



**L-Università ta' Malta**  
Faculty of Information &  
Communication Technology

Department of  
Computer Information  
Systems

## **Petrol Pump Design Project**

David Briffa (376195M), Marjohn Saliba (10803H), David Buhagiar (69903L),  
Olesia Shtanko (0018504L), Timothy Zammit(0112103L)

B.Sc. (Hons) Software Development

---

Study-unit: **User Interface Design**

Code: **CIS2207**

Lecturer: **Dr Colin Layfield**

## FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY


### Declaration

Plagiarism is defined as "the unacknowledged use, as one's own, of work of another person, whether or not such work has been published, and as may be further elaborated in Faculty or University guidelines" (University Assessment Regulations, 2009, Regulation 39 (b)(i), University of Malta).

We, the undersigned, declare that the assignment submitted is our work, except where acknowledged and referenced.

We understand that the penalties for committing a breach of the regulations include loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree programme.


Work submitted without this signed declaration will not be corrected and will be given zero marks.

David Briffa 

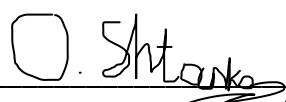
Student Name Signature

Marjohn Saliba 

Student Name Signature

David Buhagiar 

Student Name Signature

Olesia Shtanko 

Student Name Signature

Timothy Zammit 

Student Name Signature

CIS2207 Petrol Pump Design Project

Course Code Title of work submitted

12/12/2022

Date

# Contents

1. Requirements Analysis.....	1-5
2. Guidelines for the interface .....	6-14
3. Prototypes.....	15-20
4. Testing and Evaluation.....	22-24

## Distribution of work:

David Briffa: 20%



---

Timothy Zammit: 20%

---

Marjohn Saliba: 20%



---

Olesia Shtanko: 20%



---

David Buhagiar: 20%



---

# Requirements Analysis

## Overview

The objective of this system is to provide a clear and intuitive interface for purchasing and dispensing petrol at a petrol station. The system includes a manager console that allows employees with appropriate permissions to access the transaction history or change the prices of the fuel available.

Considering that this system would be used in a 24/7 environment, the only downtime would be while a manager was changing the price of fuel. Furthermore, the system prioritizes reliability, as any potential failed transaction may result in the loss of a customer.

Visually impaired drivers account for 2-3% of all drivers, and as such, the interface accounts for this by providing large, distinct buttons complemented by contrasting colours.

The age of potential users was also considered. While the average age for drivers is 48 and technological adaptation has advanced significantly, 7-9% of drivers are over 65 years old, and thus the system was designed to be as simple as possible. This was accomplished using buttons, as the chance for error is significantly decreased when users are limited in what they are able to do. The background for these buttons was set to the selection they are meant to represent, as can be seen in the payment type and cash payment interfaces. Furthermore, in the event of a wrong selection, an undo button was implemented, which takes the user back one frame while maintaining the selections made prior to the last. Finally, a cancel transaction button was also implemented, which takes the user back to the main menu and resets all of their selections.

The client found a prior experience with a pump unnecessarily convoluted, specifically that every client has to use the same interface, and possibly creating confusion when selecting which pump your car is stationed at. It was thus the team's objective to design an interface that would be implemented alongside every pump, and providing each individual customer with their own local interface. In a real-life scenario, all these pumps would be connected to a central database, allowing for easy price manipulation by management, and for all transactions happening on multiple machines to direct all of the data towards the database.

What follows is a breakdown of the individual interface pages and their functions.

## Main Pages

### Main Menu (index.html)

Main menu hosts three buttons labelled petrol, diesel and manager console. Sticking with our objective of achieving a minimalistic design, the team opted to allow users to control every aspect of the system with buttons. This choice minimizes the risk of potential errors or bugs as the user is forced into selecting from pre-defined options. Pressing 'Diesel' directs users to selecting a fuelling method. Pressing the 'Petrol' button then asks the user to select a grade, before also asking for a fuelling method. Manager Console demands a pin, which is '1234', before granting access.

### Manager Console (homeScreen.html)

The Manager console presents the user with two buttons, transactions or set fuel price. Transactions (viewTransactions.html) allows the manager to view the most recent transactions, displaying values such as the cost, date, and the type and amount of fuel purchased. Pressing the set fuel price buttons asks the manager which type of fuel they wish to change (fuelType.html). Petrol further queries about the grade in question (typeOfPetrol.html). Once the selection is complete, both options redirect to the keypad (changeFuelPrice.html). The use of a keypad again limits possible input errors, and upon completion the manager is directed to a confirmation page (successChangedFuelPrice.html), before this page automatically routes back to the manager main menu.

### Fuelling type selection (fixedDynamicPayment.html)

The client requested that users be able to purchase fuel in one of two ways: The first option is to pay for a fixed amount of gas by cash or card, while the second option allows users to swipe their card and pump as much fuel as required. Researching the topic it was discovered that in gas stations that implement the latter system, the maximum value of such a transactions is 125-175eu in order to avoid taxation. It was thus decided that the system should mirror its real life equivalent and as such the cap on our dynamic transactions is 120eu.

Selecting fixed amount directs to a page where the user is asked whether they wish to pay by cash or card (paymentInterface.html). Selecting either option directs the user to the money interface (cash.html). Selecting dynamic directs the user to the card interface (creditCard.html).

### Cash Interface (cash.html)

This interface presents the user with four buttons represented by 5, 10, 20 and 50eu notes respectively. The user may press these buttons repeatedly, increasing the total amount they wish to purchase, which is displayed below. After at least one button is pressed, a 'continue' button becomes available. If the user wished to pay by cash, this represents the user inserting the bills, and will direct to the pumping interface (fuelDisplay.html). If the user wished to pay by card, pressing continue will direct the user to the card interface. Cash transactions are limitless.

### Credit Card Interface (creditCard.html)

If the user chooses Dynamic in the appropriate page or elects to pay by card, they are directed to this interface. The user may swipe their card, represented by a button, and other buttons represent what can happen upon card swipe. Decline allows the user to swipe again, while Accept directs users to the pumping interface. As was mentioned previously, if the user wishes to use the dynamic payment feature, their transaction will be a maximum of 120eu.

### Fuelling interface (fuelDisplay.html)

Once everything is selected and payment processed, the user is directed to an interface displaying real time values. The top value represents the transaction maximum, such as the amount of cash the user has inserted or paid for by card. The second value represents how much of the money has been pumped so far, with the value below it displaying the amount of fuel in litres. The bottom value represents the current value of fuel per litre in euro. Complementing this display is a progress bar that shows how much of the fuel has been pumped. If the user selected fixed amount, once they begin pumping, they are locked in and must complete the transaction, with a button only appearing once the progress bar is full. Users who selected dynamic pumping may pump as much as desired, with a button appearing as soon as they pump anything. This button may be used to complete the transaction, leading to the final page (transaction.html).

### Transaction data (transaction.html)

Displays the transaction details and a farewell message. If the user paid by card, it informs them that their card has been charged. Pressing the main menu button clears the transaction settings and redirects to the front page, allowing the next transaction to begin.

## User Error Handling

- Go back button
  - Present on every page
  - Redirects the user one page back, clearing any stored data related to the associated page
- Cancel transaction button
  - Redirects the user to the main menu, clearing all stored data

## Usability

- Contrasting colours to aid users who are colour-blind
- Sans-serif Fonts to aid users with dyslexia
- Large text and buttons to aid users who are visually impaired
- Use of informative backgrounds on buttons to convey information
- Handling of user errors does not reset entire transaction

## Assumptions

- The software will be used in the EU, and as such will use the Euro currency
- The interface is a touch screen
- The interface is connected to credit card services
- The interface accepts cash through a cash slot
- The interface accepts card through a swiping contraption

