



SPRING BOOT 2.0

Fuel project σε java

13.01.2019

Ραφαήλ Κυριακόπουλος
AM 4414202

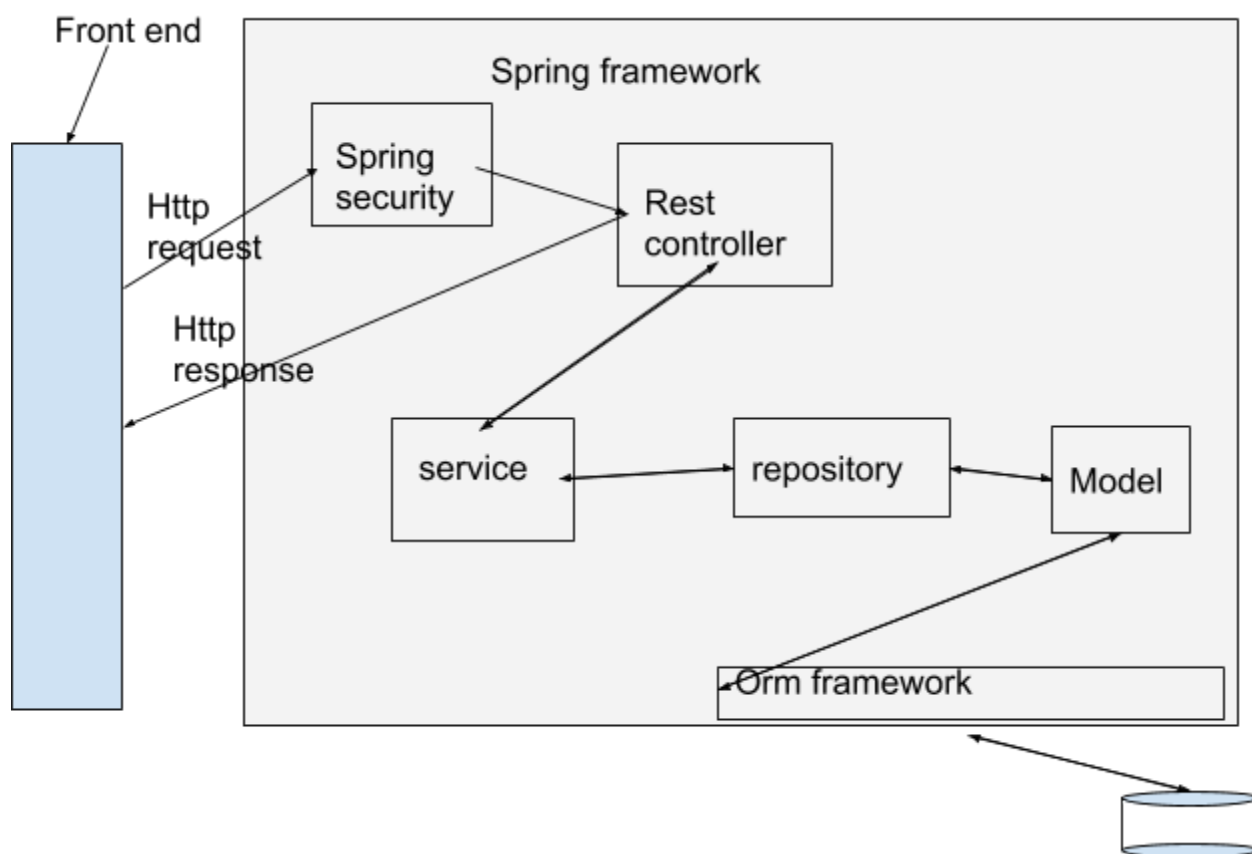
Σύνοψη

Το project δημιουργήθηκε με την γλώσσα java και χρησιμοποιήθηκε το spring framework που είναι το πιο διάσημο framework στην java είναι σαν να λέμε laravel στην php.

