CFS2160 Software Design and Development

Assignment 2: Full Program

Student name: Amanuel, Berhe

Student number: U2058463

Student course: CC100

Contents

[Project summary 3](#_Toc65277369)

[Supermarket Checkout Simulation 3](#_Toc65277370)

[Who wants to be a Zillionaire? 4](#_Toc65277371)

[Software Design / Modelling 5](#_Toc65277372)

[Design Overview 5](#_Toc65277373)

[Use Case Diagram 5](#_Toc65277374)

[Class Diagram 5](#_Toc65277375)

[Activity Diagrams 5](#_Toc65277376)

[Sequence Diagrams 5](#_Toc65277377)

[Testing 6](#_Toc65277378)

[Test Plan 6](#_Toc65277379)

[JUnit Testing 6](#_Toc65277380)

[Walk-through 6](#_Toc65277381)

# Project summary

## Supermarket Checkout Simulation

| **Core Requirements** | **Modelled** | **Implemented** | **Tested** |
| --- | --- | --- | --- |
| Include a minimum of two checkouts |  |  |  |
| Include of list of products (for simplicity no more than 10 are required) that can be “scanned” with appropriate product code, description, and price |  |  |  |
| Enable items to be “scanned” using the product code |  |  |  |
| Keep track of the total cost of items “scanned” for each customer |  |  |  |
| Enable goods to be paid for in cash |  |  |  |
| Print out a list of items “scanned” |  |  |  |

| **Extensions** | **Modelled** | **Implemented** | **Tested** |
| --- | --- | --- | --- |
| The facility to have more than two checkouts in the system i.e. an arbitrary number of checkouts |  |  |  |
| The facility to have different types of checkouts such as 10 items or less, cash only etc. |  |  |  |
| Enable the system to keep track of total payments made by all customers |  |  |  |
| The facility to pay for goods by credit card |  |  |  |
| A loyalty card |  |  |  |
| Stock control |  |  |  |

## Who wants to be a Zillionaire?

| **Core Requirements** | **Modelled** | **Implemented** | **Tested** |
| --- | --- | --- | --- |
| Allow two players to play the game |  |  |  |
| Include a selection of questions that players could be asked |  |  |  |
| Allow a player to choose the category of each question (for simplicity have three categories, one of which should be general knowledge) |  |  |  |
| Display a question at random from the available questions |  |  |  |
| Provide a facility for the player to enter the answer to each question |  |  |  |
| Inform the player whether the answer is correct, or not |  |  |  |
| Keep track of the amount of ‘money’ credited to each player |  |  |  |
| Implement the help facility ‘Ask the Public’ |  |  |  |
| Implement the help facility ‘Half-and-half’ |  |  |  |
| Allow a player to win |  |  |  |

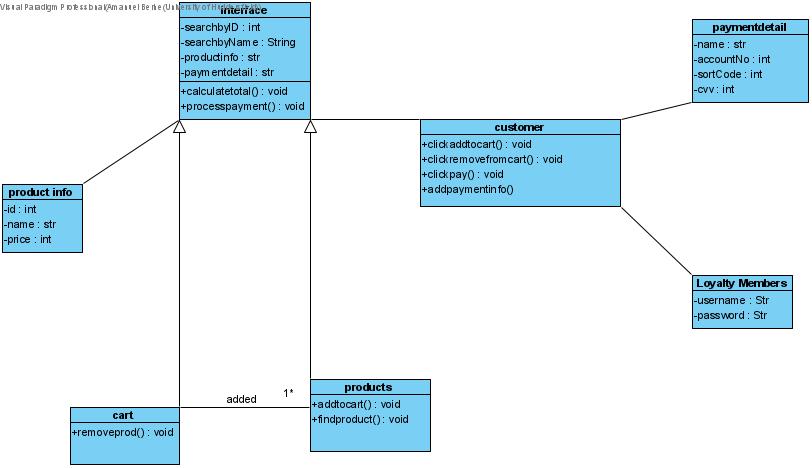
| **Extensions** | **Modelled** | **Implemented** | **Tested** |
| --- | --- | --- | --- |
| Allow more than two players to play the game |  |  |  |
| Facility for easily adding extra questions |  |  |  |
| Add another help facility, e.g. ‘Give us a Clue’ |  |  |  |
| Ensure that the same question is not asked twice in any one game |  |  |  |
| Include the use of sound or animation to make the game more interesting |  |  |  |
| Display the ratings for ‘Ask the Public’ in graphical format e.g. a bar chart |  |  |  |