## Use Case Diagram

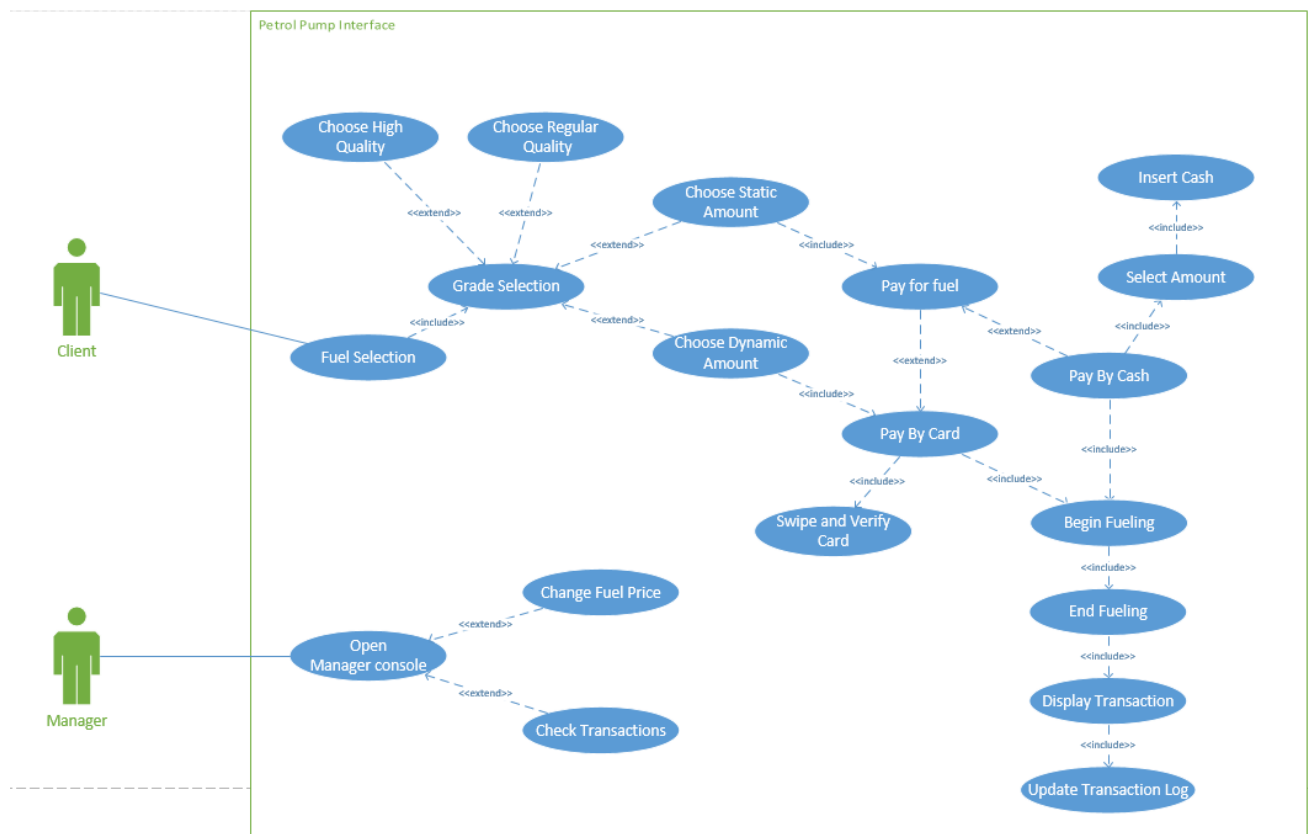


Figure 1



# Guidelines for the Interface

For this assignment, the standard User Interface Design guidelines and procedures were followed with utmost attention. These guidelines and procedures are followed by large and successful companies such as Amazon, Apple and Google. Some of these guidelines and procedures which we followed in our system can be found below.

## 1. Minimizing the problems related to data input as much as possible

Within this project, input from the user is mainly taken by the use of large selection buttons as shown in Figure 2. As a result, the input is limited to the certain options presented to the user only. Therefore there is less room for input errors from the user's side.

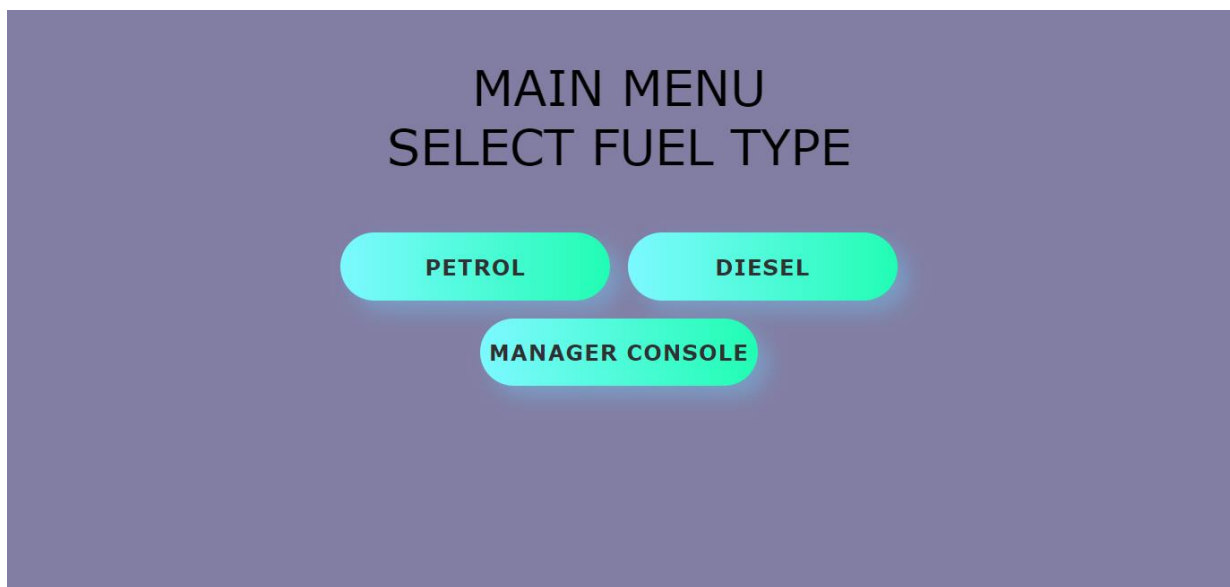


Figure 2

However, there are three scenarios in which the user is requested to make an input with the use of a keypad. These are: inputting the amount of money worth of fuel to fill the vehicle, inputting the passcode to enter the Manager's Console and entering a new fuel price inside the Manager's Console. With this input method, various input errors can take place, but the system was constructed and tested in order to avoid such scenarios.

### Inputting the passcode to enter the manager's console

In this scenario, the manager has to input the 4-digit passcode. The correct passcode for our system is 1234. Input error avoidance was taken care of by limiting the manager to 4-digit passcodes, instead of allowing the manager to have passcodes with varying lengths and different input characters from numbers. By using a keypad, and assuming that the system is touch screen, letter characters that would result in exceptions are avoided altogether. A four digit pin contributes to the security of the manager console due to the many combinations possible. Naturally, an unauthorized user is denied access, since an incorrect passcode will leave the user unable to access the Manager's Console.

### Changing price of a fuel by the manager

Similar to when inputting the pin to access the Manager's Console, a keypad with limited digits (0 - 9) is also displayed when the manager is going change the price of a particular type of fuel. As a result, the system has full control on the maximum price change possible, which in this implementation is €99.99. This can help the manager notice if he enters an incorrect fuel price by mistake, for example €101.01.

The decimal point ('.') for the change in fuel price is visible automatically without the manager having to input it himself. This feature does not allow the manager to input more than one or an invalid type of point character. Thus, it also acts as a placeholder to show the correct format for the new fuel price. The last digit entered by the manager is the last digit of the fuel price.

Example: the manager is going to change the price of diesel from €1.23 to €1.26. This is the way it should be done as displayed in Figures 3 to 5.