Στόχοι

1. Δημιουργία rest api με jwt authentication-authorization
2. Δημιουργία front end με τεχνολογίες front end **μόνο** ώστε να τρέχει σε οποιονδήποτε server .

Σχεδίασμός εφαρμογής



Back end

1:GET: Λήψη η δοδόμενων πρατηρίων και τιμών επιλεγμένου καυσίμου.

GET ▼ http://localhost:8080/gasstations/pricedata/1

Authorization Headers (2) Body Pre-request Script Tests

Key	Value
<input checked="" type="checkbox"/> Content-Type	application/json
<input type="checkbox"/> Authorization	Bearer eyJhbGciOiJIUzI1NiJ9.eyJzd...
New key	Value

Body Cookies Headers (9) Test Results

Pretty Raw Preview JSON ≡

```

1  [
2    {
3      "gasStationID": 441,
4      "gasStationLat": 39.633742,
5      "gasStationLong": 22.4324412,
6      "fuelCompID": 6,
7      "fuelCompNormalName": "AVIN",
8      "gasStationOwner": "ΜΑΚΡΑΙΩΝ Α.Ε. ΥΠΟΚ/ΜΑ Νο 53",
9      "ddID": "42010100",
10     "ddNormalName": "Δ.Δ. Λαρίσης",
11     "municipalityID": "42010000",
12     "municipalityNormalName": "ΛΑΡΙΣΑΣ",
13     "countyID": "42000000",
14     "countyName": "ΛΑΡΙΣΗΣ",
15     "gasStationAddress": "ΒΟΛΟΥ 22-24 ΛΑΡΙΣΑ",
16     "user": "user1",
17     "priceData": [
18       {
19         "productID": 1,
20         "gasStation": 441,
21         "fuelTypeID": 1,
22         "fuelSubTypeID": 9,
23         "fuelNormalName": "Αμόλυβδη 95",
24         "fuelName": "Αμόλυβδη AVIN Best 95",
25         "fuelPrice": 1.379,
26         "dateUpdated": "2016-12-01T09:54:39",
27         "premium": false
28       }
29     ]
30   }
31 ]
  
```

2:GET: Πλήθος πρατηρίων (ακέραιοι), μέγιστη, ελάχιστη και μέση τιμή ανά lt


The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:8080/statistics
- Params:** (empty)
- Authorization:** No Auth
- Headers:** 1
- Body:** (empty)
- Test Results:** (empty)
- Status:** 200
- Response Format:** JSON
- Response Body:**

```
1 {  
2   "allGasStations": 158,  
3   "avgPrice": 1.1781826,  
4   "minPrice": 0.618,  
5   "maxPrice": 1.875  
6 }
```