# Software Design / Modelling

## Design Overview

[Place here your brief notes (maximum of 500 words) to explain your design]

## Use Case Diagram



[Place here your Use Case Diagram]

## Class Diagram

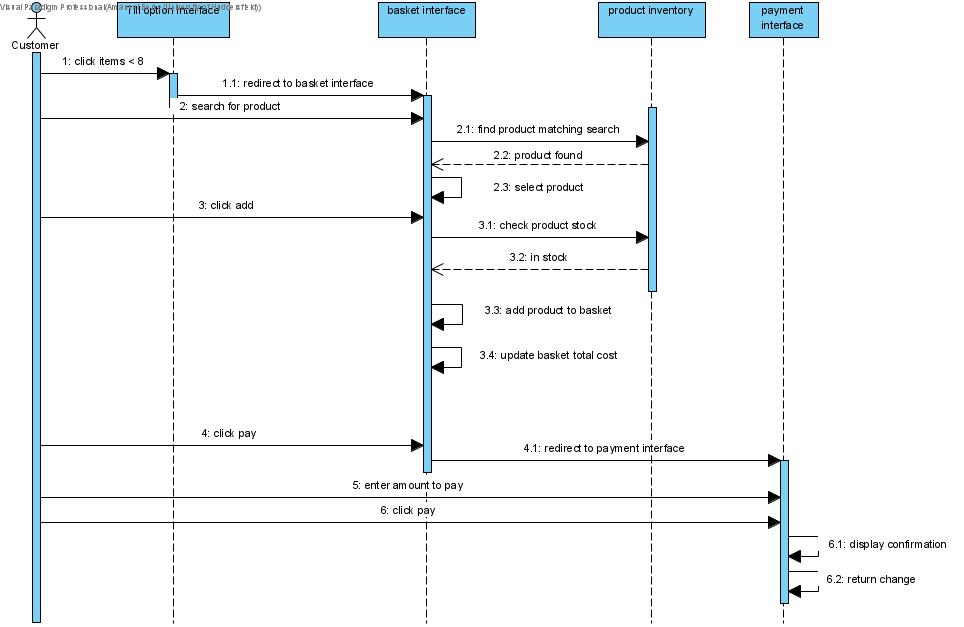


[Place here your Class Diagram]