Figure 3: Pressing digit 1 from the keypad

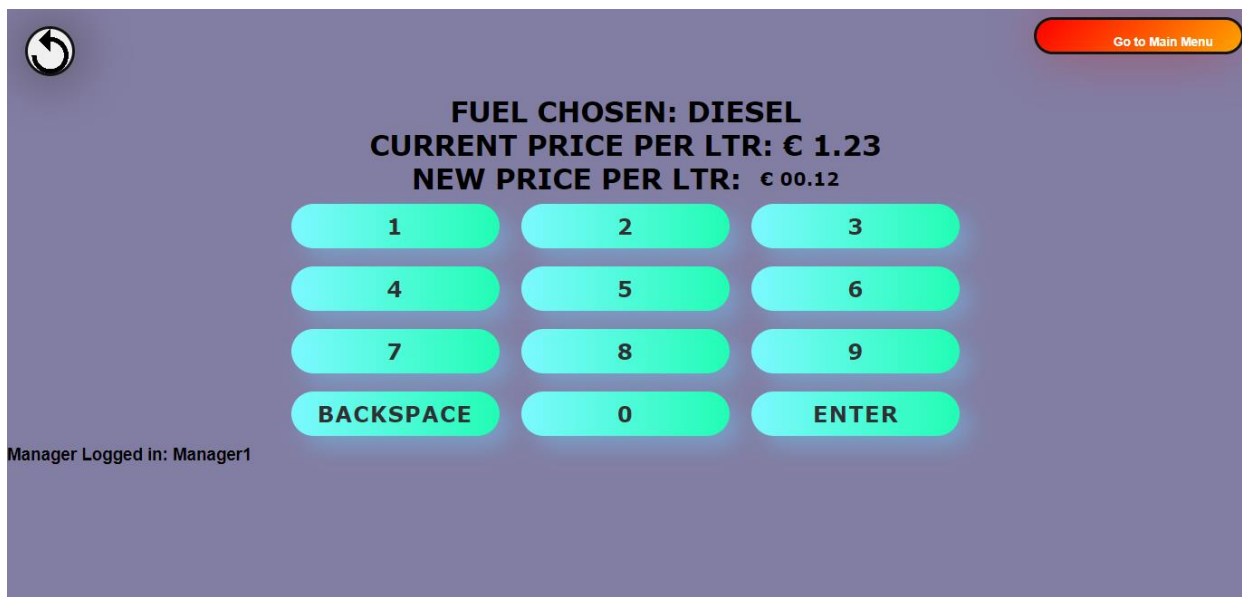


Figure 4: Pressing digit 2 from the keypad

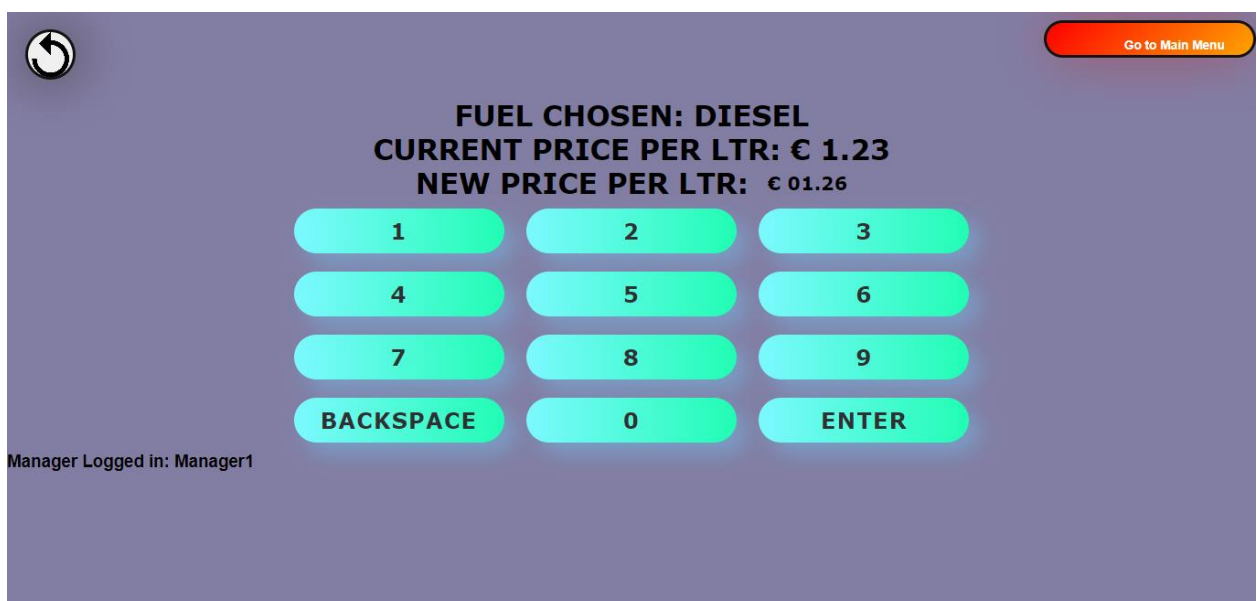


Figure 5: Pressing digit 6 from the keypad and hit enter.

This is the easiest way for the user to input a price because it is the way people in Malta write (from left to right). This leads to more accurate and realistic input while reducing the possibility for errors.

### Selecting a 'Charge by Amount' payment when a customer is getting fuel

For a 'Charge by Amount' payment, both when paying by card or paying by cash, the customer is presented with four buttons where he can select the amount of fuel from. These can be seen in Figure 6 and 7. These four buttons represent the €5, €10, €20 and €50 notes because these are the most commonly used notes when giving fuel to your vehicle. Since the money amounts to opt for are predefined amounts, there is less room for error from the user side, which results to a more precise amount.

Moreover, the continue button only appears when the user has selected at least one of the money notes. This is so to prevent the error of a user trying to buy a €0 worth of fuel. This can be shown in the Figures 6 and 7.

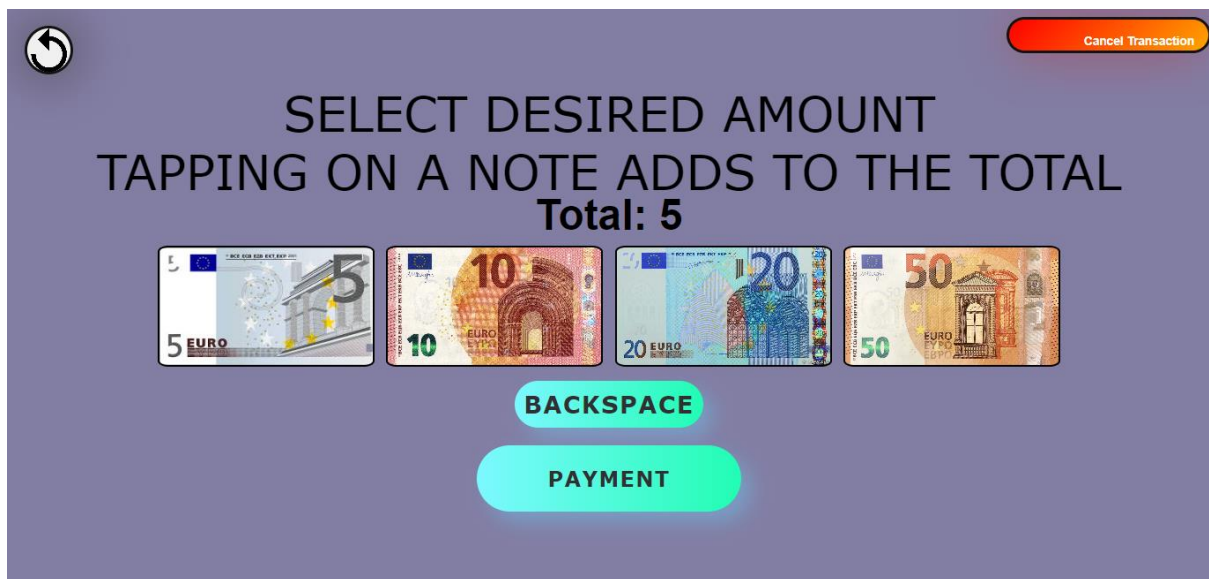


Figure 6: Paying a 'Charge by Amount' payment by card

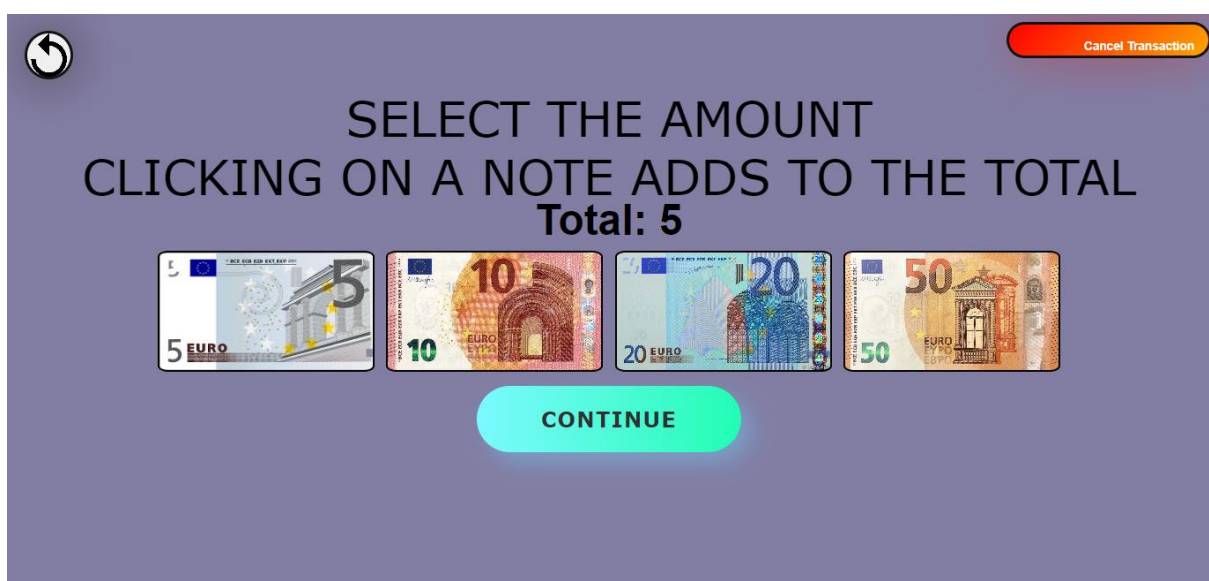


Figure 7: Paying a 'Charge by Amount' payment by cash

## 2. Help the system users identify and recover from Errors

While the system's priority is always the prevention of errors, errors can still occur. When this is the case, direct feedback is given to the user in order to act upon. In many cases feedback is given under where the input value is displayed, in order to get the user's attention.

### Inputting no or an incorrect passcode in manager's console

When the manager or an unauthorized user tries to input an empty passcode, feedback is given stating "PLEASE ENTER A PASSCODE." as evident in Figure 8.

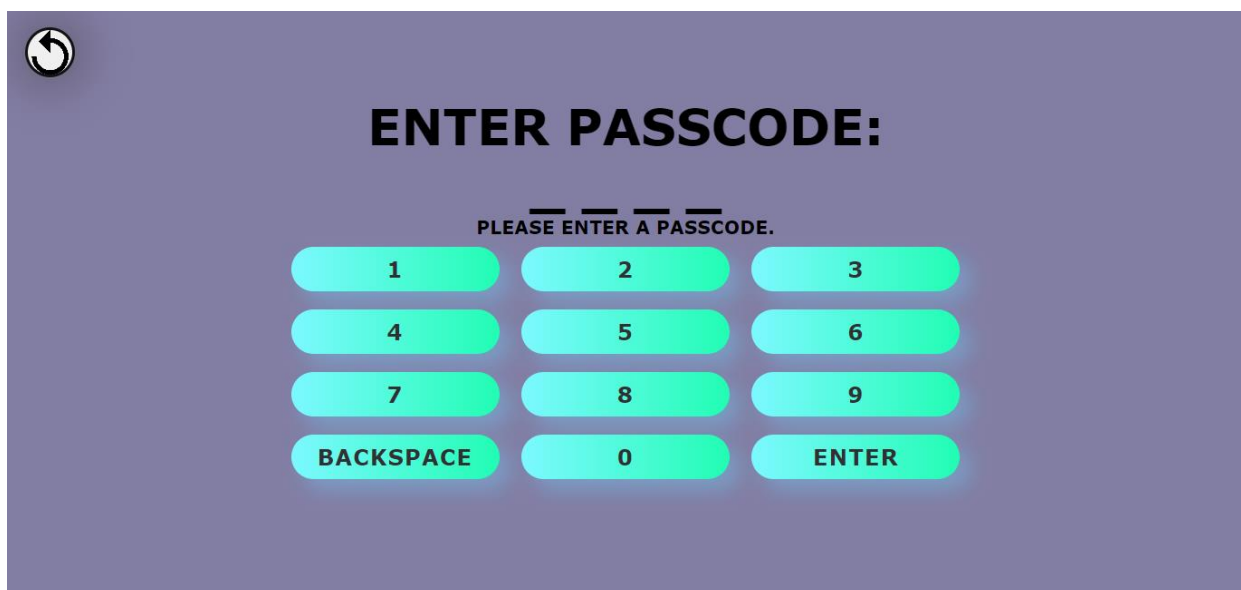


Figure 8

When the manager or an unauthorized user inputs an incorrect pin, an "INCORRECT PASSCODE" alert is displayed. The feedback is once again given beneath where the input value is displayed, with the aim of obtaining the user's attention. This can be seen in Figure 9.

