**Text

Description automatically generated**

**Petrol Pump Design Project**

David Briffa (376195m),

B.Sc. (Hons) Software Development

Study-unit: **User Interface Design**

Code: **CIS2201**

Lecturer: **Dr Colin Layfield**

**FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY**

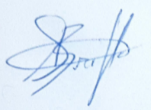
Declaration

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

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Name Signature

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Name Signature

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student Name Signature

\_\_CIS2201\_\_\_\_\_\_\_ \_\_\_\_\_\_\_Petrol Pump Design Project\_\_\_\_\_\_\_\_\_\_\_\_

Course Code Title of work submitted

\_\_\_12/12/2022\_\_\_\_\_\_\_

Date

Contents

[1. Requirements Analysis 1](#_Toc77316839)-5

[2. Prototypes 6-10](#_Toc77316843)

[3. Testing and Evaluation 11-](#_Toc77316843)

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

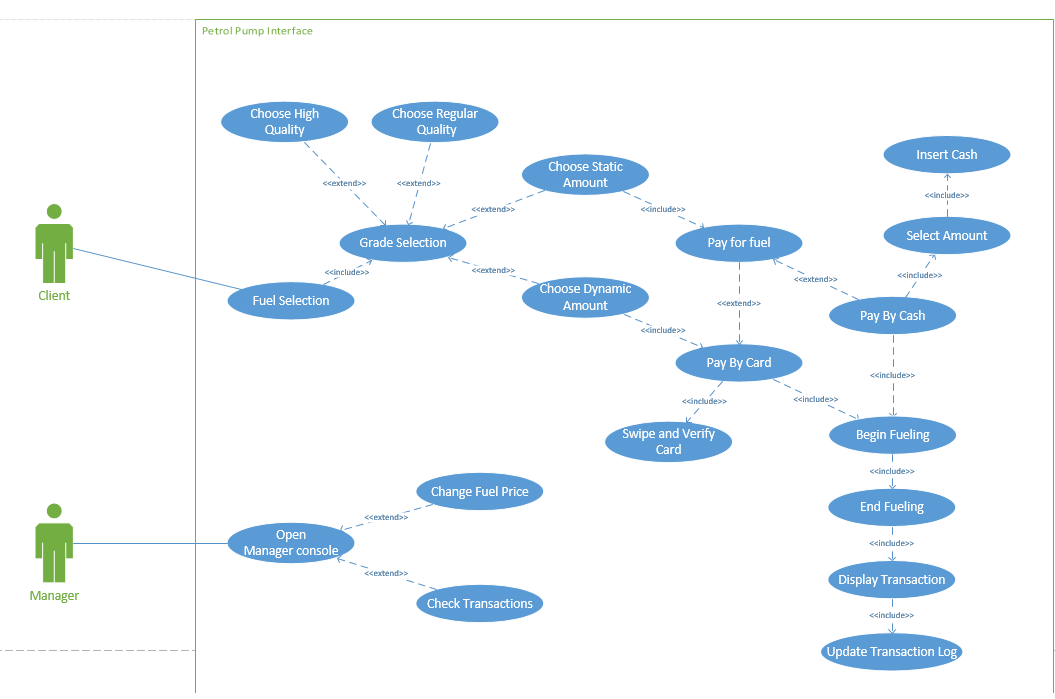
* Go back button
  + Present on every page
  + Redirects the user one page back, clearing any stored data related to the associated page
* Cancel transaction button
  + Redirects the user to the main menu, clearing all stored data