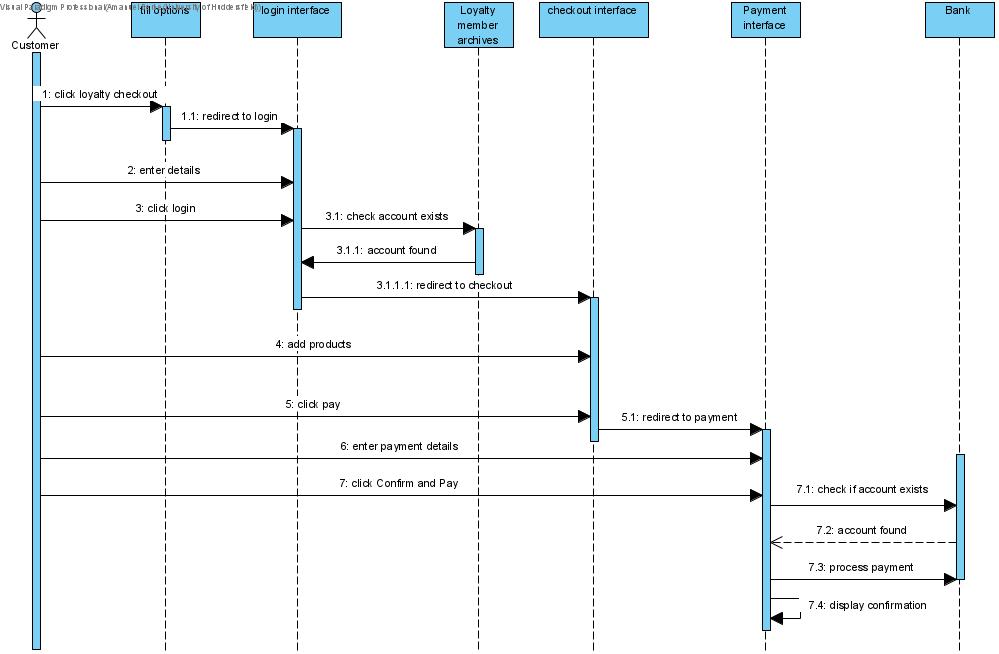
## Activity Diagrams

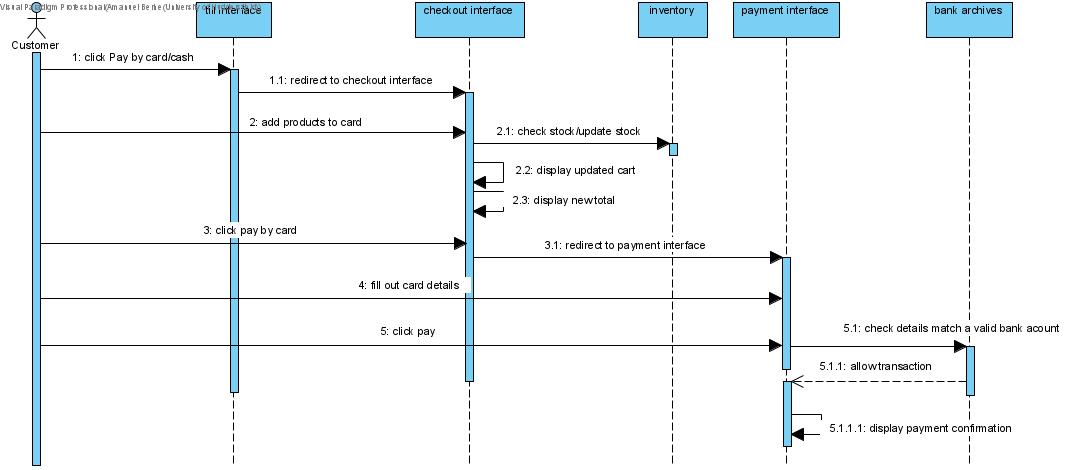
[Place here your Activity Diagrams (multiple diagrams)]

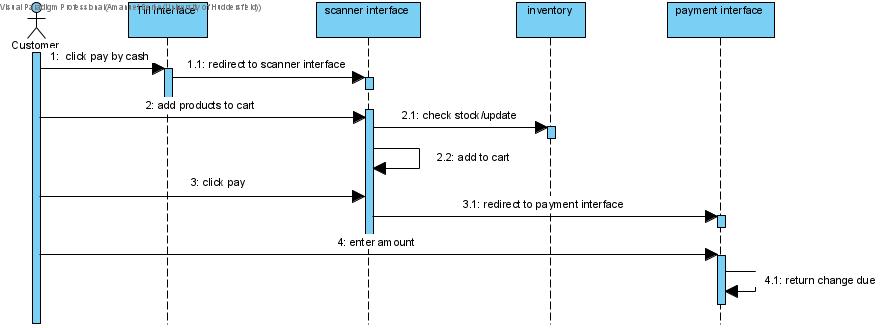
## Sequence Diagrams



[Place here your Sequence Diagrams (multiple diagrams)]







# Testing

## Test Plan

| **Test Description** | **Input** | **Expected Result** | **Actual Result** | **Evidence / Screenshot** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## JUnit Testing