3:GET: Λήψη τιμοκαταλόγου ενός πρατηρίου

GET  http://192.168.1.5:8080/gasstations/441/pricelist

Authorization Headers (2) Body Pre-request Script Tests

Type No Auth

body Cookies Headers (9) Test Results

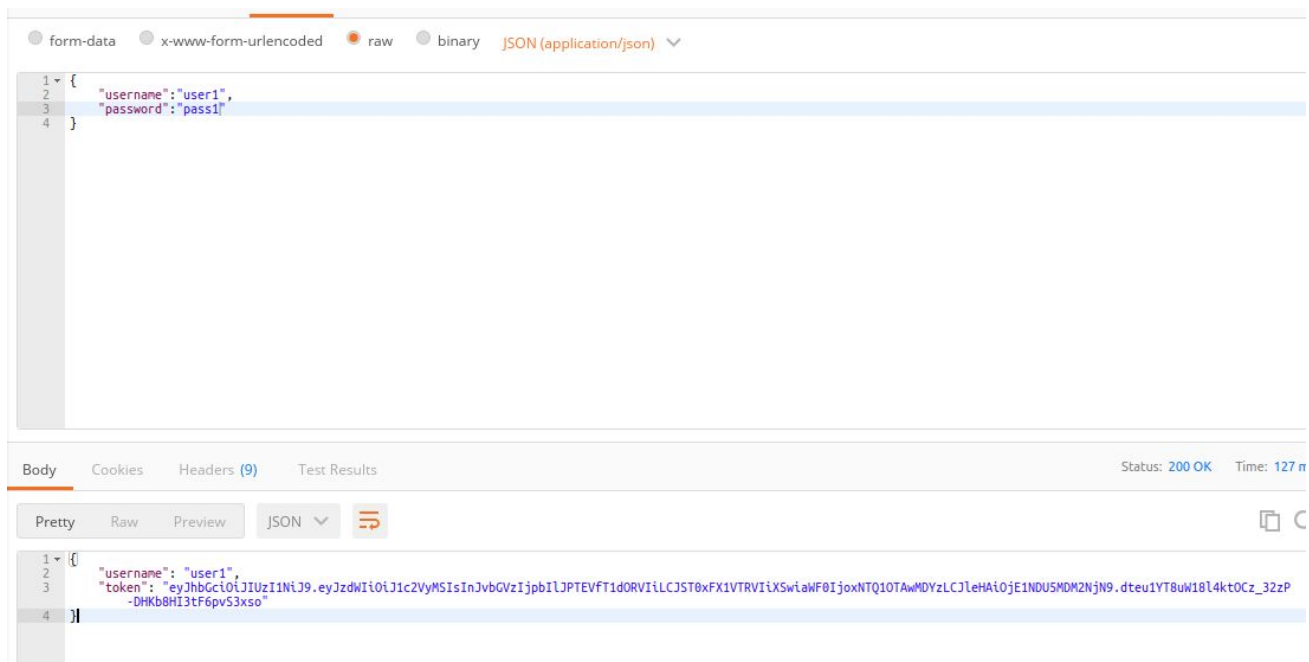
Pretty Raw Preview JSON  

```

1 {
2   "gasStationID": 441,
3   "gasStationLat": 39.633742,
4   "gasStationLong": 22.4324412,
5   "fuelCompID": 6,
6   "fuelCompNormalName": "AVIN",
7   "gasStationOwner": "ΜΑΚΡΑΙΩΝ Α.Ε. ΥΠΟΚ/ΜΑ Νο 53",
8   "ddID": "42010100",
9   "ddNormalName": "Δ.Δ. Λαρίσης",
10  "municipalityID": "42010000",
11  "municipalityNormalName": "ΛΑΡΙΣΑΣ",
12  "countyID": "42000000",
13  "countyName": "ΛΑΡΙΣΗΣ",
14  "gasStationAddress": "ΒΟΛΟΥ 22-24 ΛΑΡΙΣΑ",
15  "user": "user1",
16  "priceData": [
17    {
18      "productID": 1,
19      "gasStation": 441,
20      "fuelTypeID": 1,
21      "fuelSubTypeID": 9,
22      "fuelNormalName": "Αμόλυβδη 95",
23      "fuelName": "Αμόλυβδη AVIN Best 95",
24      "fuelPrice": 1.379,
25      "dateUpdated": "2016-12-01T09:54:39",
26      "premium": false
27    },
28    {
29      "productID": 2,
30      "gasStation": 441,
31      "fuelTypeID": 2,
32      "fuelSubTypeID": 0.

```

4:POST: Login $\chi\rho\acute{\eta}\sigma\tau\eta$ (η database έχει ήδη η $\chi\rho\acute{\eta}\sigma\tau\epsilon\varsigma$).



