

**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА**

**ФАКУЛТЕТ ПО ИЗЧИСЛИТЕЛНА ТЕХНИКА И АВТОМАТИЗАЦИЯ**

**КАТЕДРА „СОФТУЕРНИ И ИНТЕРНЕТ ТЕХНОЛОГИИ“**

**Специалност „Софтуерни и интернет технологии“**

**Курсов проект по „Технология на софтуерното производство“**

**Система за продажба на нови и употребявани автомобили**

**Изготвили:**

Владимир Викторов Андреев и Александър Мирославов Живков

(Спец. СИТ, гр. 4б, Фак. №: 19621703, 19621702)

**ВАРНА**

**2022**

Разработващ екип:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Име, презиме и фамилия | Фак. № | Роля |
| 1. | Александър Мирославов Живков | 19621702 | Ръководител на проекта, Back-End Developer |
| 2. | Владимир Викторов Андреев | 19621703 | Front-End Developer |
| 3. | Александър Мирославов Живков,  Владимир Викторов Андреев | 19621702,  19621703 | Разработка на база данни |
| 4. | Александър Мирославов Живков,  Владимир Викторов Андреев | 19621702,  19621703 | Изготвяне на документация към проекта |

Срок за представяне: 10 май, 2022г.

Водещ преподавател:

/доц.В.Божикова/

1. **Задание за разработка**

ЗАДАНИЕ

**за курсов проект по дисциплината**

**"Софтуерни технологии"/”Технология на софтуерното производство”**

**Цел на курсовия проект:**

Изисква се да се разработи „Система за продажба на нови и употребявани превозни средства” .

**Функционални изисквания:**

* Софтуерът трябва да позволи на служител на представителство на марка или автокъща да въвежда записи за различни видове превозни средства, да добавя нови такива, както и да редактира стари и да ги изтрива. Целта на софтуера е да улесни ежедневния работен процес на служителите.
* Освен това, тази система трябва да улесни смяната на собствеността на дадено превозно средство, както и воденето на справки за собствениците, закупили превозни средства от даденото представителство.
* Софтуерът поддържа добавяне, редактиране и изтриване на сервизни обслужвания за всяко превозно средство, както и на механици, работещи в даденото представителство.
* Системата трябва да съхранява данни (в БД) за:
* Потребители (Users),
* собственици (Owners)
* плащания (Payments)
* превозни средства (Cars)
* последно сервизно обслужване (Services) и
* механици (Mechanics).

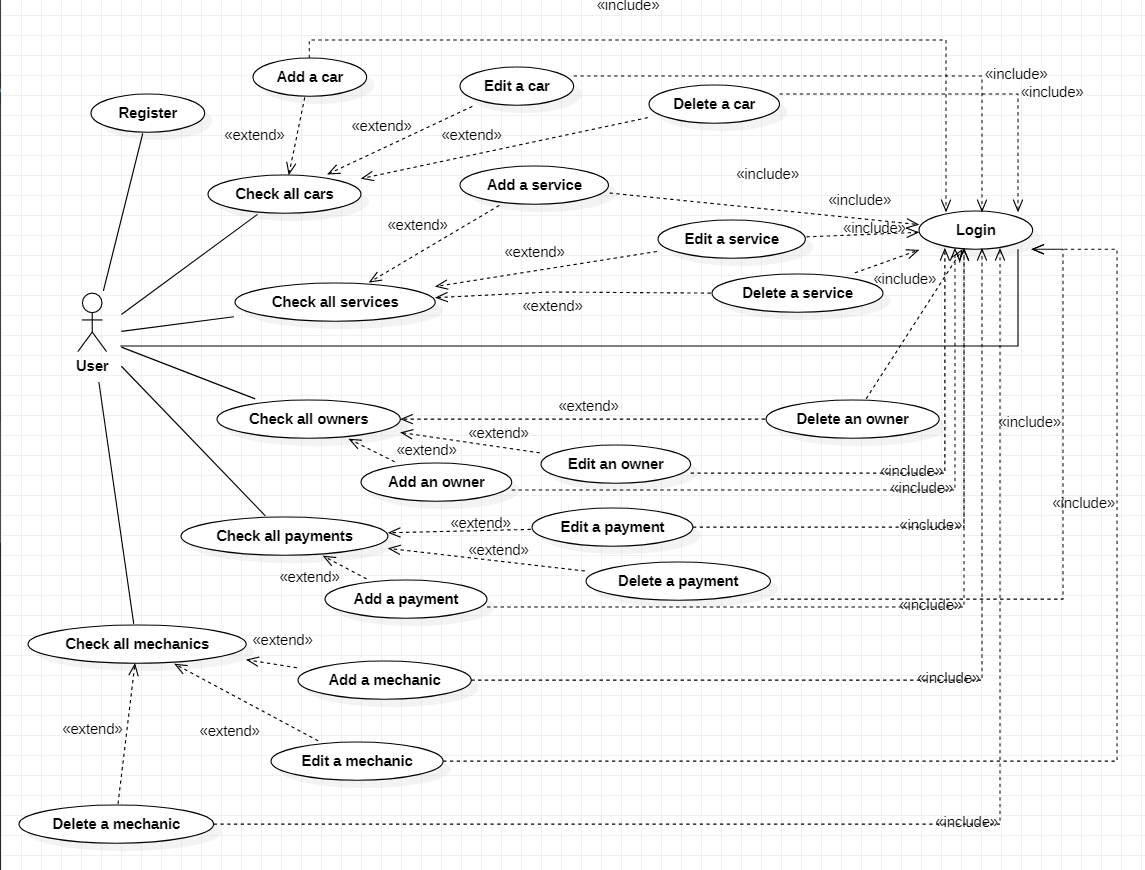
**Не-функционални изисквания:**

* Използва се унифицирана форма за логване в системата от всички потребители.
* За всяка от таблиците в БД се използва унифицирана форма за реализиране на стандартните операции: въвеждане, корекция, изтриване и търсене на данни.
* Генериране на необходимите справки: на екран.
* Потребителите могат да нанасят корекции по записите само и единствено ако са логнати в системата.
* За въвеждането на всеки един запис е необходимо данните да отговарят на предварително посочени изисквания.

1. **Реализация на проекта:**

**Проектиране :**

***Use case diagram:***



***Models Class Diagram:***

Diagram

Description automatically generated

***ERD Diagram:***

Graphical user interface, diagram

Description automatically generated with medium confidence

*Създават се 5 таблици: плащания, обслужвания, коли, собственици, механици. Таблицата AspNetUsers е автоматично генерирана от Asp Identity. За една кола може да има едно плащане, едно сервизно обслужване и един собственик, всяко обслужване се извършва от един механик, като един механик може да има много сервизни акции. Един собственик може да има няколко автомобила, както и няколко автомобила могат да имат един и същи вид плащане.*

***Съдържание на таблиците:***

* ***Cars:***
  + *CarId (primary key, autoincrement)*
  + *Manufacturer – марка на колата*
  + *Model – модел*
  + *Engine – двигател*
  + *Transmission – скоростна кутия*
  + *DriveType – вид задвижване*
  + *Vin – номер на рама*
  + *Price – цена*
  + *DateOfFirstReg – дата на първа регистрация*
  + *Mileage – пробег*
  + *OwnerId – Id на собственика (foreign key)*
  + *PaymentId – Id на вида плащане (foreign key)*
  + *Discount – отстъпка за лоялни клиенти в %*
  + *VehicleType – вид купе*
  + *ServiceId – Id на последното обслужване (foreign key)*
* ***Owners:***
  + *OwnerId (primary key, autoincrement)*
  + *Name – име на собственика*
  + *CarsBought – брой закупени коли от представителството*
* ***Payments:***
  + *PaymentId – (primary key, autoincrement)*
  + *Type – вид на плащането*
* ***Mechanics:***
  + *MechanicId – (primary key, autoincrement)*
  + *Name – име на механика*
* ***Services:***
  + *ServiceId – (primary key, autoincrement)*
  + *Name – име на обслужването*
  + *Type – вид обслужване*
  + *Price – цена на обслужването*
  + *MechanicId – Id на механика, извършил обслужването (foreign key)*