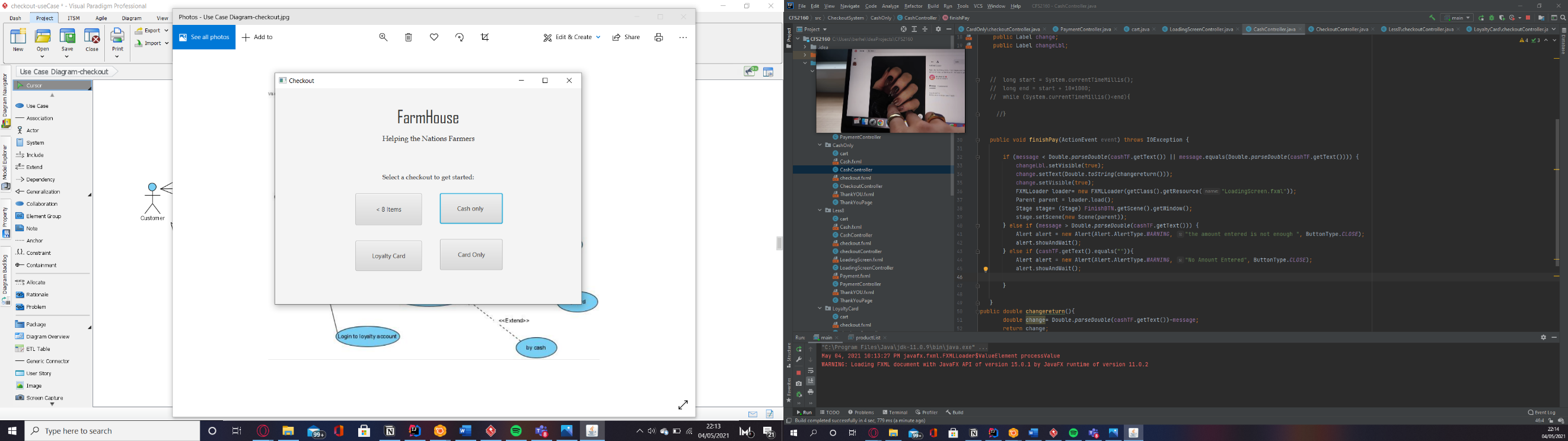
Classes tested using JUnit:

* Class
* Class
* Class

## Walk-through

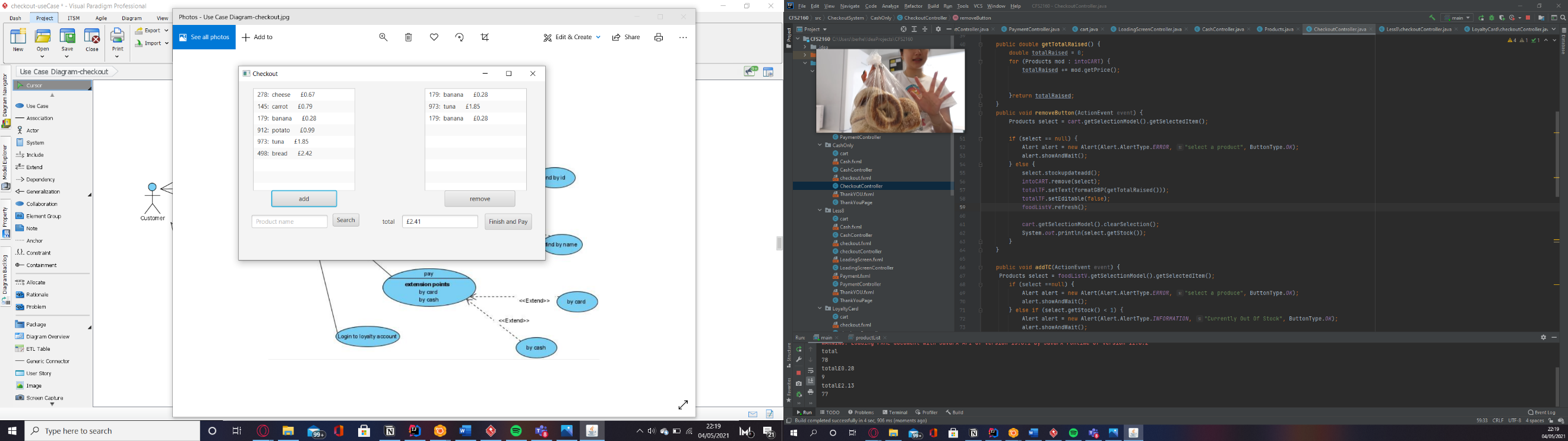
[Place here your walk-through of the system showing the features implemented]