5:POST: Αποστολή δεδομένων παραγγελίας από πλευράς $\chi\rho\acute{\eta}\sigma\tau\eta$ (ποσό τητα)

The screenshot displays a REST client interface with a POST request to `http://localhost:8080/orders`. The request body is a JSON object with the following structure:

```
1 {
2   "orderId" : 17,
3   "quantity":1,
4   "priceData":{
5     "productID":52
6   },
7   "user":{
8     "username":"user1"
9   }
10 }
11 }
```

The response body is also a JSON object:

```
1 {
2   "message": "Your order has been stored",
3   "username": "user1",
4   "orderId": "3"
5 }
```

6DB:GET: Λήψη παραγγελιών πρατηρίου

GET

Authorization Headers (1) Body Pre-request Script Tests

Type

Body Cookies Headers (3) Test Results

Pretty Raw Preview JSON

```
1  [
2    {
3      "orderID": 7,
4      "priceData": {
5        "productID": 2
6      },
7      "user": {
8        "username": "user1"
9      },
10     "quantity": 1,
11     "when": "2018-12-24T01:26:50"
12   }
13 ]
```

7:PUT: Αλλαγή τιμής σε καύσιμο

The screenshot displays a REST client interface with a PUT request to `http://localhost:8080/pricedata/1`. The request body is a JSON object with the following fields:

```

{
  "fuelNormalName": "Αμόλυβδη 95",
  "fuelName": "Αμόλυβδη AVIN Best 95",
  "fuelPrice": 3
}