**Програмиране:**

***Структура на проекта:***

* **wwwroot** - В тази папка се намират всички статични елементи като css, снимки и др.;
* **Аreas** – Контролери, модели и изгледи за логин и регистрация (ASP Identity);
* **Controllers** - Контролери за превозни средства, обслужвания, собственици, механици и плащания. Съдържат get и post методи за създаване, редактиране, изтриване на данни, както и методи за търсене (филтриране) по зададени параметри;
* **Data** - Съдържа DbContext-a на приложението;
* **Migrations** – Съдържа миграциите, на базата на които се създава базата данни в SQL Server чрез Entity Framework
* **Models** - Модели , на базата на които се генерират таблиците в базата;
* **Views** - Изгледите на всяка една страница от приложението;

***Метод за създаване или редактиране на кола (POST заявка):***

[HttpPost]

[ValidateAntiForgeryToken]

[Authorize]

public async Task<IActionResult> AddOrEdit([Bind("CarId,Manufacturer,Model,Engine,Transmission,DriveType,Vin,Price,DateOfFirstReg,Mileage,OwnerId,PaymentId,Discount,VehicleType,ServiceId")] Car car)

{

Owner owner = \_context.Owners.Find(car.OwnerId);

if(car.CarId == 0) //if the car is new

{

if(!owner.Name.Equals("No owner")) // if the new car has a owner

{

owner.CarsBought++;

\_context.Update(owner);

}

car.Price -= (car.Price \* car.Discount)/100;

\_context.Add(car);

}

else // if the car isn't new (edit)

{

if (!owner.Name.Equals("No owner")) //if the car has a owner

{

owner.CarsBought++; //check to see if this is regarding the new owner or the old one

\_context.Update(owner);

}

car.Price -= (car.Price \* car.Discount)/100;

\_context.Update(car);

}

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

**Метод за зареждане на данните на кола (GET заявка):**

[Authorize]

public IActionResult AddOrEdit(int id)

{

if (id == 0) // new car - empty fields

{

ViewData["OwnerId"] = new SelectList(\_context.Owners, "OwnerId", "Name");

ViewData["PaymentId"] = new SelectList(\_context.Payments, "PaymentId", "Type");

ViewData["ServiceId"] = new SelectList(\_context.Services, "ServiceId", "Name");

return View(new Car());

}

else //edit a car

{

ViewData["OwnerId"] = new SelectList(\_context.Owners, "OwnerId", "Name");

ViewData["PaymentId"] = new SelectList(\_context.Payments, "PaymentId", "Type");

ViewData["ServiceId"] = new SelectList(\_context.Services, "ServiceId", "Name");

return View(\_context.Cars.Find(id));

}

}

**Реализация на поле за търсене на коли:**

public IEnumerable<Car> GetSearchResults(String search)

{

var searchByOwnerName = \_context.Cars.Include(c => c.Owner).Include(c => c.Payment).Include(c => c.Services)

.Where(t => t.Owner.Name.Equals(search)).ToList(); //checks if search param is for owner

if (searchByOwnerName.Count > 0)

{

return searchByOwnerName;

}

var searcrByCarManufacturer = \_context.Cars.Include(c => c.Owner).Include(c => c.Payment).Include(c => c.Services)

.Where(t => t.Manufacturer.Equals(search)).ToList();//checks if search param is for manufacturer

if (searcrByCarManufacturer.Count > 0)

{

return searcrByCarManufacturer;

}

var priceToSearch = 0.0; //var for price checking

try

{

priceToSearch = Double.Parse(search); //exception handling for parsing

}

catch (ArgumentNullException ane) { }

catch (FormatException fe) { }

catch (OverflowException oe) { }

var searcrByPrice = \_context.Cars.Include(c => c.Owner).Include(c => c.Payment).Include(c => c.Services)

.Where(t => t.Price >= priceToSearch).ToList();//check if search param is for price

if (searcrByPrice.Count > 0)

{

return searcrByPrice;

}

return null;

}

***Метод за визуализация на всички записи на коли:***

public async Task<IActionResult> Index(String search)

{

var applicationDbContext = \_context.Cars.Include(c => c.Owner).Include(c => c.Payment).Include(c => c.Services);

if (String.IsNullOrEmpty(search)) //if nothing is typed in the search bar

{

return View(await applicationDbContext.ToListAsync()); //returns all entries in DB

}

else if(GetSearchResults(search)!=null)

{

return View(GetSearchResults(search));

}

else return View(await applicationDbContext.ToListAsync());//default

}

***Фрагмент от Views/Cars/Index.cshtml за реализация на поле за търсене:***

<**form** **asp-action**="Index">

<div class="input-group md-form form-sm form-2 p1-0">

<input name="search" class="form-control my-0 py-1 red-border" type="text" placeholder="Search by manufacturer/owner/price" aria-label="Search" />

<div class="input-group-append">

<span class="input-group-text red lightne-3" id="searchRes">

<i class="fas fa-search text-grey" style="padding-bottom:5px;" aria-hidden="true"></i>

</span>

</div>

</div>

</**form**>

@section scripts{

<script>

$ (function(){

$("#searchRes").click(function () {

$("Form").submit();

})

});

</script>

}

***Метод за изтриване на запис на кола (POST заявка):***

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

[Authorize]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var car = await \_context.Cars.FindAsync(id);

Owner owner = \_context.Owners.Find(car.OwnerId);

\_context.Cars.Remove(car);

if (!owner.Name.Equals("No owner"))

{

owner.CarsBought--;

\_context.Update(owner);

}

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

***Метод за реализация на търсене на обслужване:***

public IEnumerable<Service> GetSearchResults(String search)

{

var searchByServiceType = \_context.Services.Include(c => c.Mechanics).Where(t => t.Type.Equals(search)).ToList();

if (searchByServiceType.Count > 0)

return searchByServiceType;

var searchServiceByMechanicName = \_context.Services.Include(c => c.Mechanics).Where(t => t.Mechanics.Name.Equals(search)).ToList();

if (searchServiceByMechanicName.Count > 0)

return searchServiceByMechanicName;

var searchCriteria = 0.0;

try

{

searchCriteria = double.Parse(search);

}

catch (Exception ex) { }

var searchByServicePrice = \_context.Services.Include(c => c.Mechanics).Where(t => t.Price>=searchCriteria).ToList();

if (searchByServicePrice.Count > 0)

return searchByServicePrice;

return null;

}

***Метод за визуализация на всички записи на обслужвания:***

public async Task<IActionResult> Index(String search)

{

var applicationDbContext = \_context.Services.Include(c => c.Mechanics);

if (String.IsNullOrEmpty(search)) //if nothing is typed in the search bar

{

return View(await applicationDbContext.ToListAsync()); //returns all entries in DB

}

if (GetSearchResults(search) != null)

{

return View(GetSearchResults(search));

}

return View(await applicationDbContext.ToListAsync());