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

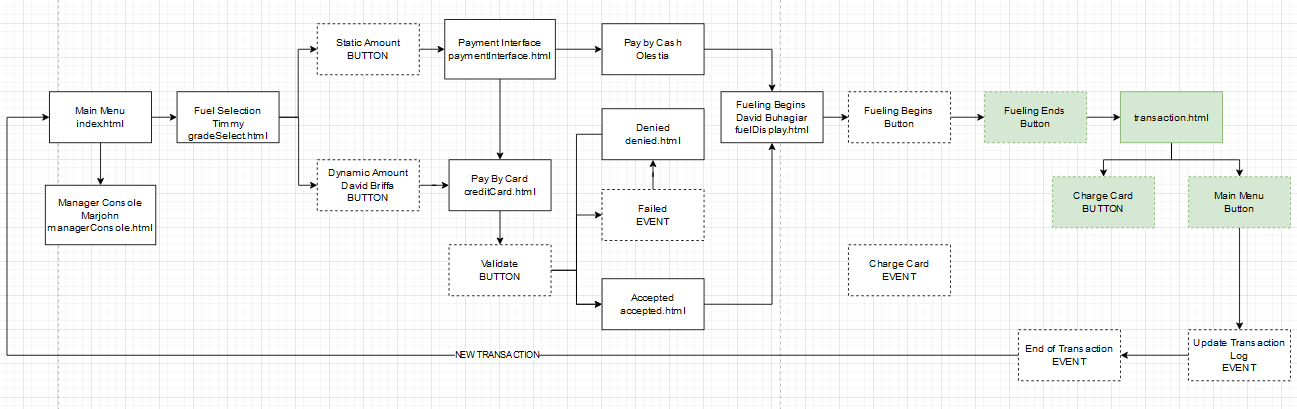
Assumptions

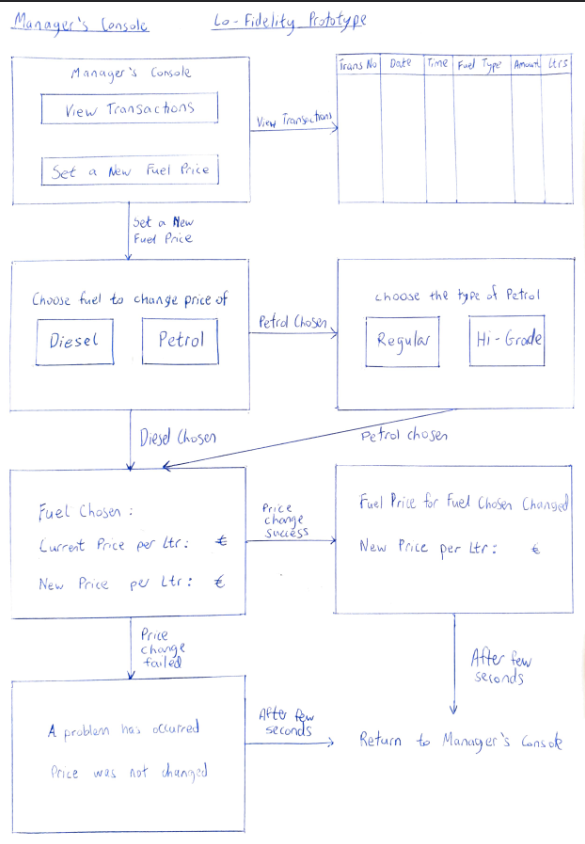
* The software will be used in the EU, and as such will use the Euro currency
* The interface is a touch screen
* The interface is connected to credit card services
* The interface accepts cash through a cash slot
* The interface accepts card through a swiping contraption