```

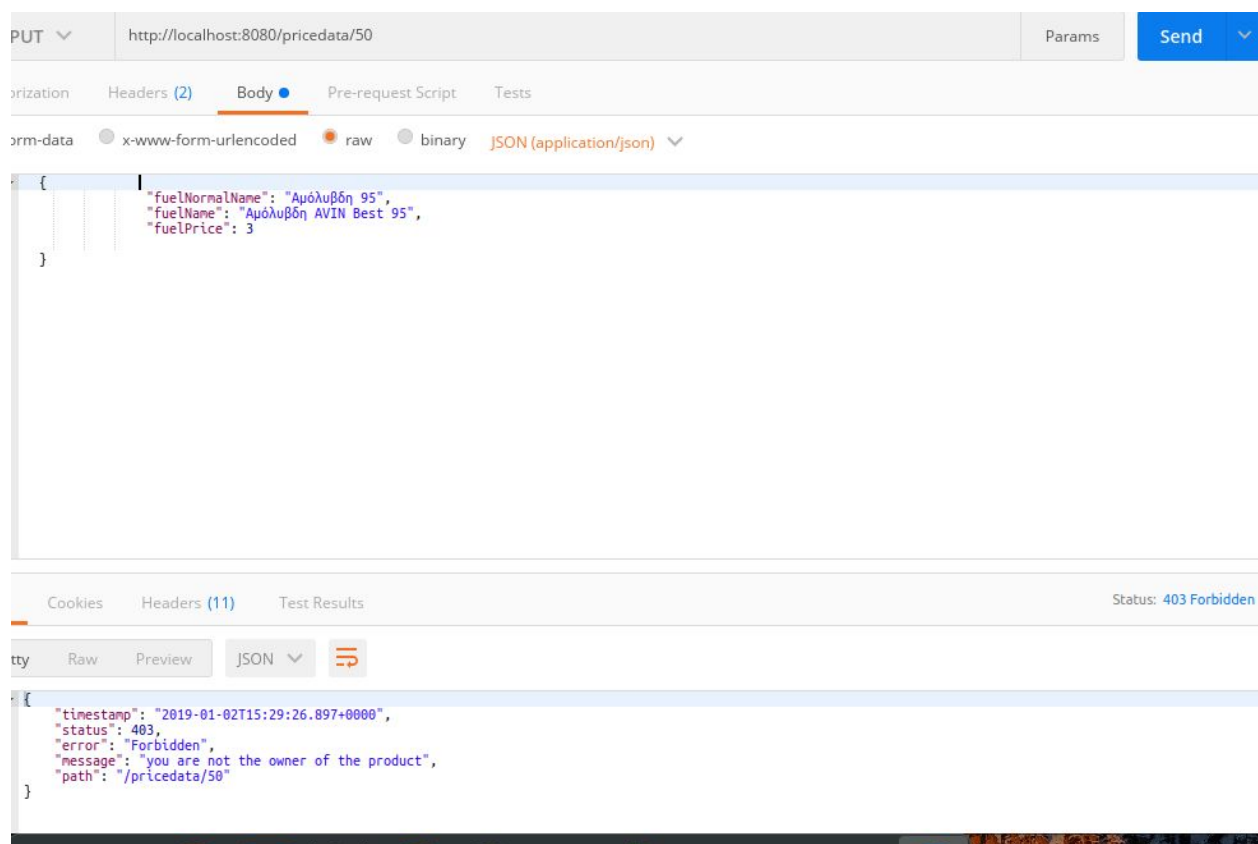
The response body is also a JSON object:

```

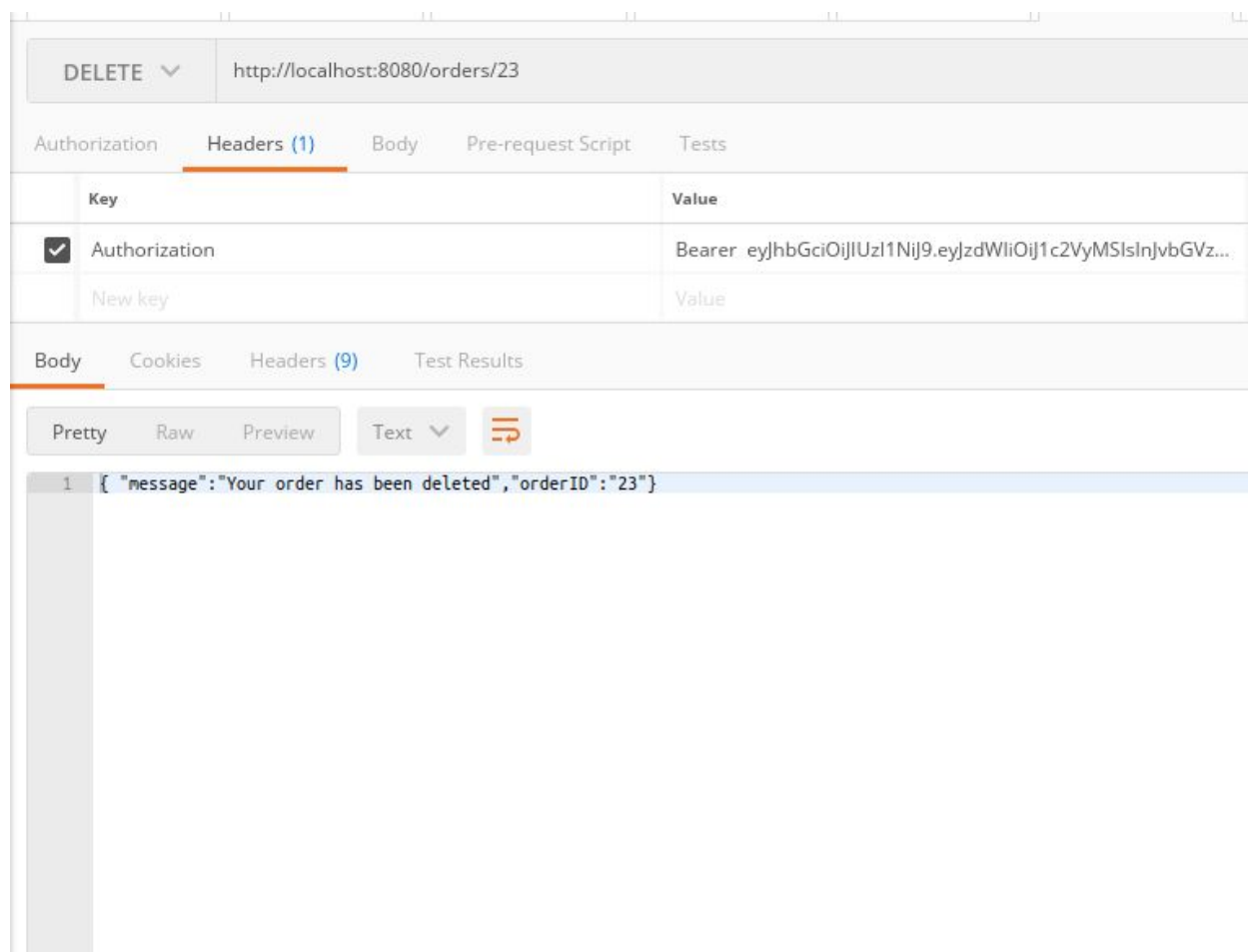
{
  "message": "Your product has been updated",
  "productID": "1"
}