}

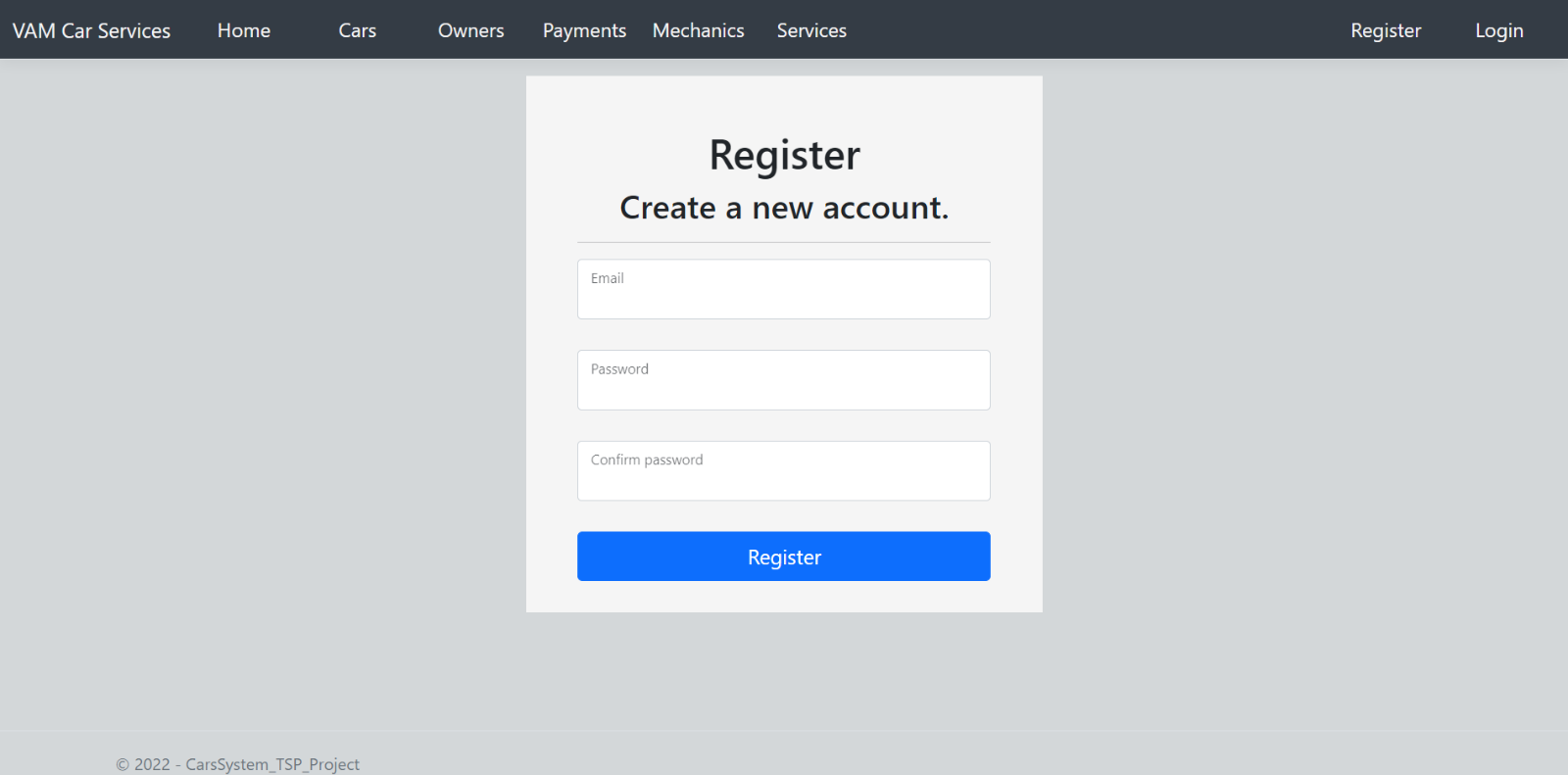
***Реализацията на полето за търсене в Views/Services/Index.cshtml е аналогична на тази от Views/Cars/Index/cshtml.***

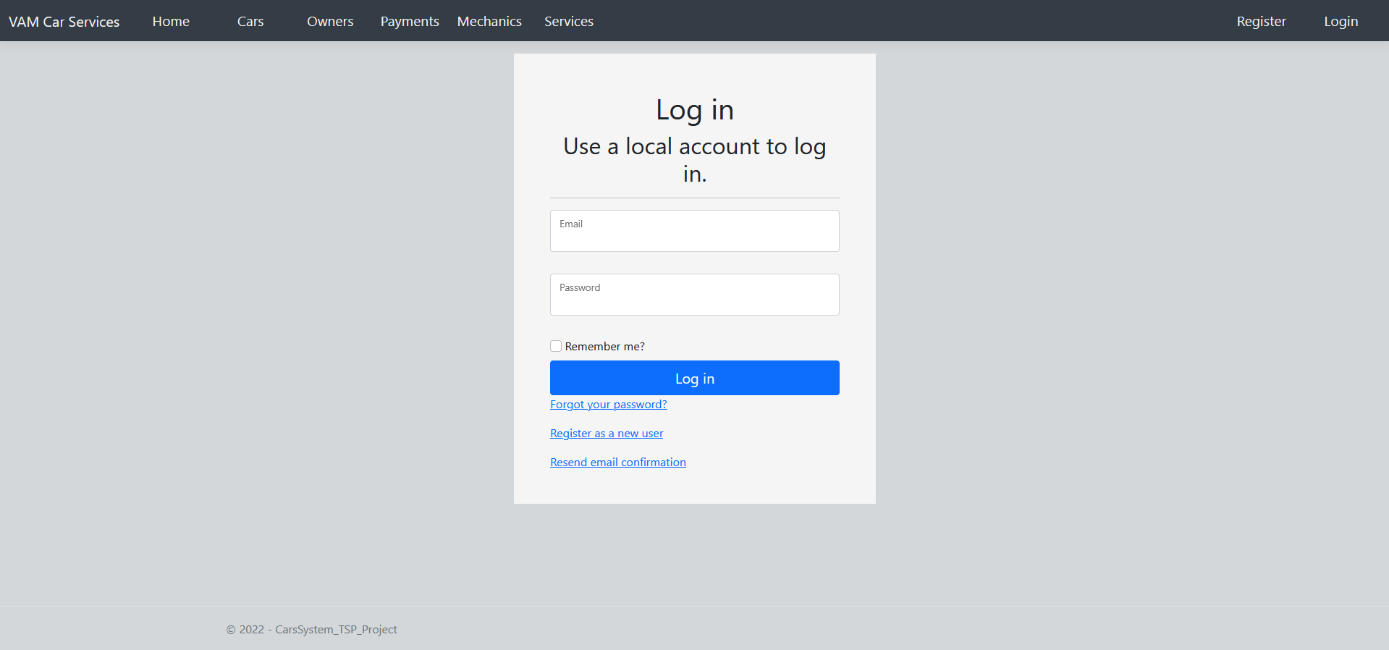
***Реализацията на AddOrEdit/Delete за останалите таблици е направена по аналогичен начин!***

1. **Ръководство за потребителя.**

При първоначално зареждане на приложението се зарежда началната страница с информация за компанията. В навигационната лента са поместени различните секции, които чрез кликване от страна на потребителя, той може да достъпи, а именно : Cars(коли), Owners(собственици), Payments(плащания), Mechanics(механици), Services(обслужвания). В същата лента са поместени и бутоните за регистрация и логин. За регистриране се попълват съответните полета за имейл, парола и потвърждение на паролата, след което потребителят може да се логне в системата, за да има достъп до пълната и функционалност.