Since a passcode is a confidential type of input, when an incorrect passcode is entered, it is displayed as the asterisk character, and upon attempting to login, the input field is immediately reset.

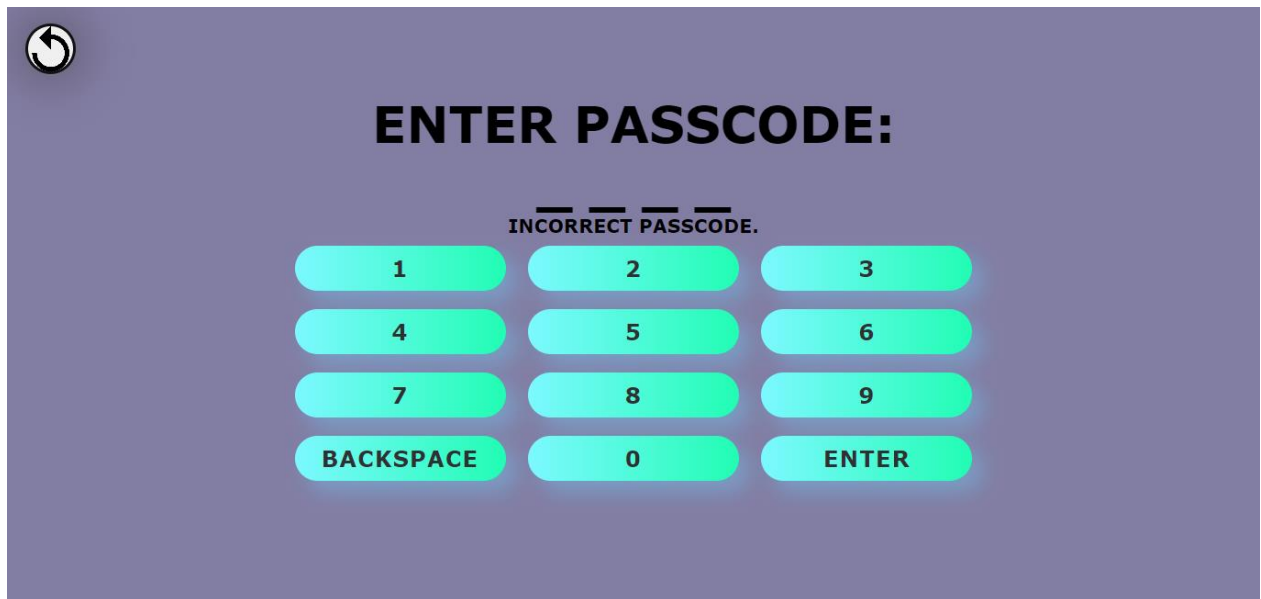


Figure 9

Entering an empty value when the manager is changing a fuel price

As the manager is using the keypad to change a fuel price and the manager attempts to enter an empty value, feedback is also given to notify the manager that a value must be provided. This is represented in Figure 10 below.

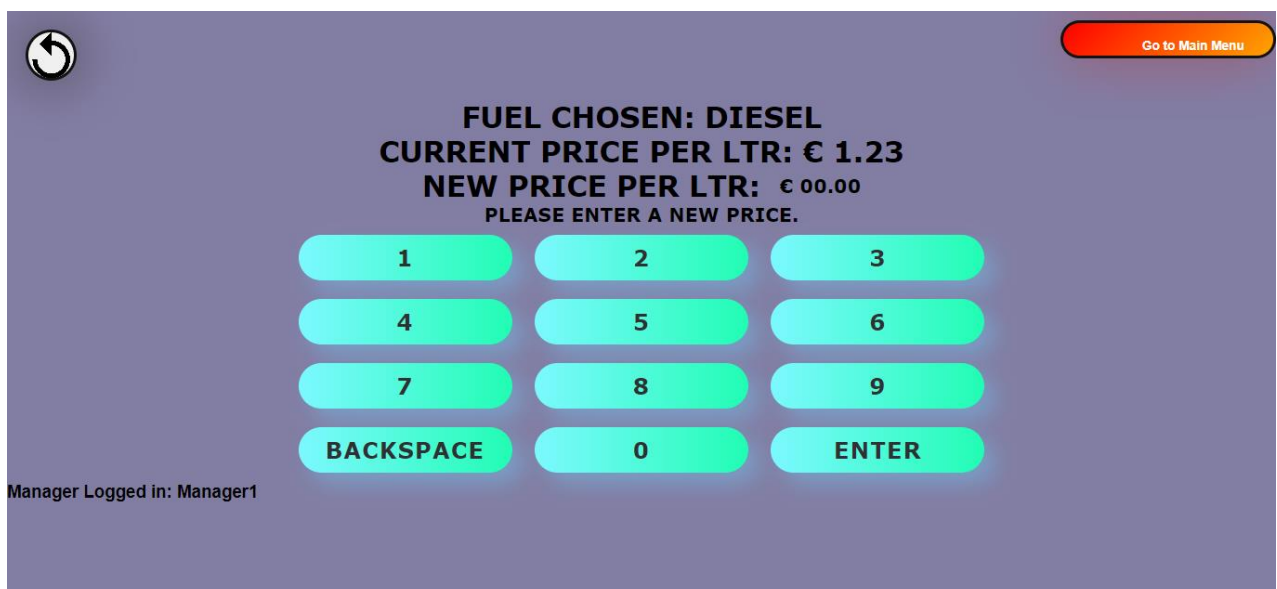


Figure 10

### Card swipe denied when a customer is buying fuel by card

At this point the bank card has been already swiped as can be seen in the animation in the program. However, if the customer's bank card is denied, the message found in Figure 11 below is displayed. Moreover, it provides the functionality for the customer to try and swipe the bank card again, in order to be able to proceed with the transaction.

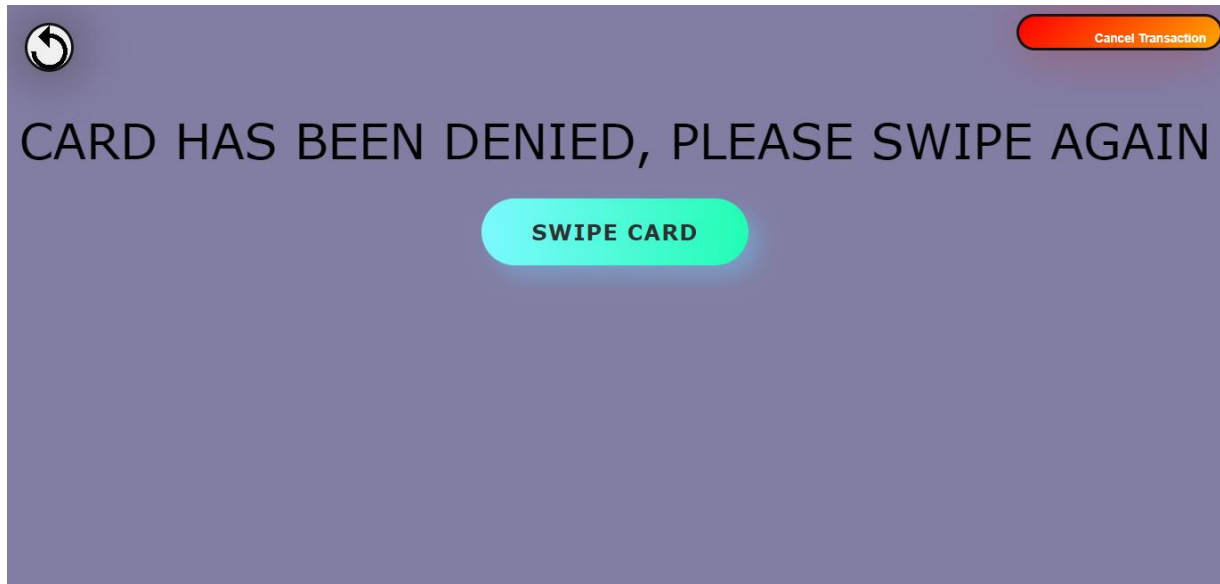


Figure 11

### 3. The system is minimalistic but maintains aesthetic design

Considering that users of all ages would be interacting with the system, it was a design goal to keep the system as simple as possible. Thus, unnecessary information and complications were kept to a minimum on our design screens. This type of design also helped to cater for users of varying technical abilities, by decreasing the chance of overwhelming them with a lot of information at once.

Furthermore, the colour scheme was selected in a way to be clearly visible both during daylight and lowlight conditions at night. However, we emphasized more on the display at lowlight conditions at night since customers will use the pumping station system mostly during this period. This is due to the fact that during the daytime, customers are given fuel by company employees directly. This translates to a design which incorporates colours such as black text on bright green coloured buttons and a grey background. Moreover, text is mostly represented in uppercase and using a sans-serif font to further obtain the user's attention. Figure 12 depicts this implementation.