```

7:PUT: Αλλαγή τιμής σε καύσιμο δεν ειναι owner



8:DELETE: Διαγραφή παραγγελίας από τη ριούχο.



```
//DELETE: Διαγραφή παραγγελίας από πρατηριούχο
@DeleteMapping("/{orderId}")
@Secured("ROLE_OWNER")
public String deleteOrder(@PathVariable Long orderId, Principal principal) {
    if(!orderService.existsById(orderId)) {
        throw new ResourceNotFoundException("orders", "orderId", orderId);
    }

    Order order = orderService.findOrderById(orderId);

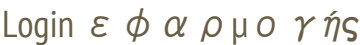
    if(!order.getUser().getUsername().equals(principal.getName())) {
        throw new ForbiddenException("You can not delete the order because you are not the owner of order id "+orderId);
    }

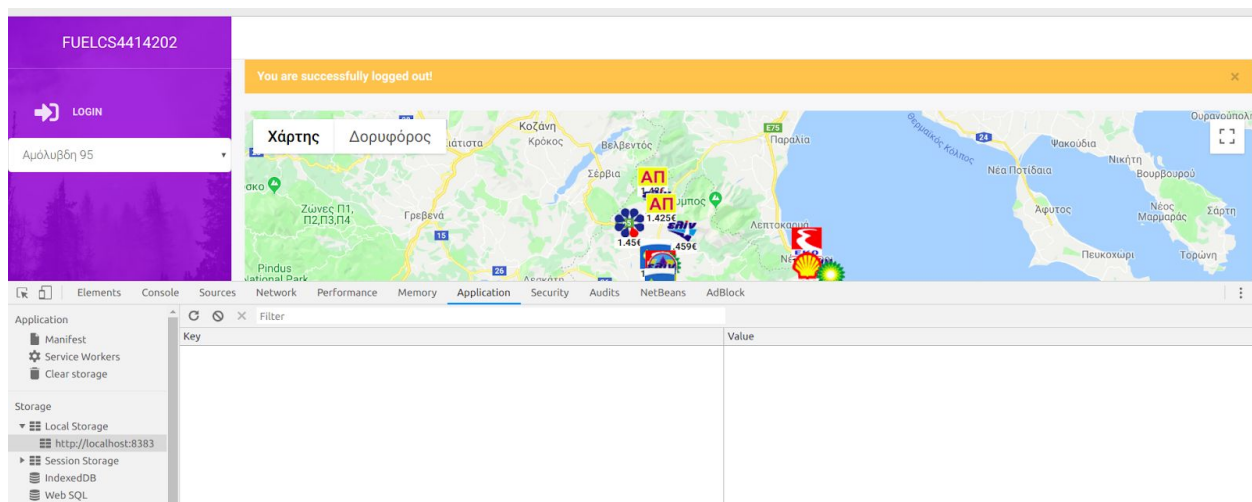
    orderService.deleteOrder(order);

    return "{ \"message\": \"Your order has been deleted\", \"orderId\": \"\"+orderId+\"\"}";
}
}
```