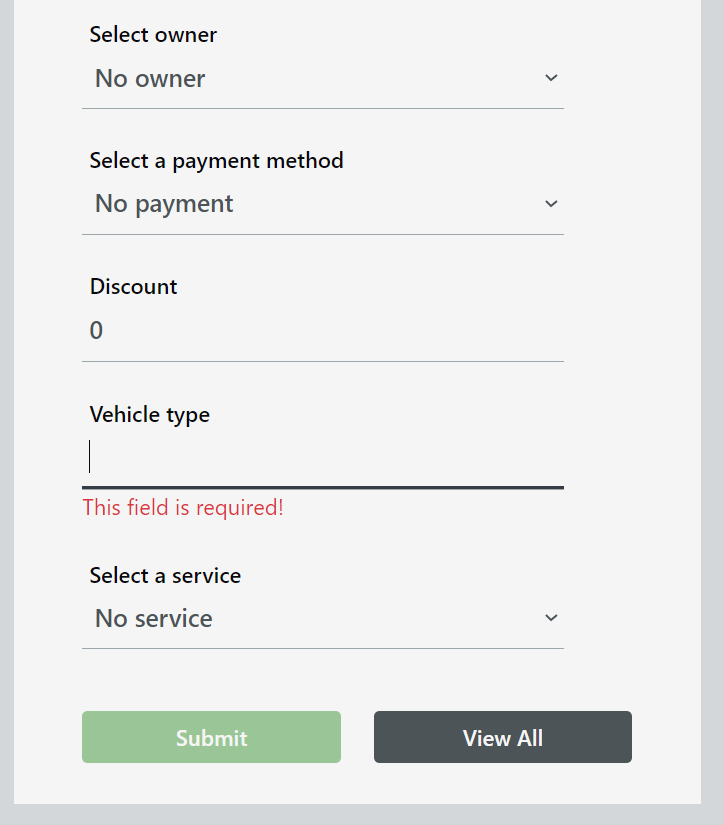




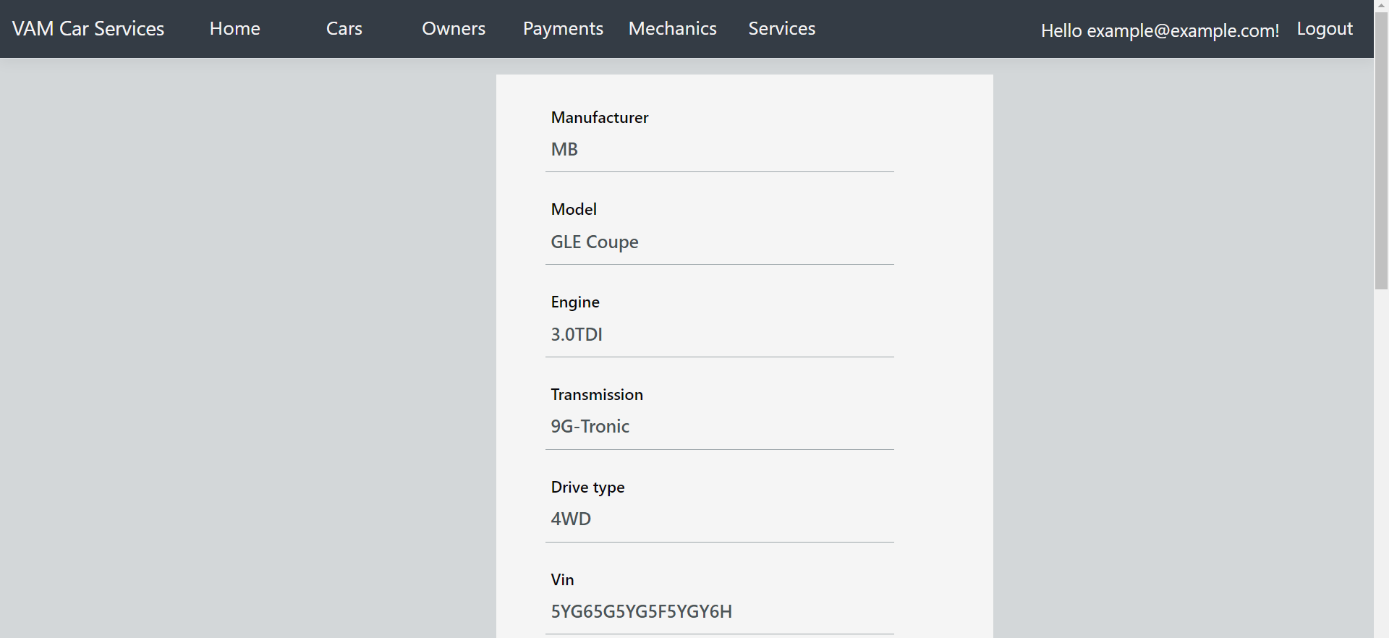


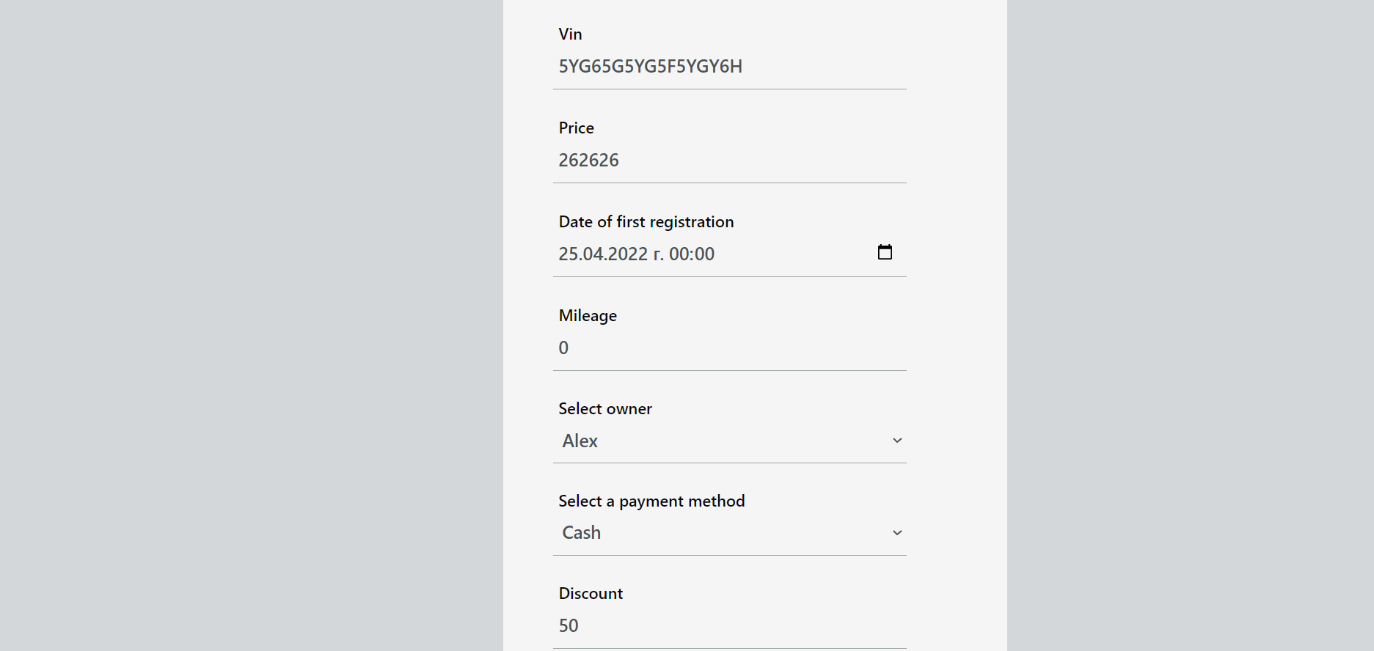
При натискане на секцията Cars, се зарежда списък в табличен вид със всички налични записи за превозни средства в базата данни. Над тях е поместено поле за търсене, чрез което може да се филтрират показаните записи по съответните критерии : производител/собственик/цена. След въвеждане и натискане на enter бутон от клавиатурата или лупата до полето за търсене, резултатите се изобразяват. При повторно натискане (когато полето е празно), на екрана отново се извеждат всички налични записи в базата данни.