Use Case Diagram

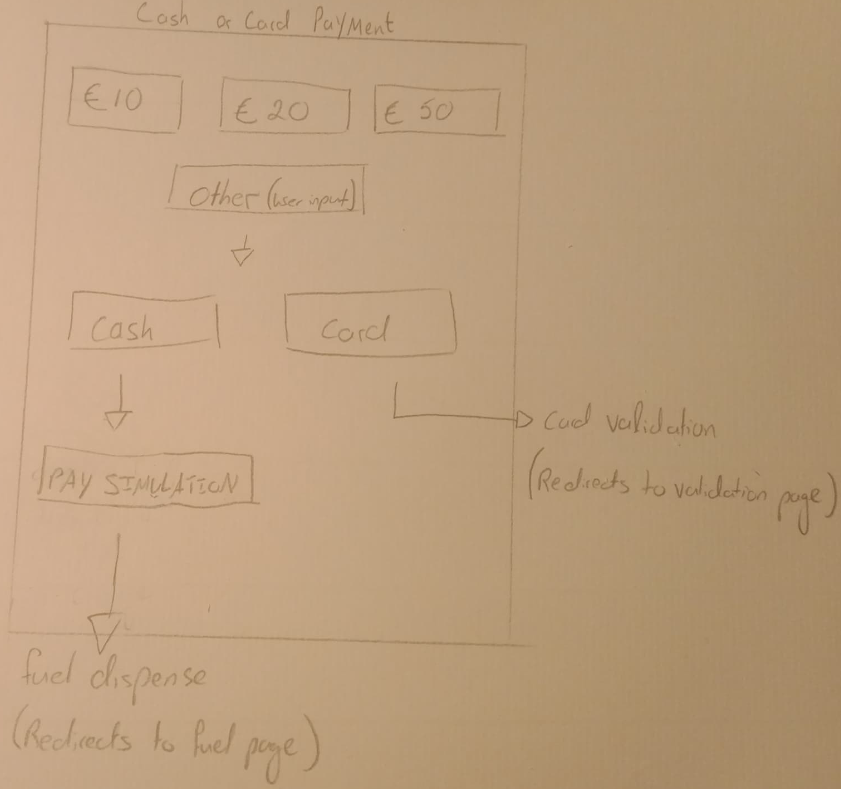
Prototypes

First Prototype-Data flow

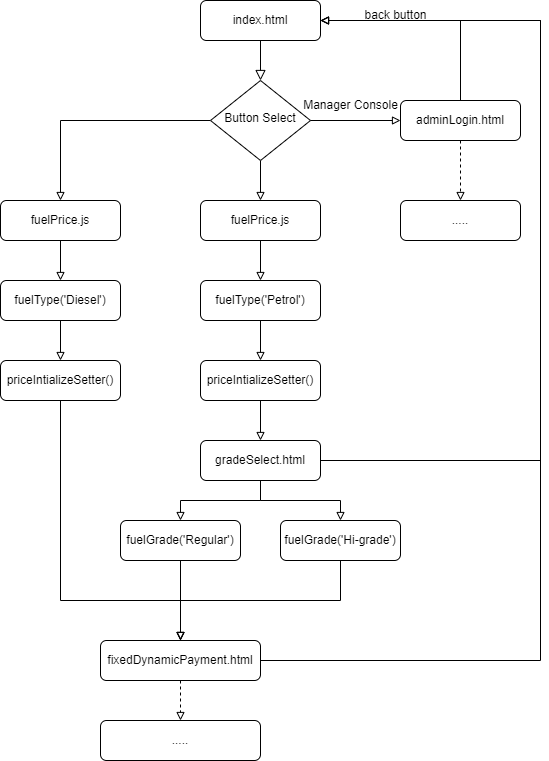


First Prototype-Manager’s Console

