CEN 308 SOFTWARE ENGINEERING

PROJECT DOCUMENTATION

CARMARKET

Prepared by:
Valeriu Cernei
Nicolae Topala

Proposed to:
Nermina Durmić, Assist. Prof. Dr.
Aldin Kovačević, Teaching Assistant

17.06.2021

Table of Contents

1. Introduction.

1.1. About the Project

Our website is a marketplace where all car holders, dealers and professionals can sell, buy and easily trade their autos. Website is made to be first of all user-friendly so anybody could use it easily.

The process of adding new cars on the site should be as easy as possible. Also, there could be an option for professionals, to have a 'Professional account' so they can manage their supplies smarter and more complex, not like simple users which are selling 1-2 cars.

Website search engine must be smart enough, and easy to use, to let site visitor easily choose a car, even if they don't know all the car brands.

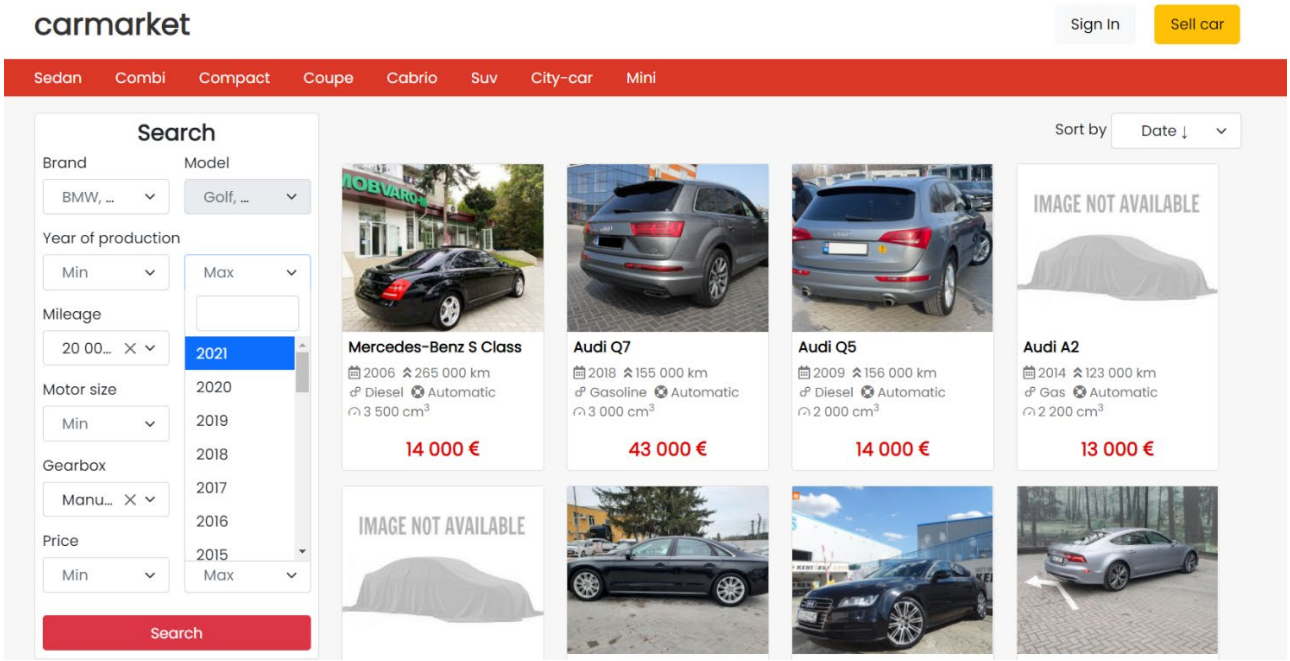
Link: https://car-market.live
Github Repo: https://github.com/valeriucernei/SE_Project

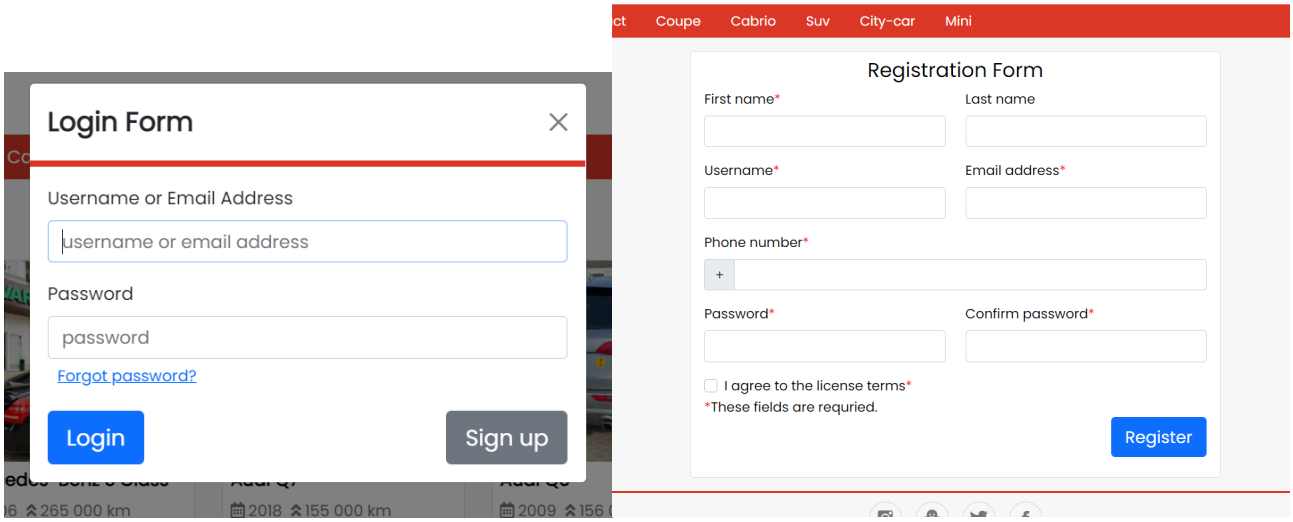
1.2. Project Functionalities and Screenshots