Cash checkout



1

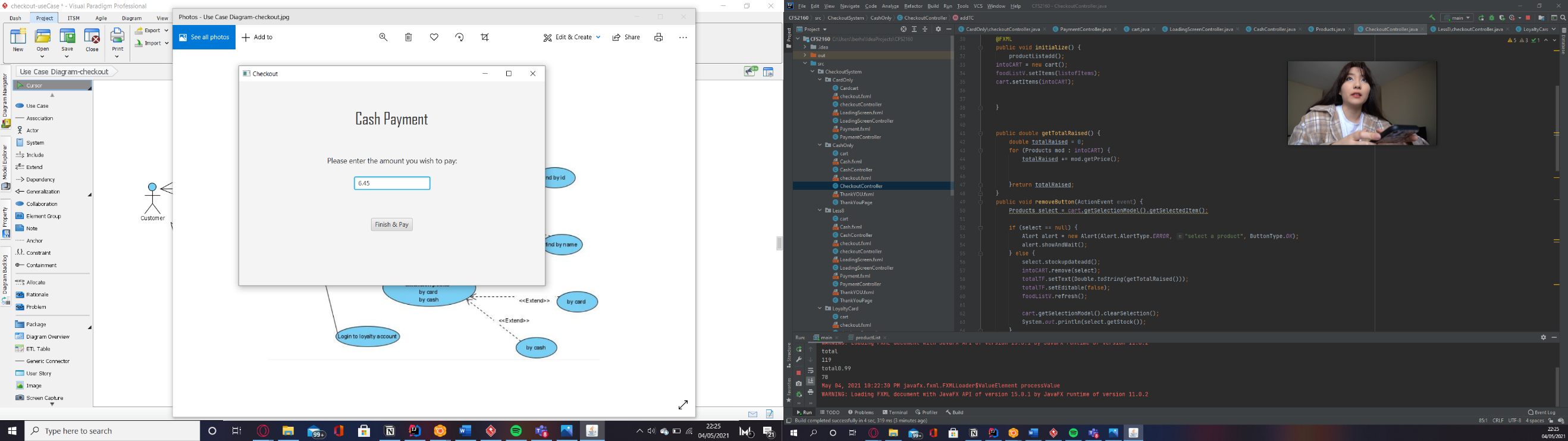
this will redirect the customer to the checkout interface

2.

2

After selecting a product from the left product list clicking add will add the product to the cart on the left.

After selecting a product, clicking remove will remove the item from the cart and update the total



4

3

After clicking pay it will display the change

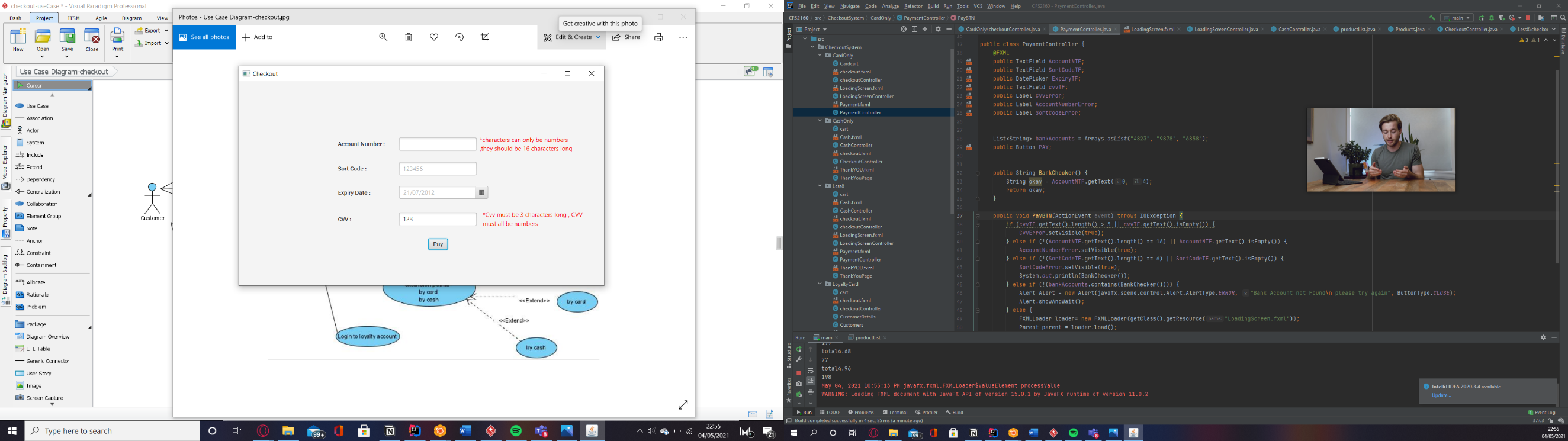
Here the customer will enter the amount of cash they would like to pay with