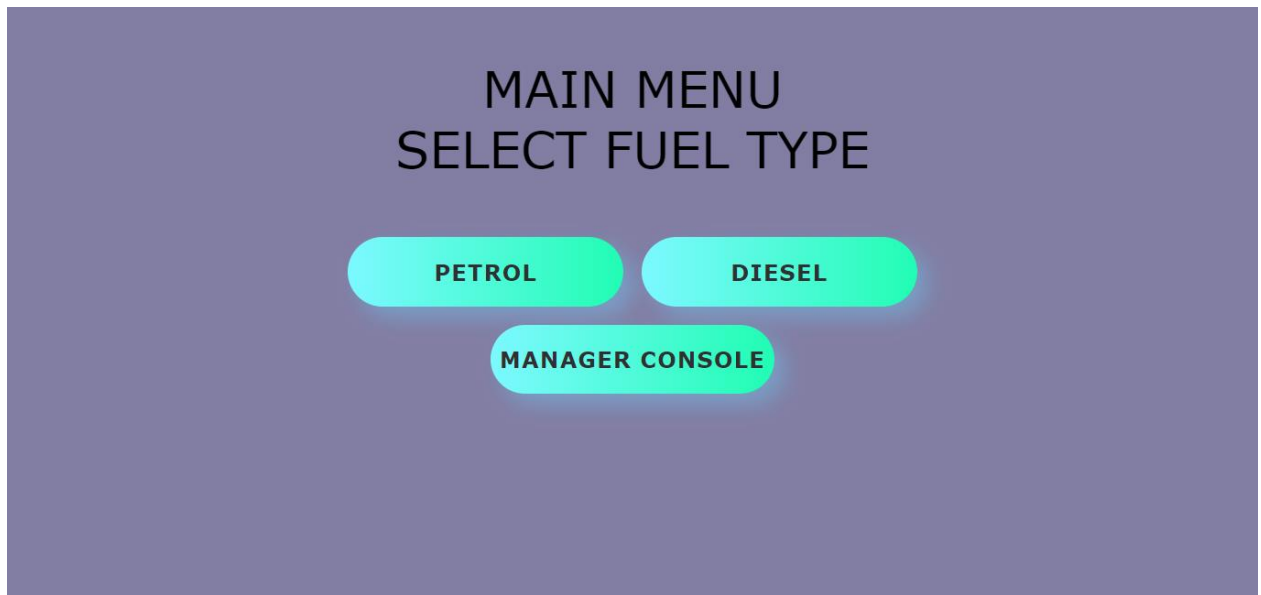


Figure 12

#### 4. Minimal memory load on the user

It is tedious and impractical for the user to remember every choice made throughout the sequence of pages while using any system. Therefore in every page there are reminders to inform the user of the selections chosen so far.

##### Fuel Selection

Upon selecting petrol, the system leads the user to select the petrol quality. The next page "SELECT PETROL QUALITY" is displayed to remind the user that of their prior selection. This is evident in Figure 13. This also happens in the manager's console when the manager is changing the price of any petrol type.

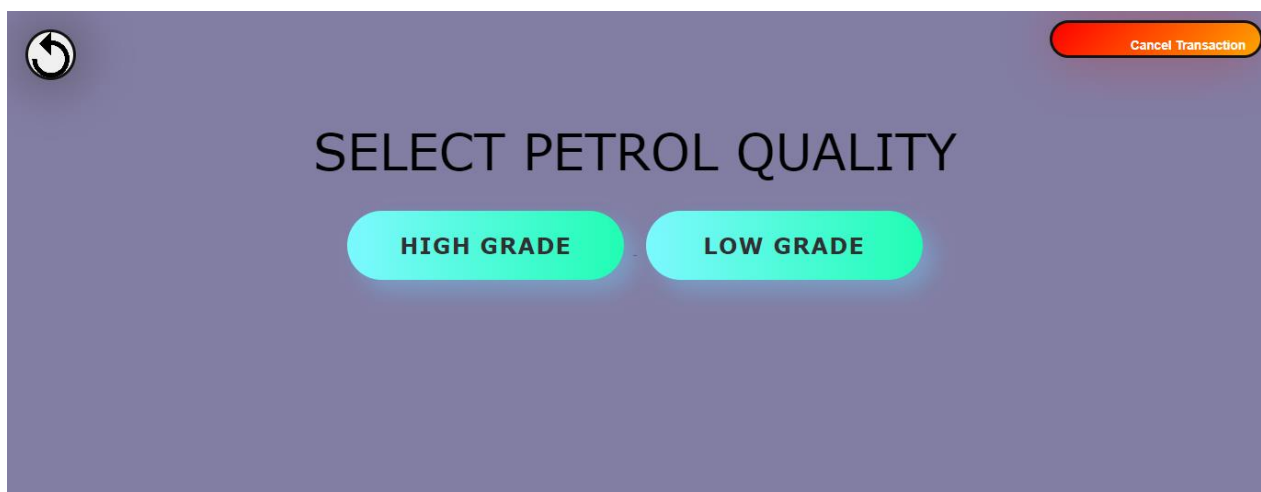


Figure 13



## Changing of fuel price

When the manager is setting a new fuel price, a page is presented after choosing the type of fuel for which to set a new price. This page is portrayed in Figure 14. In this page, the fuel chosen and current price per litre are already there as a reminder to the manager of what he chose in previous pages. The manager's username can also be seen at the bottom of the manager's page to notify who is the manager currently logged in.

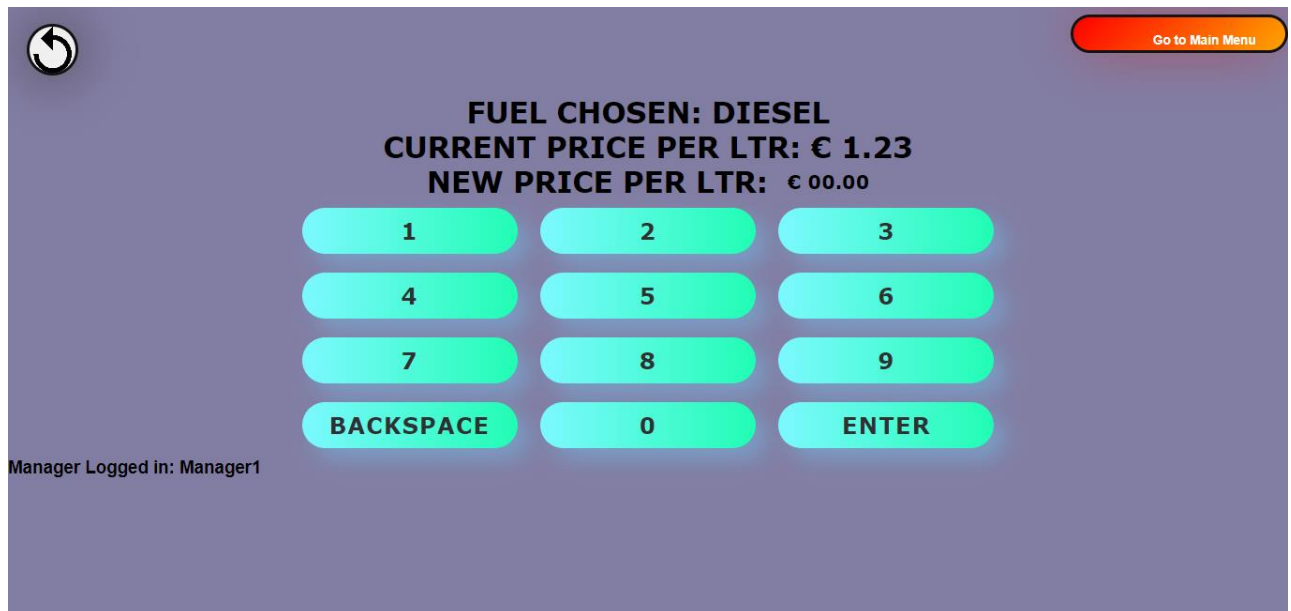
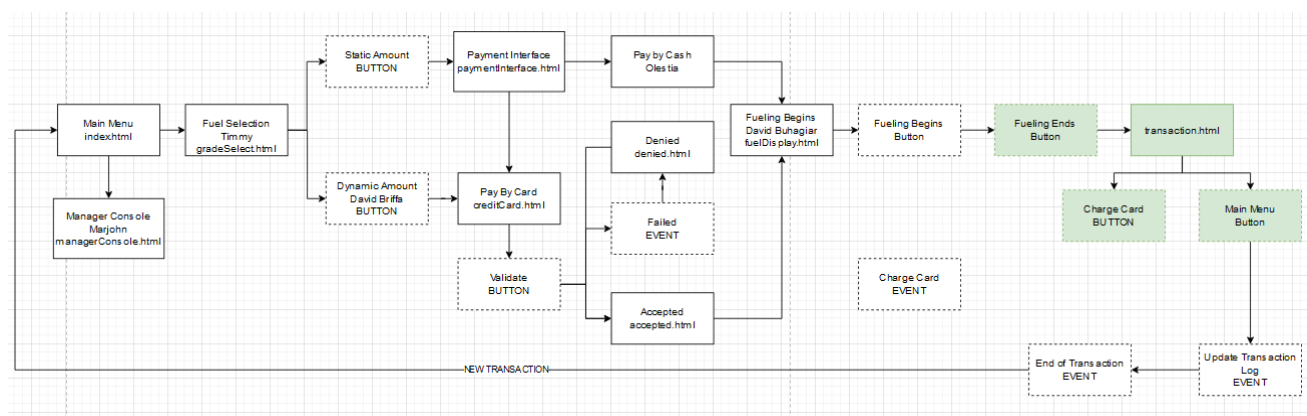


