project work hours and task management’.

Over these issues we learned that communication is the most important thing in a project and it can become a huge obstacle if not fixed.

# PROJECT WORK HOURS AND TASK MANAGING

|  |  |  |
| --- | --- | --- |
| Members | Individually work / hours | Group work / hour |
| Pascal | 100 | 50 |
| Tobi | 100 | 50 |
| Rok | 15 | 5 |

|  |  |
| --- | --- |
| Members | Task per member |
| Pascal | * Gui design * Implementing the map in the program * Basic program * Documentation |
| Tobi | * Designing the shop layout * Navigation Algorithm * Routing Algorithm * Documentation |
| Rok | * Creating a Database and implementing it to the program. * Documenting * Help on Gui design * Help on Algorithms |

# CLASS DIAGRAM

# APPLICATION SCREENSHOTS

# TEST REPORT

There have been multiple tests where the resolution of the map, the routing algorithm as well as the drawing algorithm has been tested.

It was mandatory that the drawing algorithm and the map where synchronized because otherwise the lines would cross the shelfs or be out of the map.