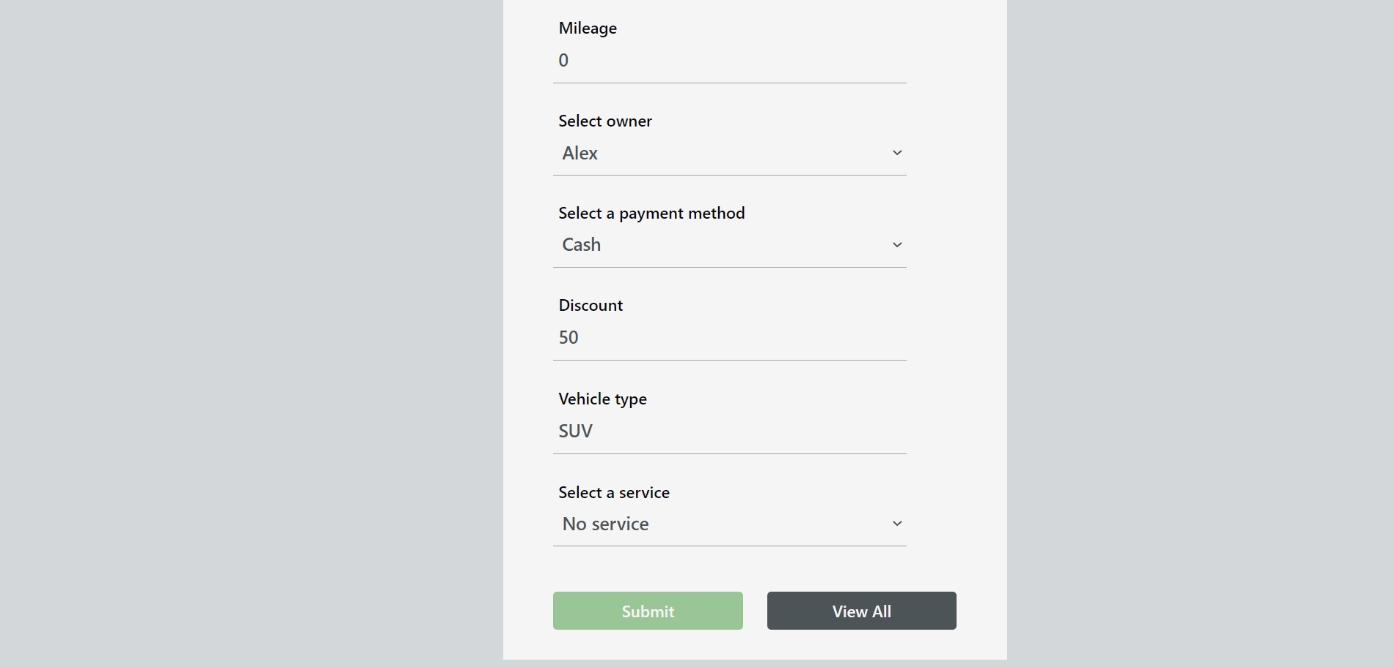
За добавяне на нов запис е необходимо да се натисне зеления бутон „+“. След това се зарежда страница, в която трябва да се попълнят данните на новото превозно средство. По време на въвеждането, при некоректност на формата на попълнените данни, се изобразяват подсказващи съобщения. След успешно въвеждане на данните спрямо критериите, се натиска зеления бутон Submit, с което записът се запазва. За връщане назад по време на въвеждането се натиска бутона View All.



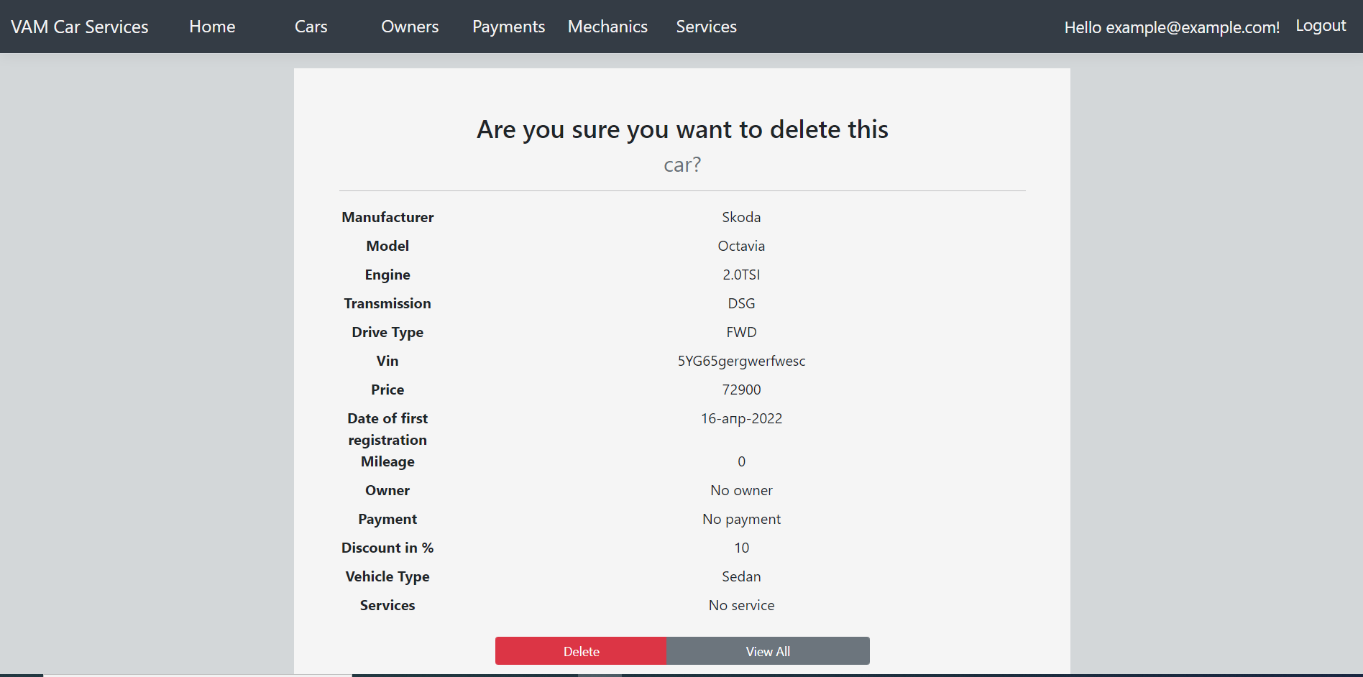
За редактиране на запис е необходимо да се натисне жълтият бутон с форма на молив. Зарежда се страница с информацията на конкретния запис и след направата на необходимите корекции, се натиска бутона Submit за запазване, или бутона View All за връщане назад.