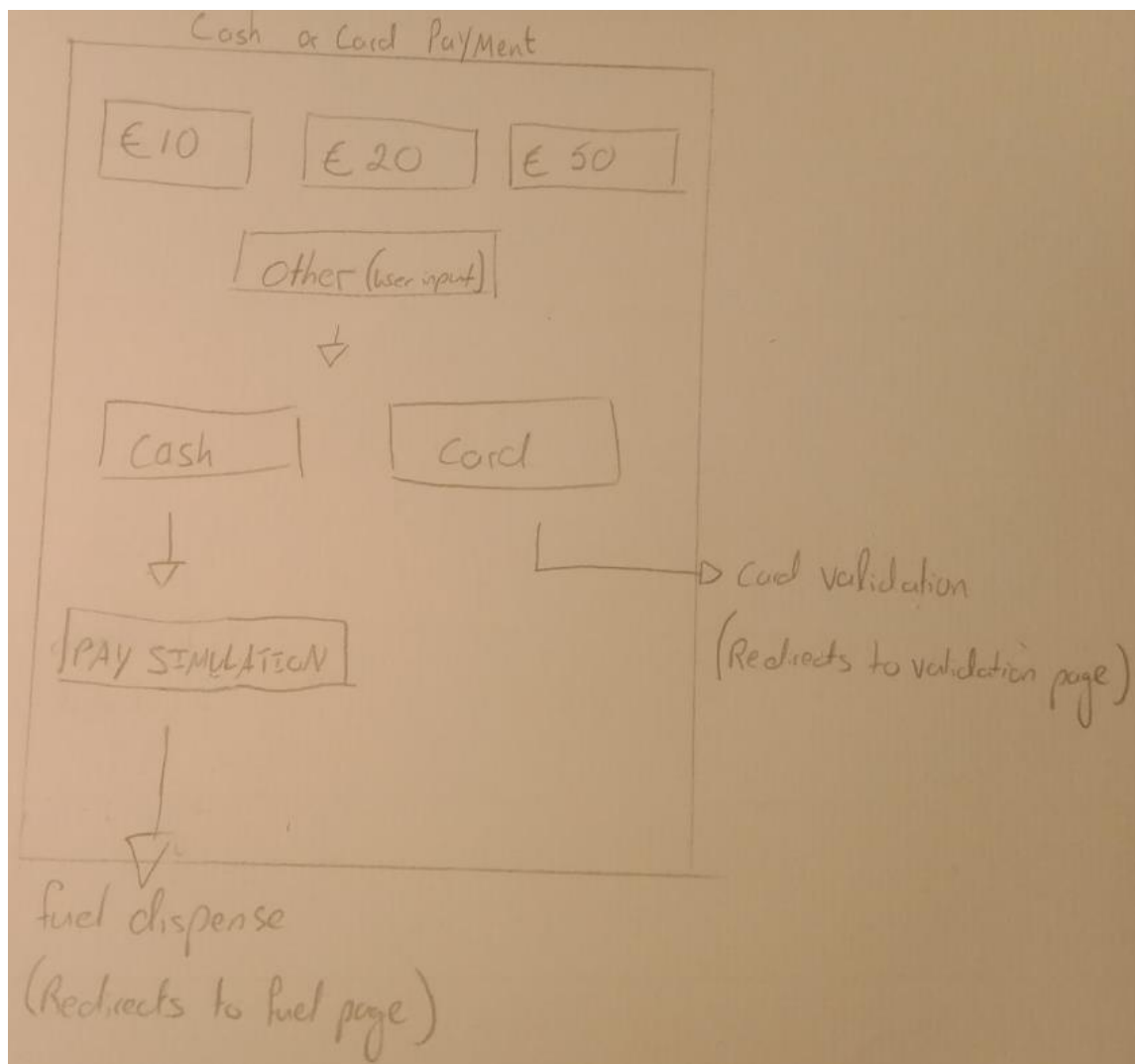
Figure 14

## Prototypes

### First Prototype-Data flow



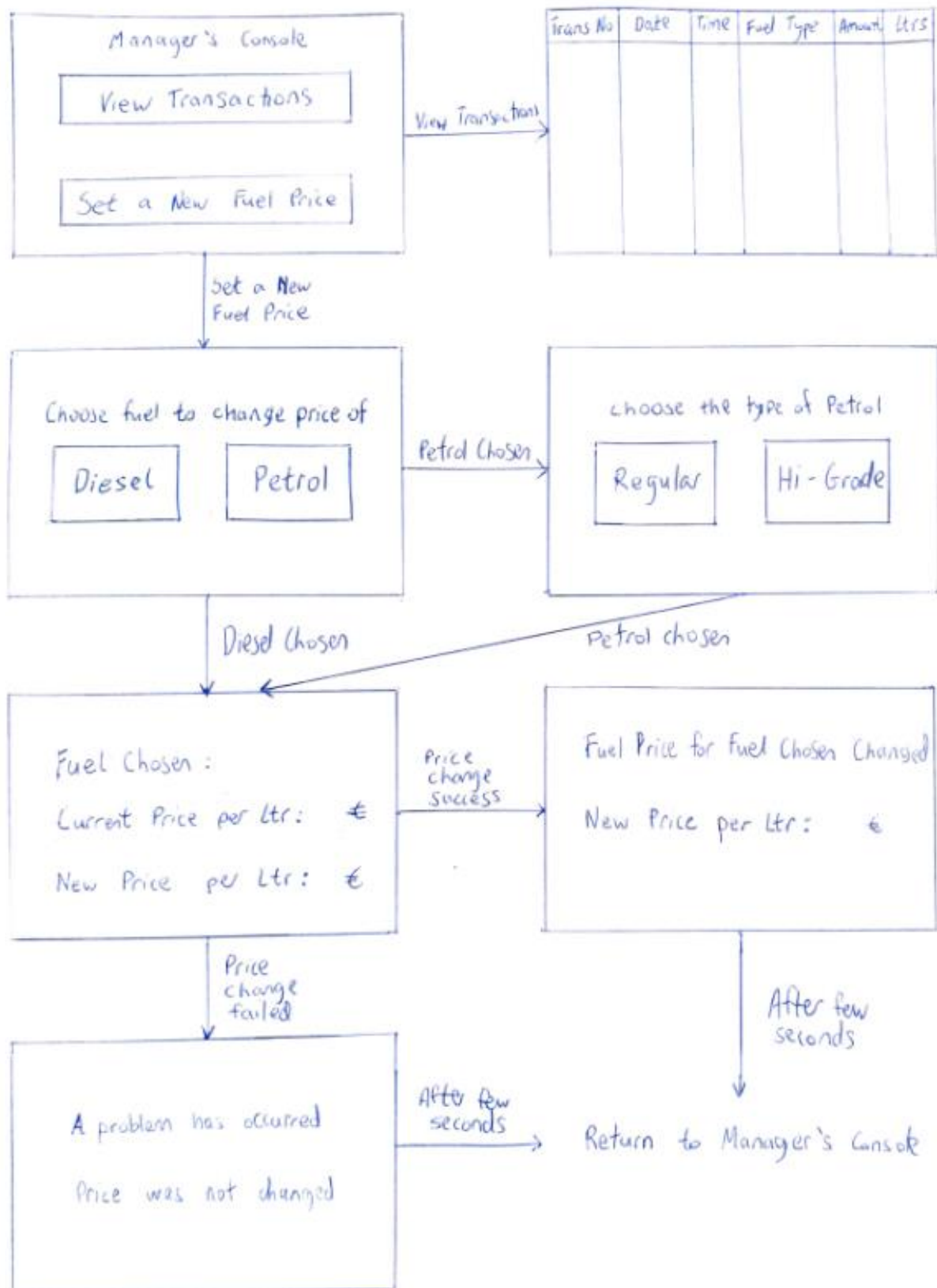
## First Prototype-Cash interface



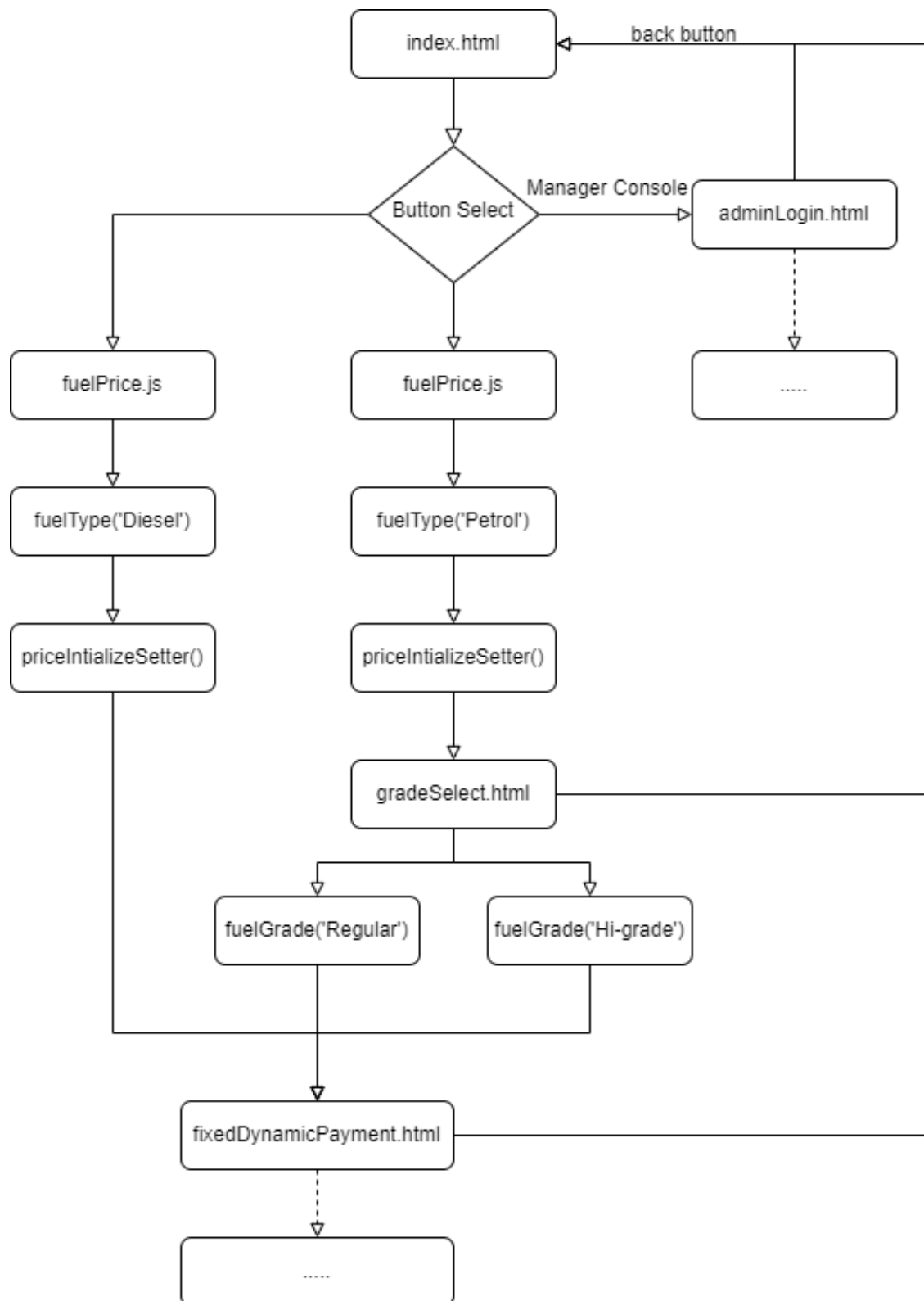
## First Prototype-Manager's Console

Manager's Console

Lo-Fidelity Prototype



## First Prototype-Fuel flow

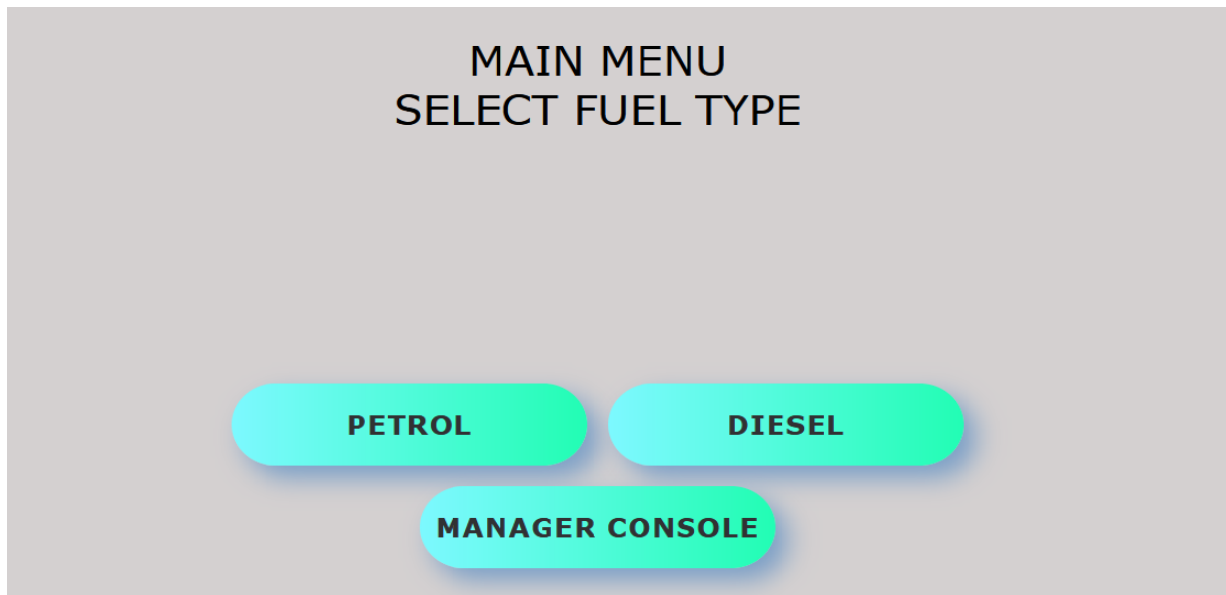


## Second Prototype-Simple Design

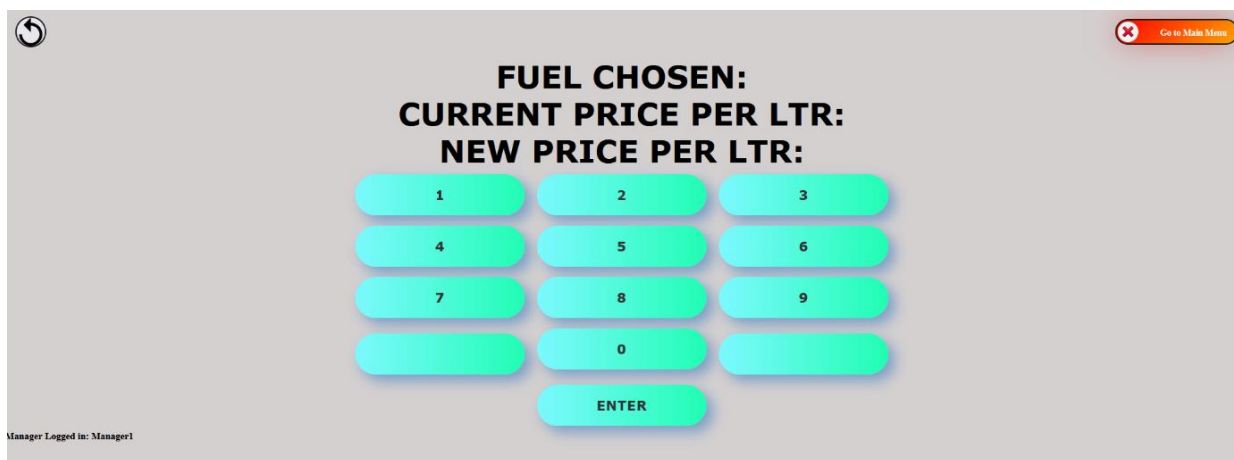


## Third Prototype- Implementation

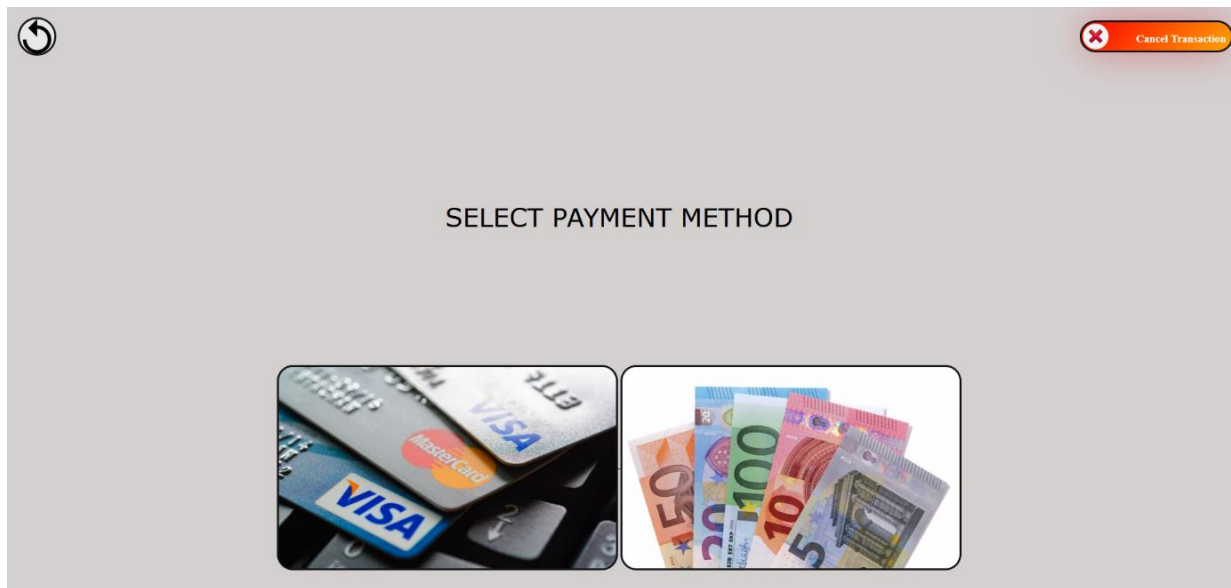
### Main menu



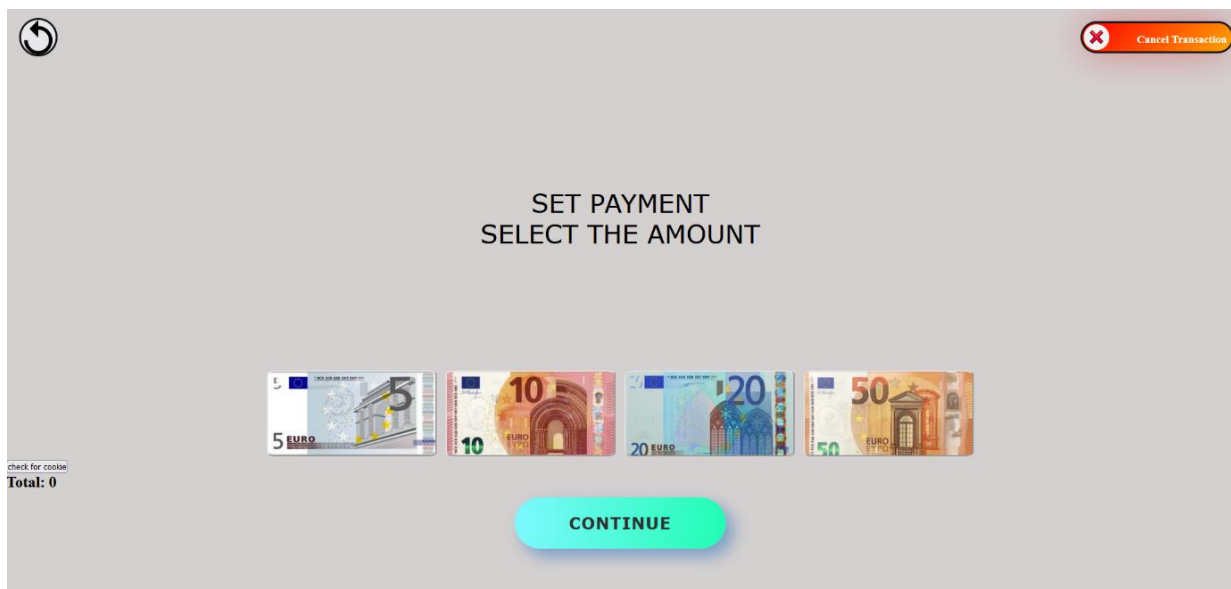
### Changing fuel price



## Selecting payment type



## Selecting amount of fuel to buy



## Pumping Interface

# FUEL DISPLAY


Total sale (€)  
120.000

This sale (€)  
0.000

Litres  
0.000

Cents per litre  
1.350

PUMP FUEL





# Testing and Evaluation

In the evaluation and benchmark testing process, the team used the “in-person” method of testing using three tasks to test all aspects of the system. Several users were chosen to try one of the three task scenarios proposed. For this evaluation there were ... users. The participants were chosen from relatives and friends. Every participant was presented with an informed consent form before starting the benchmark testing.

## Task 1: Obtain 30eur of High Grade petrol through card payment.

### Problem 1:

The users proceeded well however they found difficulty in the page labelled fixedDynamicPayment.html. It was found that the button names ‘fixed payment’ and ‘dynamic payment’ were a bit too vague and did not convey the meaning of the buttons properly. This was a recurring theme as seen in the following tasks.

### Problem 2:

A few users also pointed out that they did not have the option to undo their last input when inputting the amount, a feature that was desirable when paying by card.

## Task 2: Obtain 15eur of diesel through cash payment.

### Problem 1.2:

As mentioned in Task 1, users also found difficulty with the buttons in the page fixedDynamicPayment.html.

