**Project supermarket - Part 3**

* Project delivery
  + On **Sunday, 22 May, before 11:59 pm,** you must upload your project in order to receive points for PE of Java.
  + You also upload a txt-file with all the code of the MainController and the html-files.
* Remember:

20% Permanent Evaluation (PE): submitted project

80% Exam in June

* + - 25% theory -> closed book
    - 75% practice -> open book with internet access
      * 1 question = extension on project
      * Other question not linked to project

If your uploaded project is incomplete, you may continue to work on it before the Java exam so that you can be better prepared. However, this version does not count for PE.

* **Please note**: we know that a lot of information/code is exchanged between students to help each other. This is absolutely no problem when doing the exercises. However, the project has PE points, so must be made **INDIVIDUALLY/PERSONALLY**.
* So if you use/copy someone else's code or you help someone with the code, we can no longer assess someone’s knowledge/capabilities individually. This is a form of fraud: both for the person copying and for the person who has copied. See OER (Education and Examination Regulations). Make sure that you make your project individually and do not "share" your code with other students. So do not put your code on "social media/shared cloud" to help each other and do not use it there. The Java teachers are there to help everyone...

1. **Saving data**

In the MainController you will find (in comments) three private attributes:

private ArrayList<Staff> **staffArrayList**;  
private ArrayList<Customer> **customerArrayList**;  
private ArrayList<Supermarket> **supermarketArrayList**;

Take these lines out of comments **and** do the same for the methods fillStaffMembers(), fillCustomers()and fillSupermarkets().

Now create one new method in which you fill the 3 attributes: staffArrayList**,**  customerArrayList**,** supermarketArrayList**.** How? By calling the methods using the methods fillStaffMembers(), fillCustomers()and fillSupermarkets(). Make sure that this new method is implemented from the start of the application (see the "saving data" part of the presentation Lesson 8 MVC part 2).

**In the fillCustomers-method add a new customer** with your name and year of birth. Add also some products to the shoppingList of this customer. Don’t forget to add this new customer to the customers ArrayList.

1. **Showing customers and staff**
2. Provide a link on the home page that takes you to a page (**5\_xxx,** replace xxx with a logical name you have to choose yourself) that lists all saved staff members. Use the *toString* method to display this information.   
   Also provide a link to the home page.

Afbeelding met tekst, binnen, schermafbeelding

Automatisch gegenereerde beschrijving

1. Afbeelding met tekst

   Automatisch gegenereerde beschrijvingAlso provide a link on the home page that redirects to a page (**6\_xxx,** replace xxx with a logical name you have to choose yourself) that lists all saved customers. Show the name of these customers, their year of birth, their cardnumber and their shoppinglist. At the end provide also a link to the home page.

Check whether your own name is in this list, since you had to register yourself as a customer.

1. **Saving new staff**

Make sure that a new staff member that is created is also added to the ArrayList of staff members.

1. **New customer extension**

Currently the html-page with prefix “**1\_”** contains three text boxes, a submit button and a link to the homepage.

* Afbeelding met tekst

  Automatisch gegenereerde beschrijvingChange the text on the button:
* Add a list of radio buttons with the names of the supermarkets (from the ArrayList stored in the MainController).   
    
  If you want to select the first radio button by default, you can add this attribute to your input tag: **th:checked="${supermarketStat.index} == 0"**

If you enter MainController via the submit button you have to make sure that:

* + The customer is registered as a customer at the supermarket.   
    Do this as follows: a new customer object is still being created as you coded it before, but now you register the customer at the supermarket (which you clicked on) using the *registerCustomer* method that you have already programmed in the Supermarket class.
  + Add this new customer to the ArrayList of customers

Finally, you also redirect the user to the html-page with prefix “2\_” where you show the data (*toString*) of this customer. You will see that the customer’s cardnumber is no longer -1.



1. **New supermarket and showing supermarkets**

On your home page, put **an extra link 'New supermarket'** which will takeyou to a new html page whose name starts with **7\_.** On this html page you provide:

* + a hyperlink (at the bottom of your page) to return to the home page.
  + a form with which the user can enter the name of the new supermarket.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

When the form is sent (via the submit button) this supermarket is added to the ArrayList with supermarkets and you will be taken to the next html page (**8\_xxx,** replace xxx with a logical name you have to choose yourself). where all supermarkets from this list are shown (and a link to return to the home page).

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Now add to index.html an extra link **List of supermarkets** which will take you directly to the html-page with prefix “**8\_”**.

1. **New departments and showing departments**

Add to your index page **an extra link 'New department’** which will takeyou to a new html page whose name starts with “**9\_”.**

Afbeelding met tekst

Automatisch gegenereerde beschrijving

The 2 dropdown lists contain, respectively, all existing *supermarkets* and *staff members*.

|  |  |
| --- | --- |
| Afbeelding met tekst  Automatisch gegenereerde beschrijving |  |

With the button "Confirm department" you go to the MainController where you create a new Department object based on the entered data.   
You then link this department to the selected supermarket.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Then you go to a page (**10\_xxx,** replace xxx with a logical name you have to choose yourself) on which you show all departments (= already existing departments + the newly registered department) of the supermarket you selected (and a link to the home page).

1. **Error Handling**

Now also make sure that errors are dealt with appropriately: if the user has not indicated a supermarket or a person responsible when creating a new department, send the user to a page **error.html** where you show an appropriate error message with a link to go back to home page.

|  |  |
| --- | --- |
|  |  |

1. Afbeelding met tekst

   Automatisch gegenereerde beschrijving**Supermarket and departments in detail**

Adjust the page (with prefix “**8\_”)** so that the name of each supermarket becomes a hyperlink.

Clicking on one of the supermarket links will take you to the page (**10\_xxx, ,** replace xxx with a logical name you have to choose yourself).

Expand the page (with prefix “**10\_”**) in the following way.

* In addition to the *name of* each department, you now show who is *responsible for* it and whether the department is refrigerated or not. (“No refrigeration” or “Refrigerated department”)
* Also show the photo (if there is any).
* Create or search for an image that can be shown when there is no photo available for a certain department or when an incorrect filename has been entered. You can do this by adding the onerror attribute to the image tag. Your image tag will look like this:

<**img onerror="this**.**src**=**'img/noimage.jpg'" th:src="@{${department.getPhoto()}}"/>**

*where noimage.jpg is the figure you use when the photo is not available.*

|  |  |
| --- | --- |
|  |  |

\*Notice that the Delhaize supermarket contains the extra department that we created in part 6 **New departments and showing departments** of this assignment.

1. **Search department by name**

Provide a **search option** on the home page. Someone can enter the name of an department and if this name is an existing department within one of the supermarkets, you will be taken to a page (with prefix “**11\_”**) where you can see all the information about that department. Be careful: the user has to enter the full name of the department he is looking for, also taken into account the capital characters in the word.

Loop through all the supermarkets and use the existing method,*searchDepartmentByName*in the class Supermarket to create this functionality.

If you find an department with the name you entered, forward the user to a page (**11\_xxx,** replace xxx with a logical name you have to choose yourself) where you show all the details of that department.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

If you have not found an department with that name, send the user to the previously created error page where you will display an appropriate error message with a link to return to the home page.

Afbeelding met tekst

Automatisch gegenereerde beschrijving