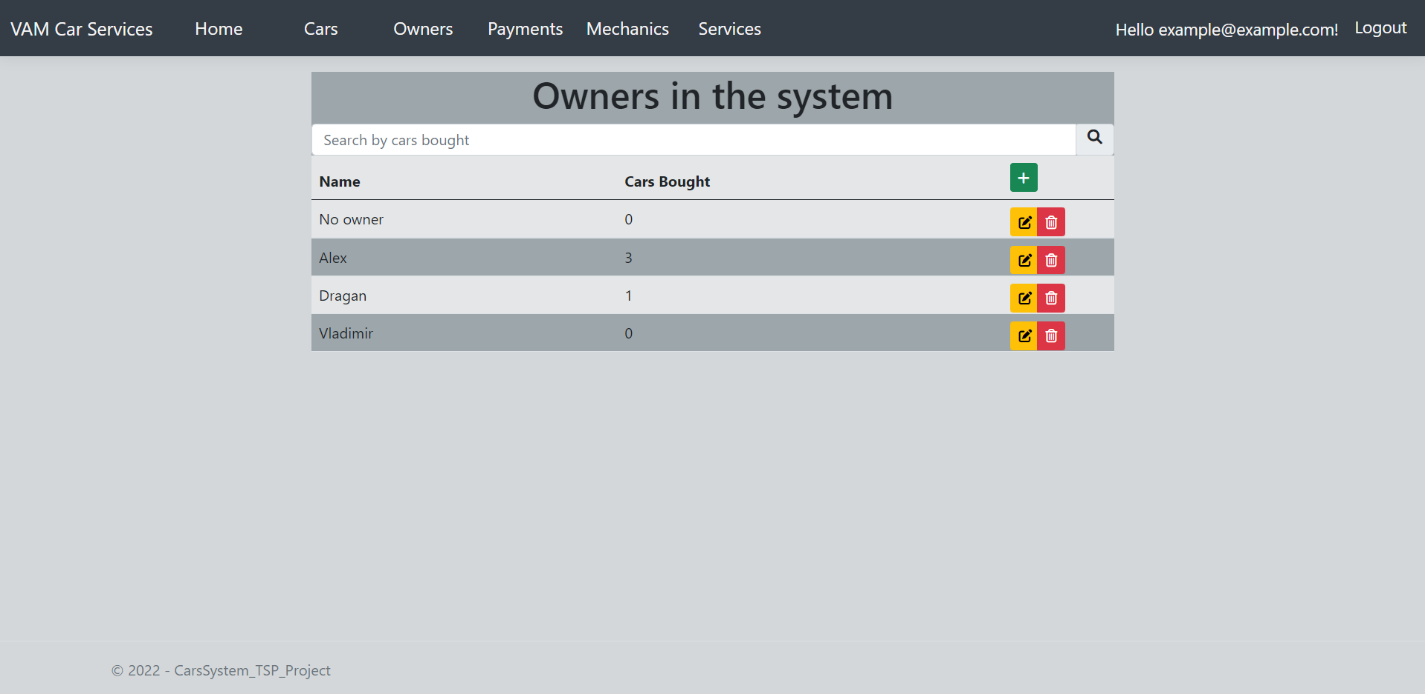


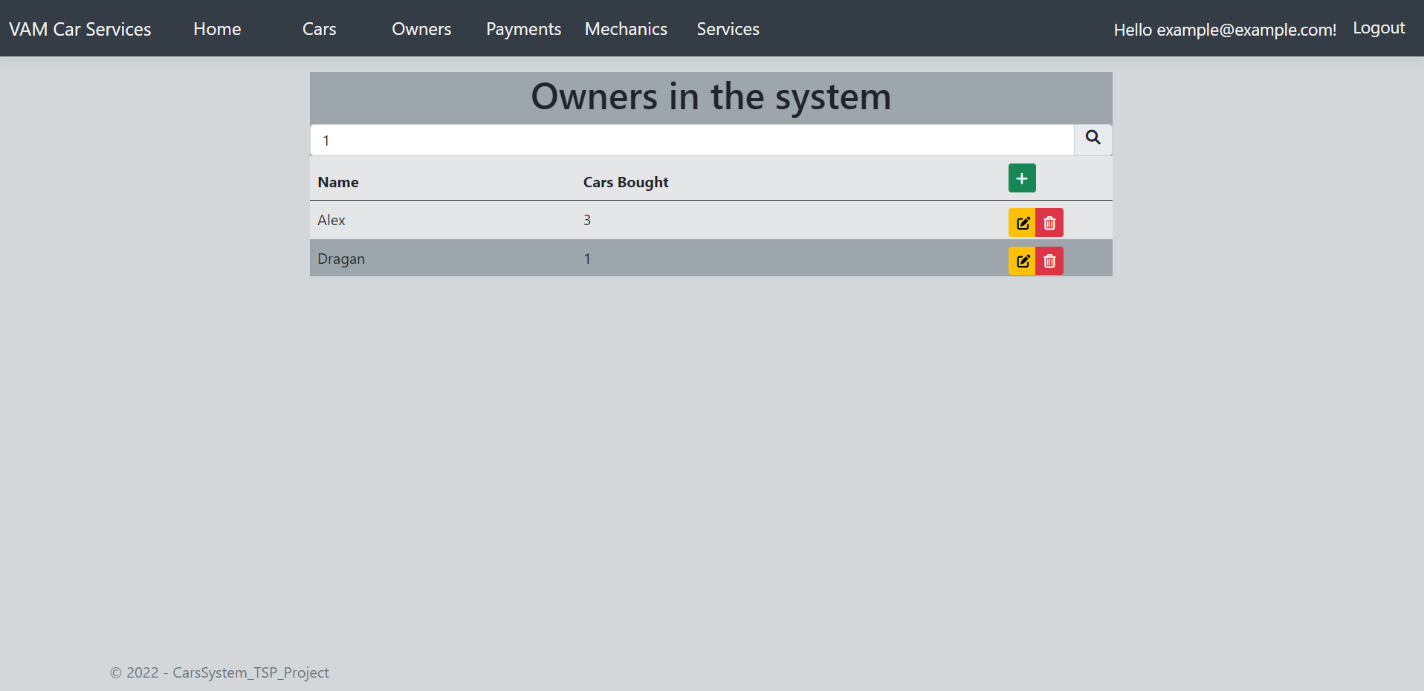


За изтриване на даден запис е необходимо да се натисне върху червения бутон с форма на кошче. След това се зарежда информацията за записа, както и два бутона – за изтриване (Delete) и връщане назад View all.

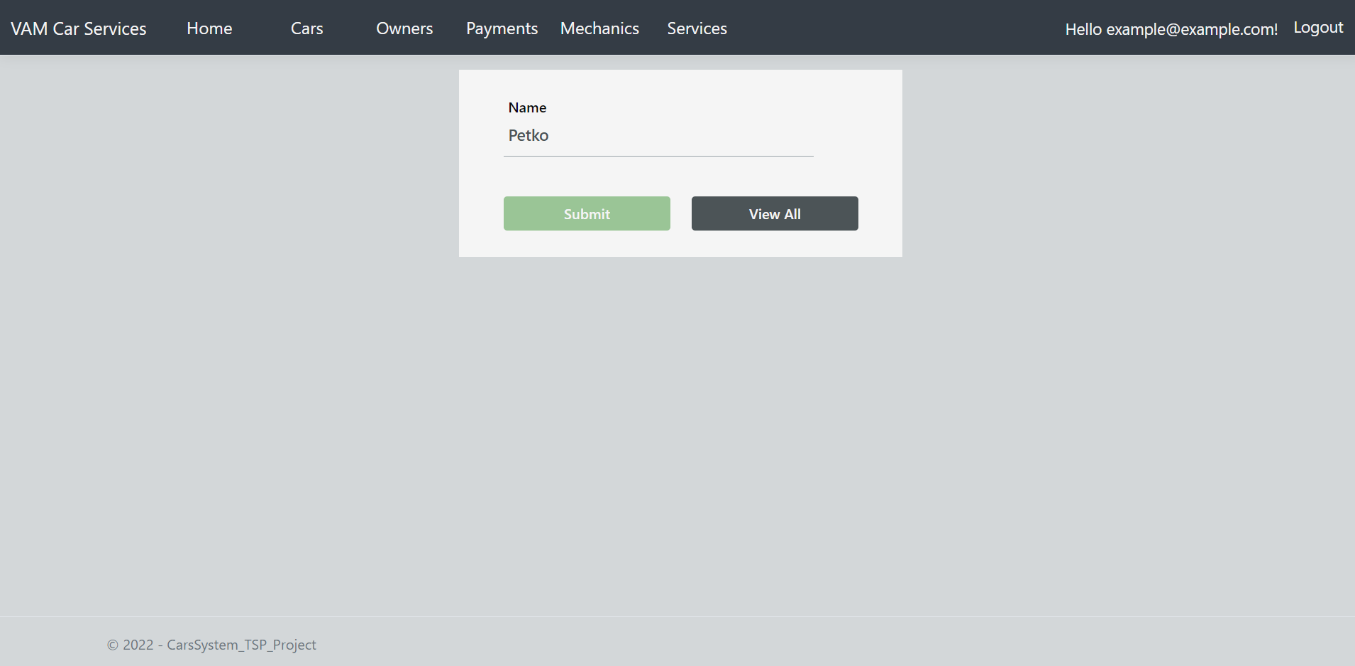


При натискане на секцията Owners, се зарежда списък в табличен вид със всички налични записи за собственици в базата данни. Над тях е поместено поле за търсене, чрез което може да се филтрират показаните записи по съответните критерии : брой закупени коли от представителството. След въвеждане и натискане на enter бутон от клавиатурата или лупата до полето за търсене, резултатите се изобразяват. При повторно натискане (когато полето е празно), на екрана отново се извеждат всички налични записи в базата данни.