### Problem 3:

Some users also were a bit unsure what the pictures meant when selecting whether to pay by cash or card.

### Task 3 Obtain 10 litres of Low-Grade petrol through charge by amount

#### Problem 1.3:

As mentioned in Task 1, users also found difficulty with the buttons in the page `fixedDynamicPayment.html`.

### Task 4 Change the price of fuel (High grade petrol for example) as a manager

From the given feedback, there were no problems changing the price of fuel. This is evident in Figure 10, which shows that the system will not accept an empty input. The fuel change is effective, as shown in the following three figures:

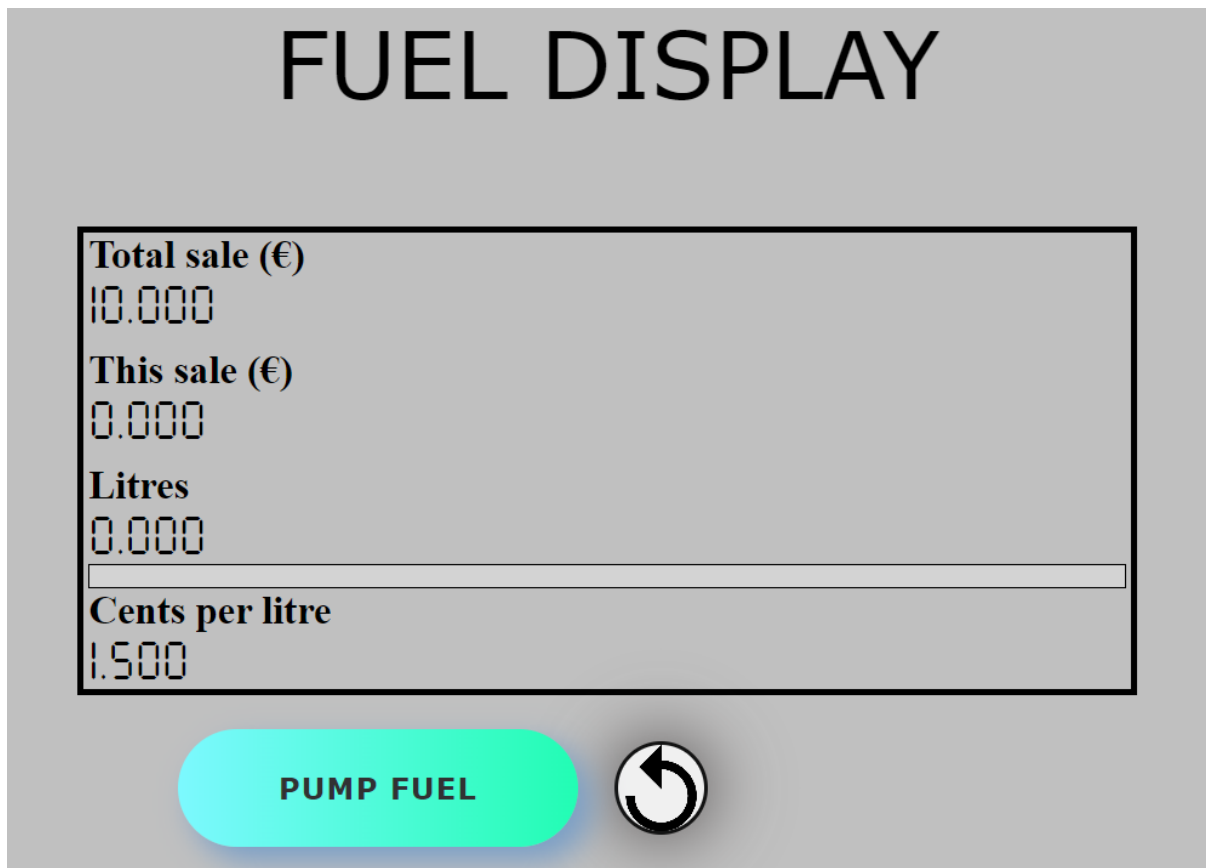


Figure 15 – Before price change



Figure 16 – Making the fuel price change

# FUEL DISPLAY

**Total sale (€)**

10.000

**This sale (€)**

0.000

**Litres**

0.000

**Cents per litre**

1.600

**PUMP FUEL**



Figure 17 – Showing that the price change is effective

## Solutions:

### Problem 1:

After asking the users what the button names should be changed to the button 'fixed payment' was changed to 'charge by amount' and 'dynamic payment' was changed to 'charge by litre'. This made the meaning and function of the individual buttons much easier to discern.

### Problem 2:

To solve this problem first a new html file called card.html was created. This is because before, when a user chose cash or card, the same page was being used to represent cash insertion. However, users requested an undo button for individual cash amounts. The same function is not available when paying by cash as it is not possible to return the cash put in by the user unless cancelling the transaction. Complementing this page was a function so that when a user is paying by card and presses the 'backspace' button the last value entered is removed from the total. The user may click backspace until the total reaches zero.

### Problem 3:

The cash and card buttons in paymentInterface.html were labelled.