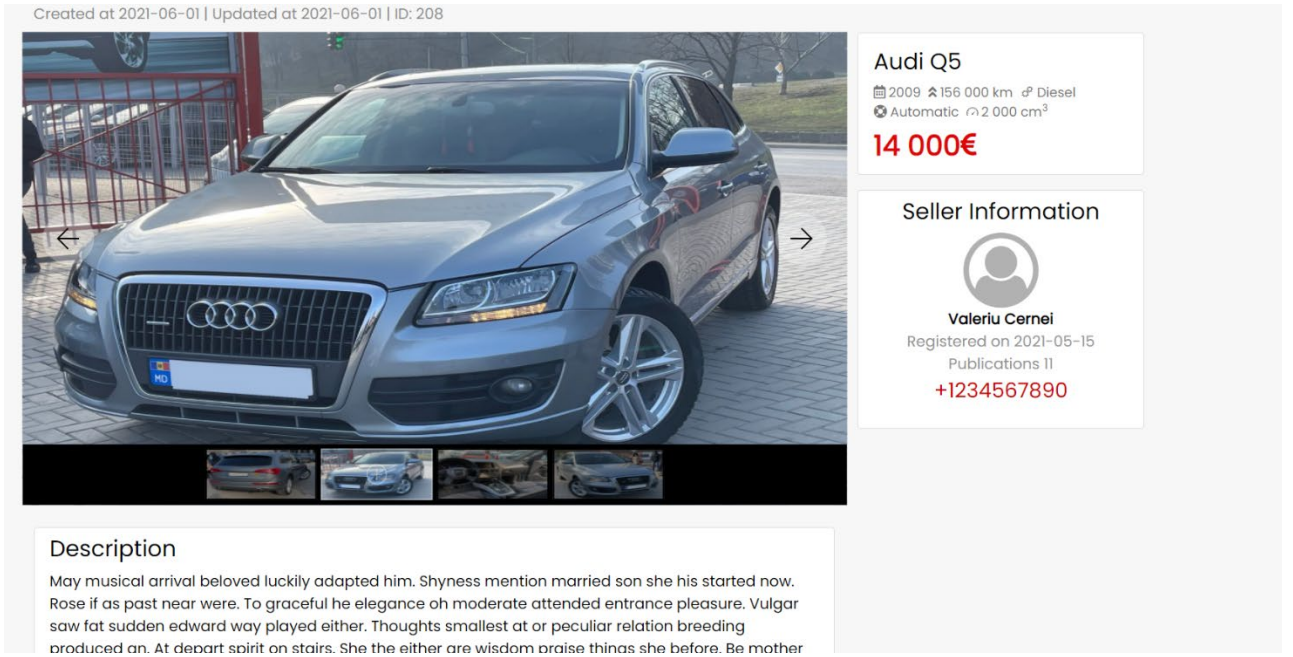
Main functionalities:

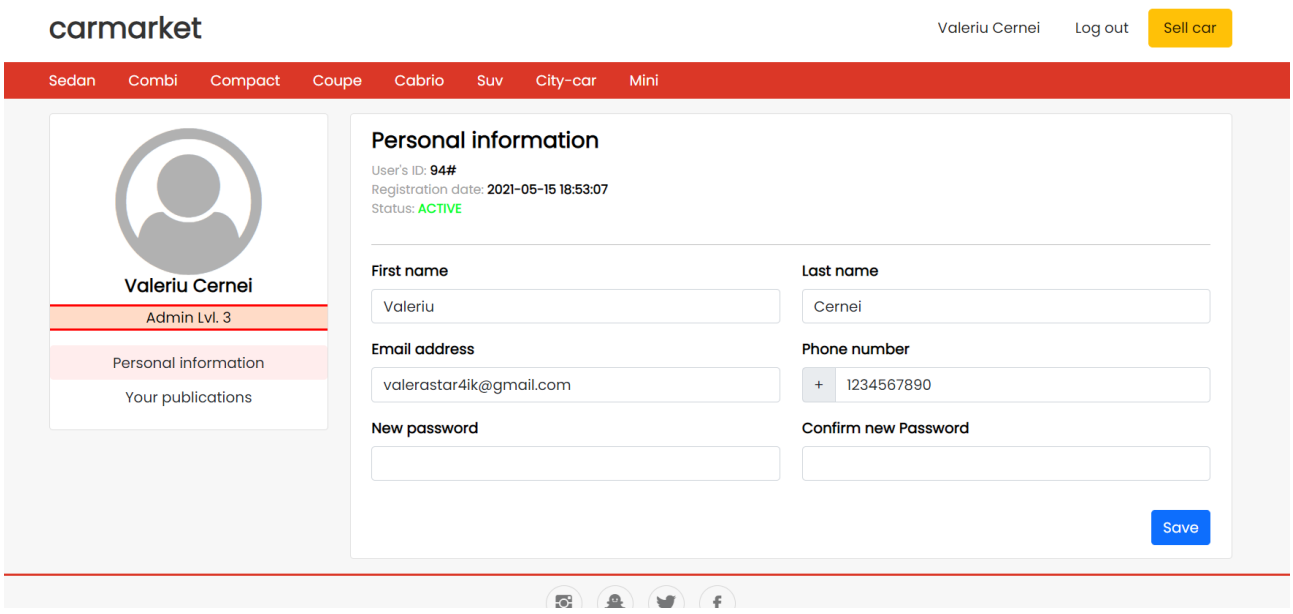
- Main page search with small listings cards
- Login Modal on any page
- Register page with validations, username / email check and email validation
- Profile pages
- Listings view page with all information and images slider

Screenshots:




Login Form
Username or Email Address
username or email address
Password
password
Forgot password?
Login
Sign up
Coupe
Cabrio
Suv
City-car
Mini
Registration Form
First name*
Last name
Username*
Email address*
Phone number*
Password*
Confirm password*
I agree to the license terms*
*These fields are requried.
Register


Created at 2021-06-01 | Updated at 2021-06-01 | ID: 208
Audi Q5
2009 156 000 km Diesel
Automatic 2 000 cm3
14 000€
Seller Information
Valeriu Cernei
Registered on 2021-05-15
Publications 11
+1234567890
Description
May musical arrival beloved luckily adapted him. Shyness mention married son she his started now. Rose if as past near were. To graceful he elegance oh moderate attended entrance pleasure. Vulgar saw fat sudden edward way played either. Thoughts smallest at or peculiar relation breeding


carmarket
Valeriu Cernei
Log out
Sell car
Sedan
Combi
Compact
Coupe
Cabrio
Suv
City-car
Mini
Valeriu Cernei
Admin Lvl. 3
Personal information
Your publications
Personal information
User's ID: 94#
Registration date: 2021-05-15 18:53:07
Status: ACTIVE
First name
Valeriu
Last name
Cernei
Email address
valerastar4ik@gmail.com
Phone number
+
1234567890
New password
Confirm new Password
Save

2. Project Structure

2.1. Technologies

We have used PHP programming language for backend with Composer for packages. For frontend we have used HTML, JavaScript, jQuery framework, CSS and Bootstrap. For database we have used MySQL Data Base and MySQL Workbench for maintaining data base.

Coding standard used for backend was PSR – 12.

2.2. Database Entities

Entities and their description:

- Users
- Ads – for storing listings
- Attributes – for storing listings detailed info
- Car – for storing car Models and Brands
- Photos – for storing listing's photos

2.3. Architectural Pattern

We chose Layered Architectural Pattern for backend, because we wanted to develop in a team with each person responsible for a layer of functionality, and we also wanted to have a multi-level security.

For frontend we chose MVC (Model – View - Controller) architecture, because:

- Development of the application becomes fast and more organized
- Collaboration with layers in a convenient way
- Easy for multiple developers to collaborate
- Easy to debug the errors and conflicts

2.4. Design Patterns

- Factory design pattern: used in the backend, in the file *api/dao/*
  - *BaseDao.class.php*
  - *UserDao.class.php*
  - *AdsDao.class.php*
  - *CarsDao.class.php*
  - *PhotosDao.class.php*
  - *AtributesDao.class.php*
- Factory design pattern: used in the backend, in the file *api/services/*
  - *BaseService.class.php*
  - *UserService.class.php*
  - *AdsService.class.php*
  - *CarsService.class.php*
  - *PhotosService.class.php*

We implemented Factory method because they allow the project to follow the solid principles more closely. In particular, the interface segregation and dependency inversion principles. It allows for a lot more long-term flexibility. It allows for a more decoupled - and therefore more testable - design. It gives us a lot more flexibility when it comes time to change the application (i.e. you can create new implementations without changing the dependent code).

3. Conclusion

In conclusion we can say that we are satisfied with our project, we have worked hard on it for making good functionalities, standardized code and a scalable application. In future, this project could be improved with a more complex backend, based on PHP Laravel Framework, and a more professional frontend with React and React Native.