Products can be found using the product code or the name of the product

This total will be updated based on the customers adding or removing products

Card Checkout:

After following steps 1 to 2 in the above example clicking finish and pay will then lead to:

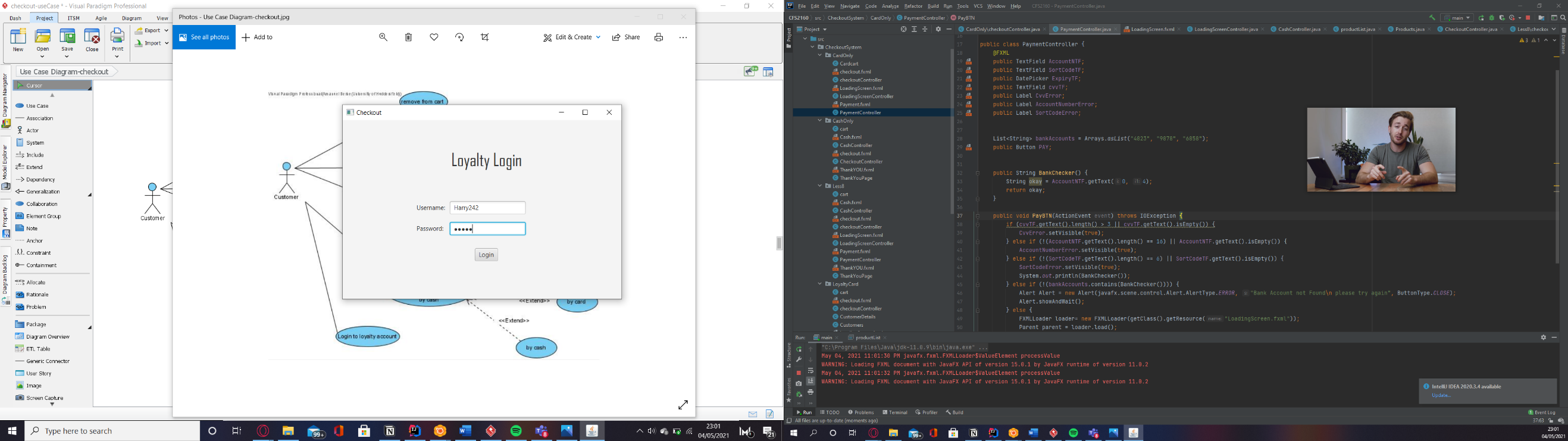


If pay is clicked and parameters are not met a red text will appear next to the incorrect information informing the customer