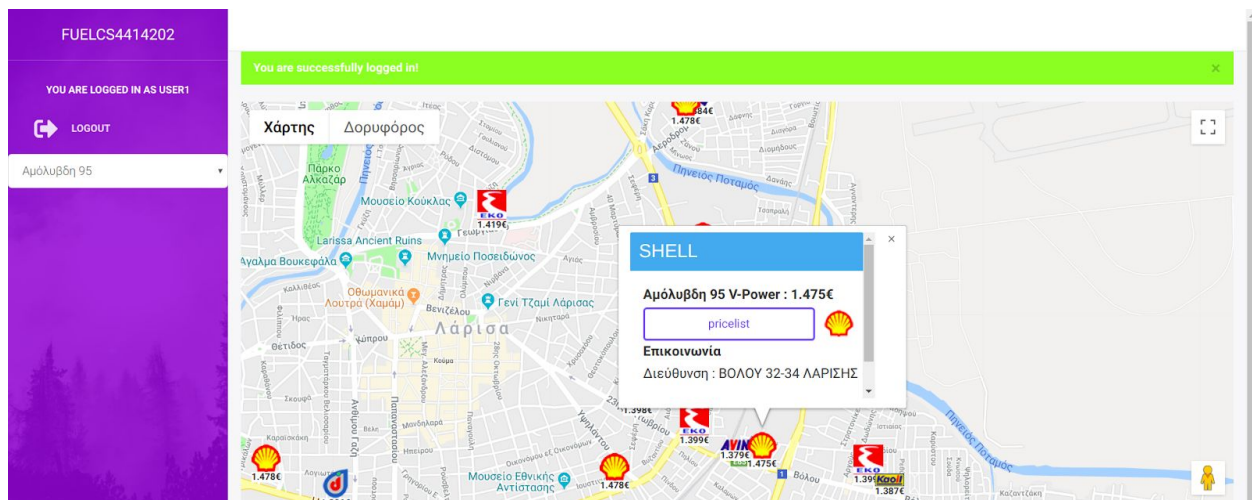
Front end

Πρώτο άνοιγμα εφαρμογής



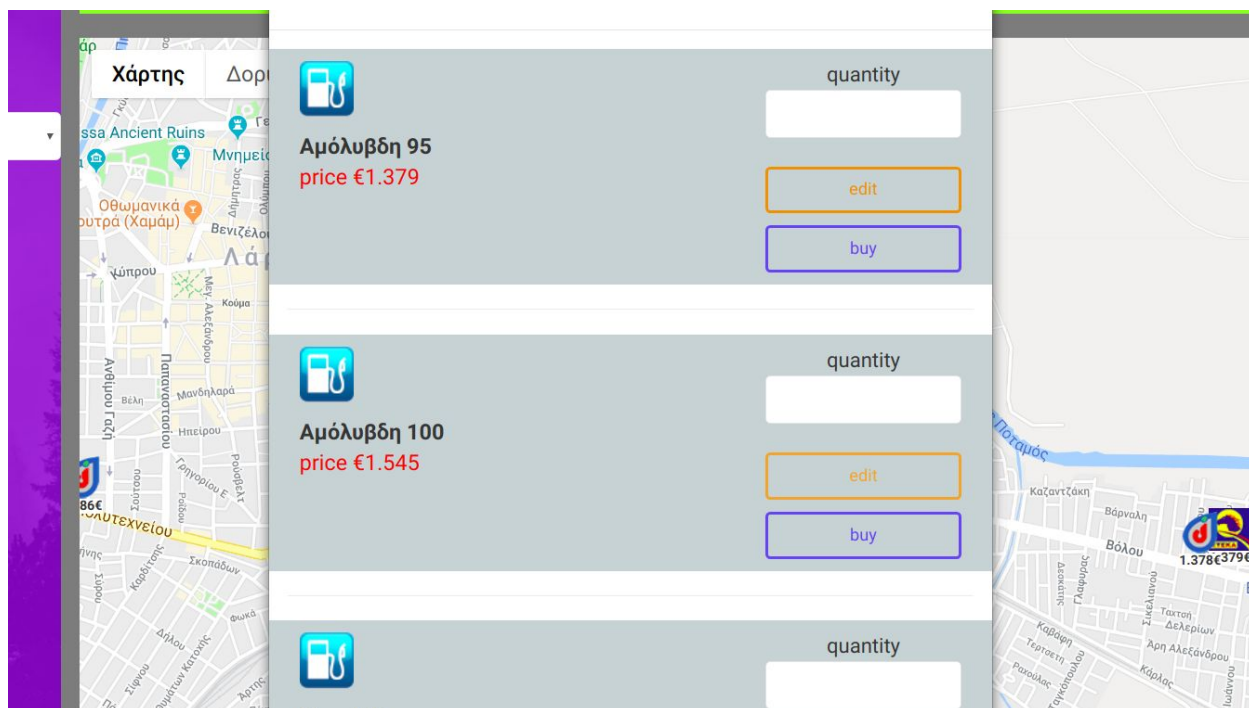


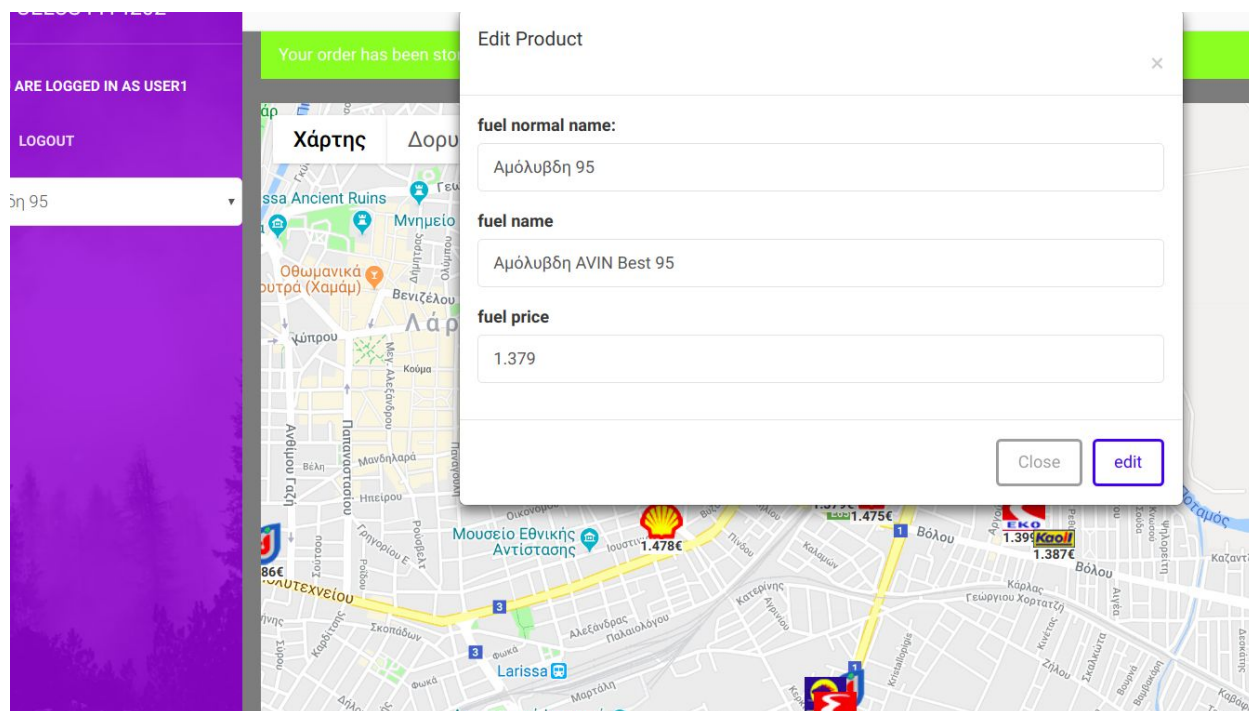
Αν είμαστε συνδεδεμένοι χρήστες μπορούμε να πατήσουμε στο price list και να κάνουμε παραγγελία





Αν είμαστε ιδιοκτήτες του πρατηρίου τότε μπορούμε να κάνουμε edit ένα product





Αν είμαστε ιδιοκτήτες μπορούμε να δούμε τις παραγγελίες