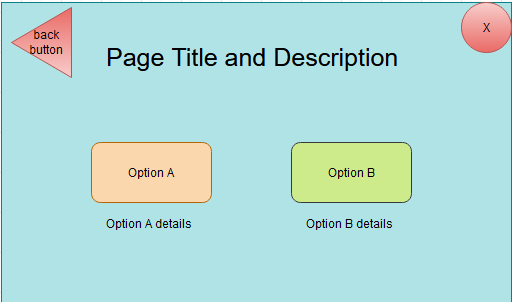
First Prototype-Cash interface



First Prototype-Fuel flow

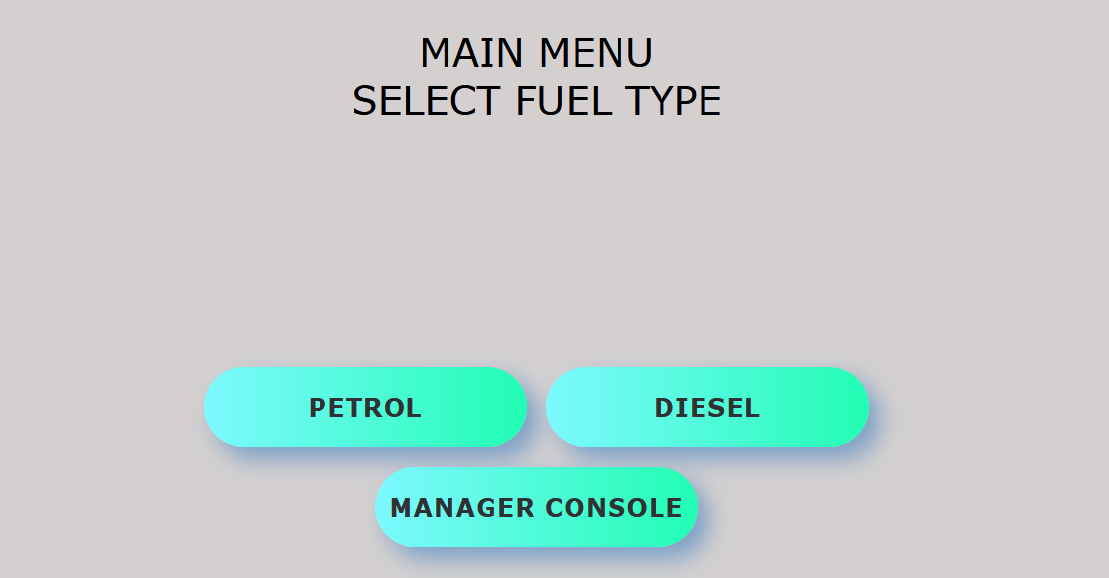


Second Prototype-Simple Design

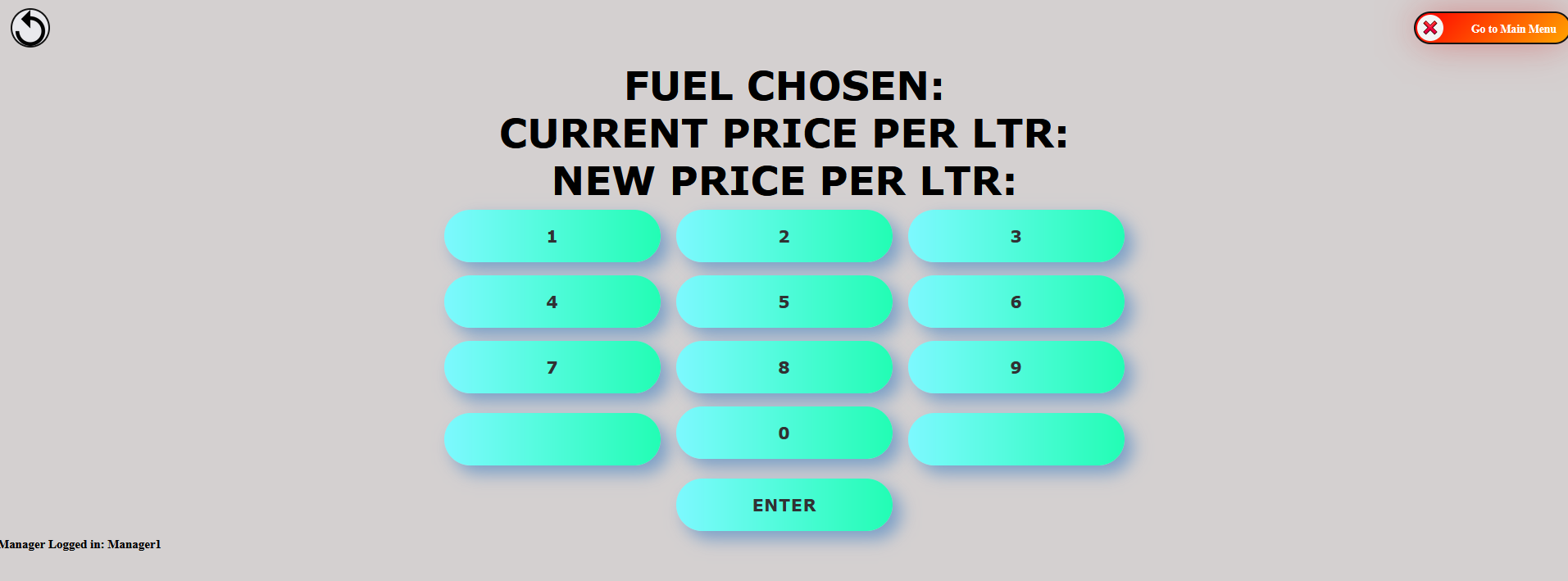