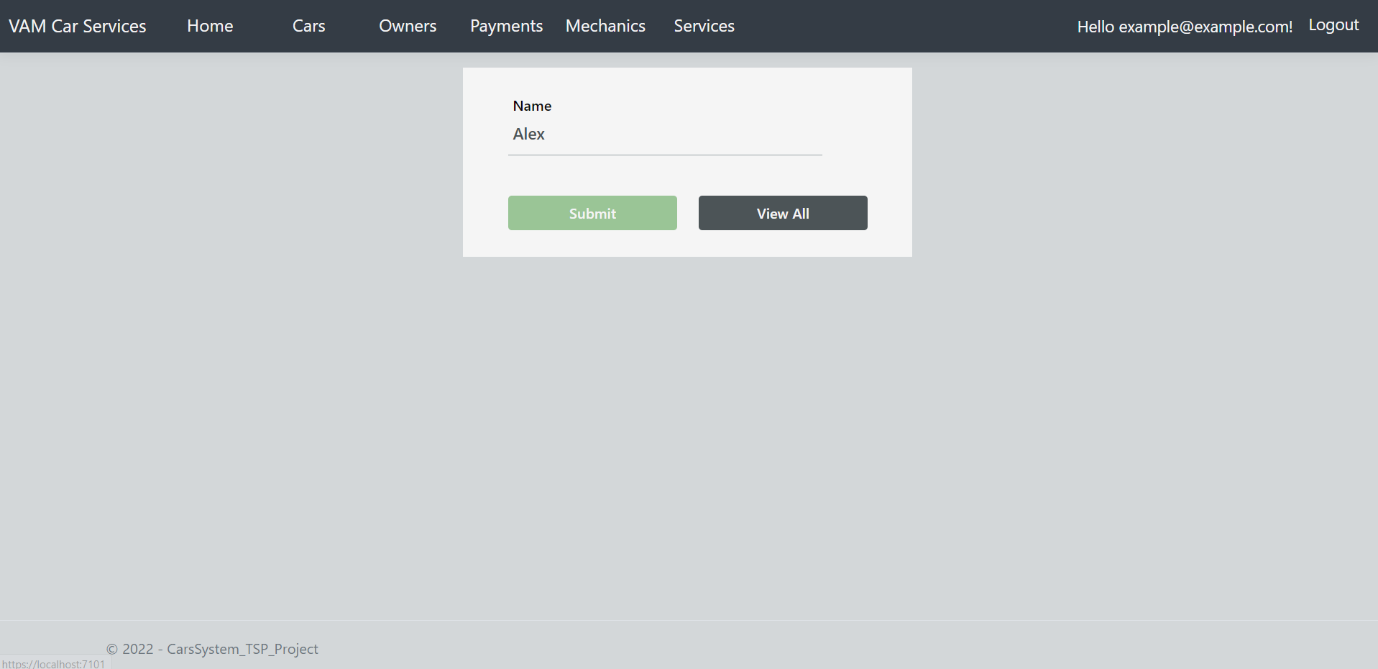




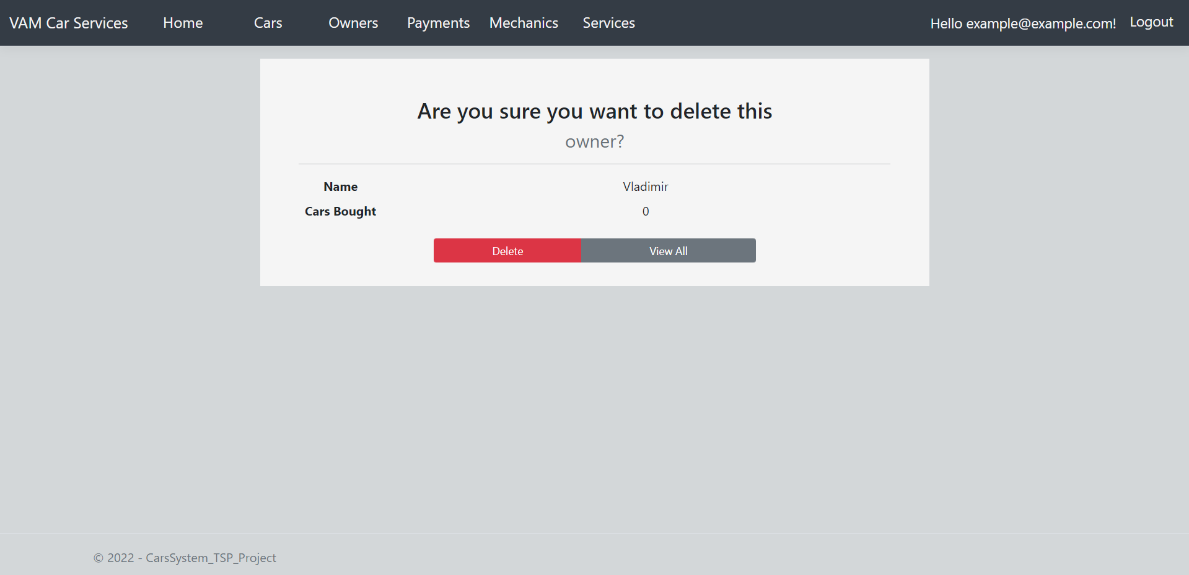
За добавяне на нов запис е необходимо да се натисне зеления бутон „+“. След това се зарежда страница, в която трябва да се попълнят данните на новият собственик. По време на въвеждането, при некоректност на формата на попълнените данни, се изобразяват подсказващи съобщения. След успешно въвеждане на данните спрямо критериите, се натиска зеления бутон Submit, с което записът се запазва. За връщане назад по време на въвеждането се натиска бутона View All.



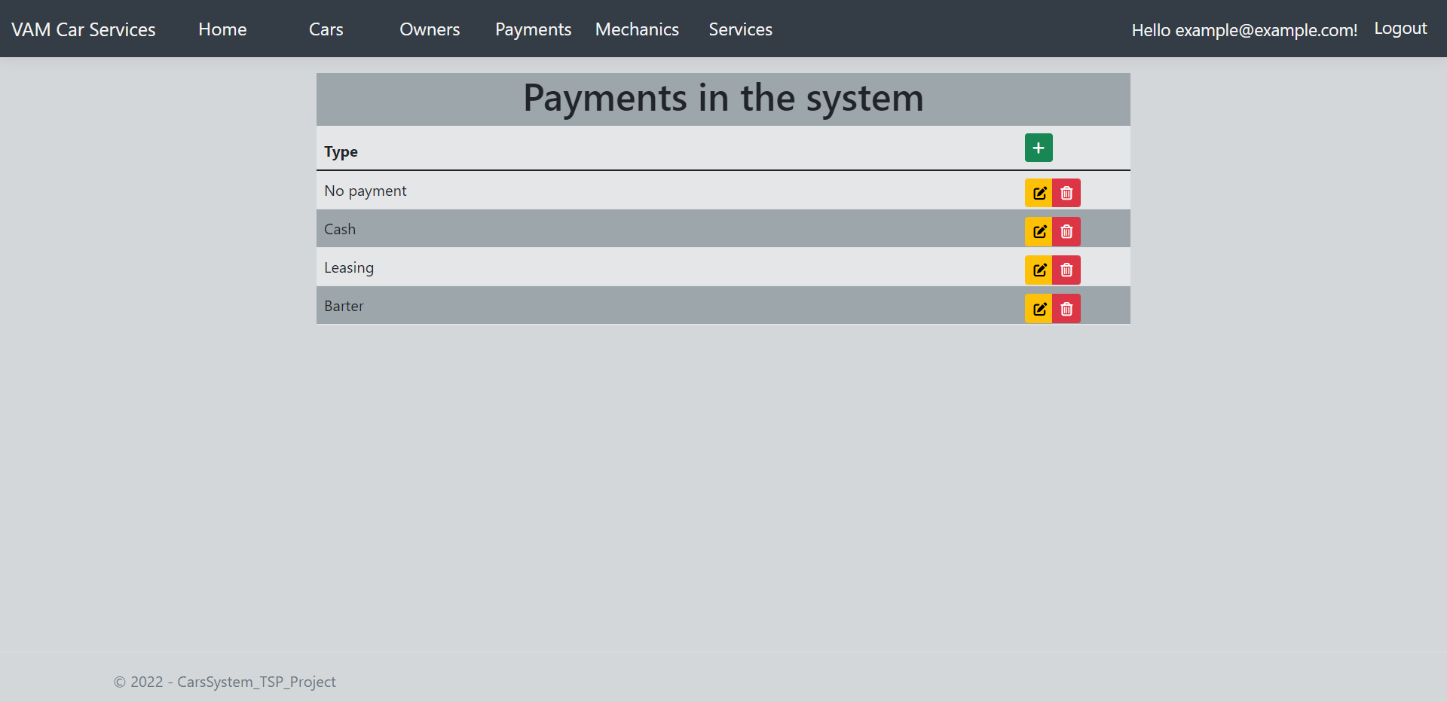
За редактиране на запис е необходимо да се натисне жълтият бутон с форма на молив. Зарежда се страница с информацията на конкретния запис и след направата на необходимите корекции, се натиска бутона Submit за запазване, или бутона View All за връщане назад.