## Conclusion

After the tasks were completed successfully with guidance from the moderators, the participants were asked to answer five questions about the interface they were using. The questions were paired with a scale of 1-5, with 1 being completely dissatisfied/disagree and 5 completely satisfied/agree.

Question 1: How satisfied are you with the overall quality of the service of the fuel pump interface?

Question 2: How satisfied are you with the design of the interface?

Question 3: How satisfied are you with the readability of the interface?

Question 4: How satisfied are you with the ease of use of the interface are you?

Question 5: How likely are you to prefer this interface over some of the already existing ones?

Below is the data collected from the participants:

Question/scale	1	2	3	4	5
1	0	0	0	6	4
2	0	0	0	5	5
3	0	0	4	4	2
4	0	0	3	5	2
5	0	0	0	6	4

## Informed Consent form

### Informed consent form

This is an informed consent form regarding the participation in a benchmark testing and a feedback survey of s fuel pump system. In this test you will interact with the interface that simulates a typical screen of a payment machine at a fuel pump station. You will be given a simple task that will require you to interact with the interface. You are encouraged to give feedback, whether positive or negative, on your experience at any point of your task. The tests' purpose is for the developers to evaluate how well interactable their system is. After the test you will be asked five(5) questions about your experience with the system.

By signing this informed consent form, you (the user/participant), are confirming your awareness and participation in a survey and user testing of the fuel pump station User Interface system developed by David Briffa, David Buhagiar, Marjohn Saliba, Timmy Zammit and Olesia Shtanko. No information of your identity will be disclosed at any point in time. Your participation and feedback will be taken record of for educational purposes only.

Signature \_\_\_\_\_