Third Prototype- Implementation

Main menu



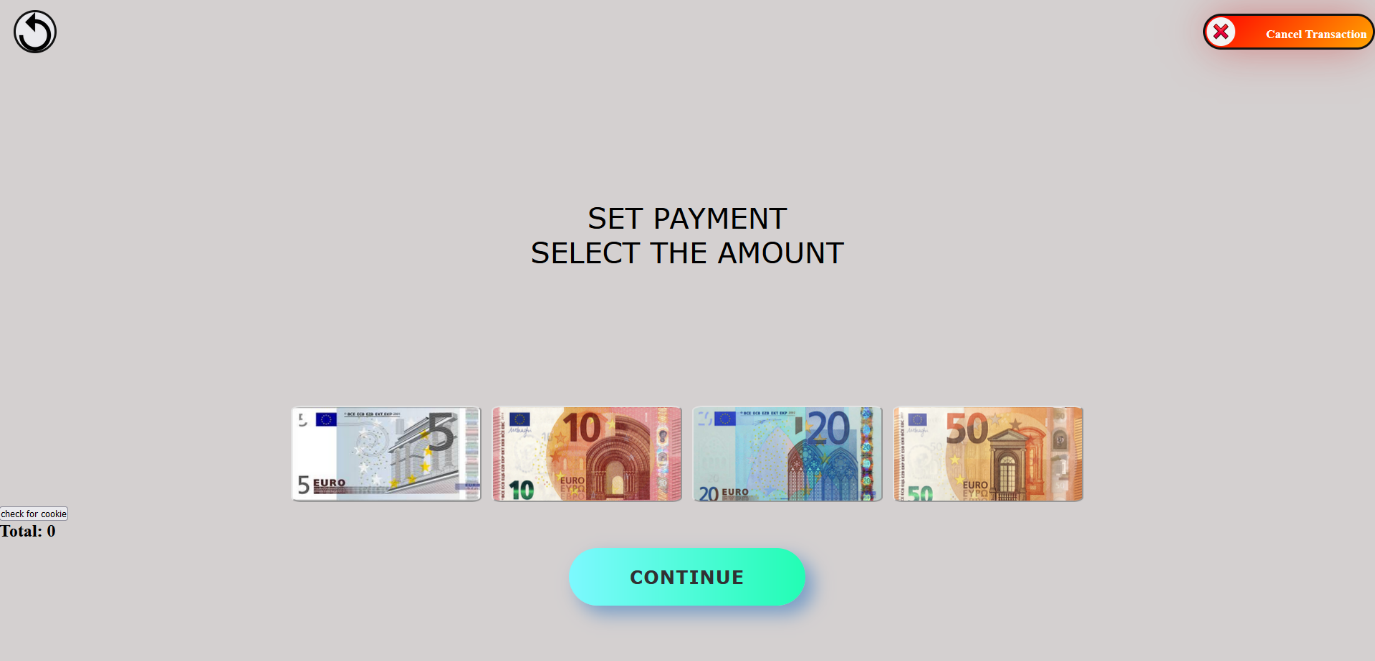
Changing fuel price



Selecting payment type



Selecting amount of fuel to buy



Testing and Evaluation