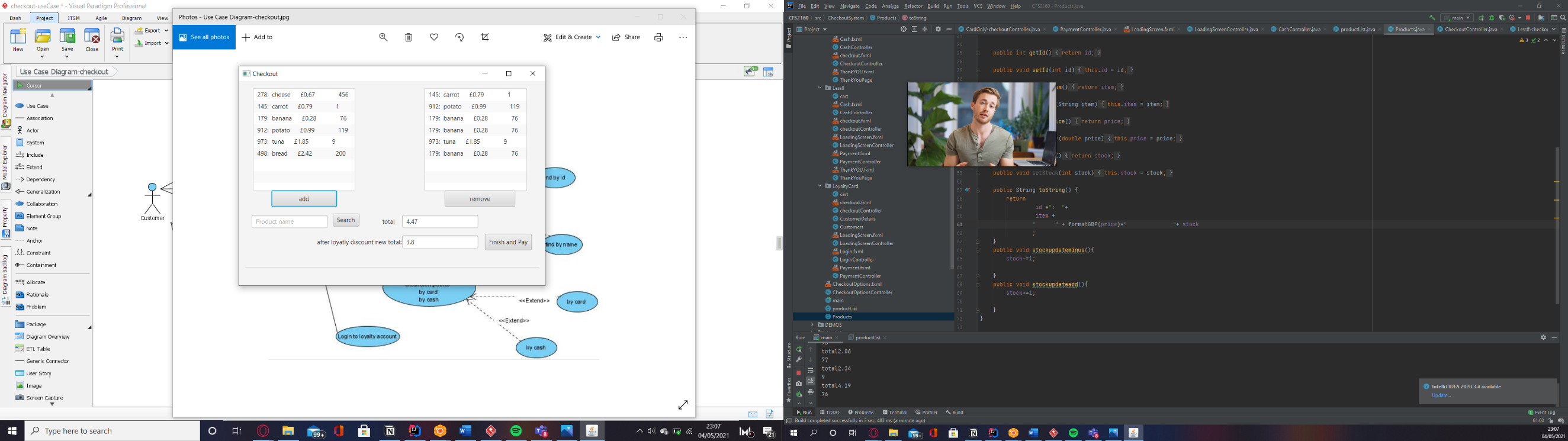
Customer can fill out bank details within the parameters set

Loyalty members :

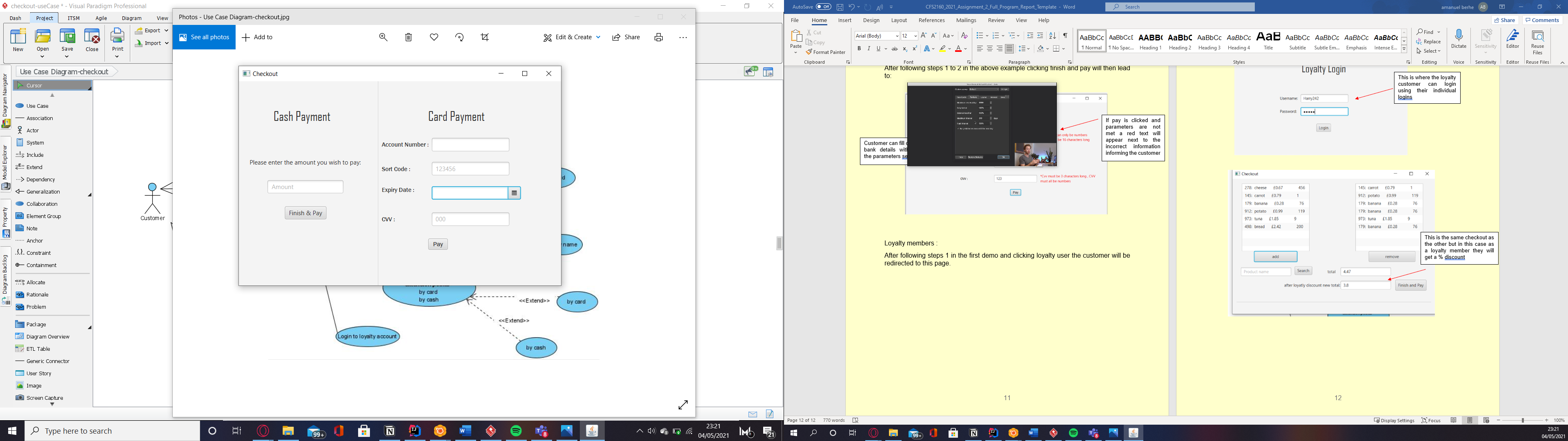
After following steps 1 in the first demo and clicking loyalty user the customer will be redirected to this page.



This is where the loyalty customer can login using their individual logins



This is the same checkout as the other but in this case as a loyalty member they will get a % discount



After clicking pay the customer will then be redirect to the payment section where they can pay by cash or card