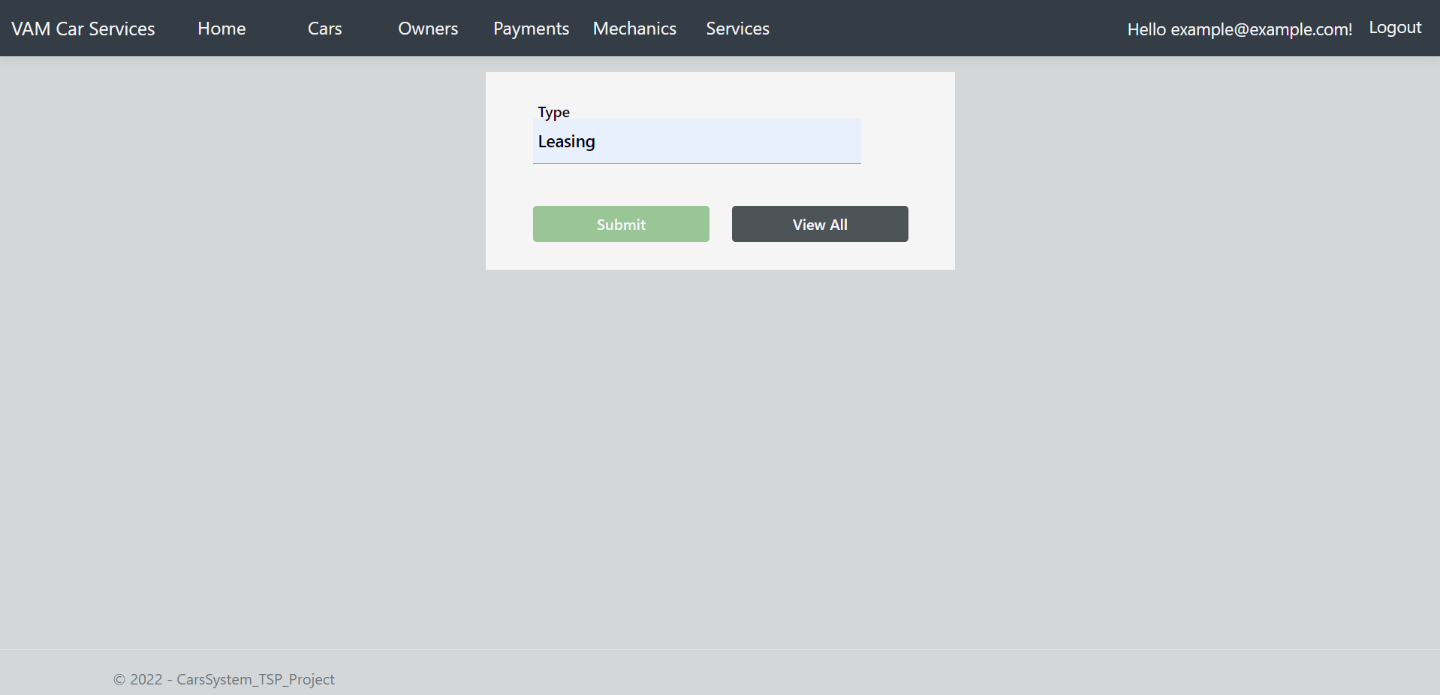
За изтриване на даден запис е необходимо да се натисне върху червения бутон с форма на кошче. След това се зарежда информацията за записа, както и два бутона – за изтриване (Delete) и връщане назад View all.



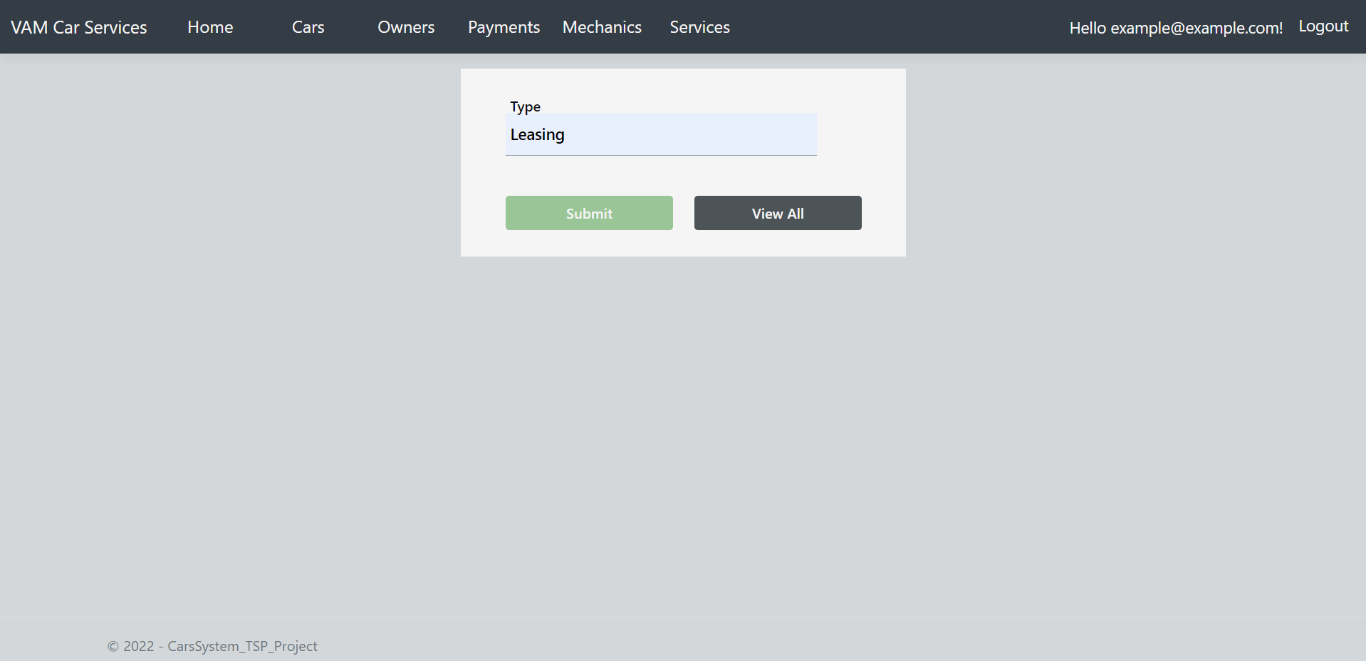
При натискане на секцията Payments, се зарежда списък в табличен вид със всички налични записи за видовете плащане в базата данни.



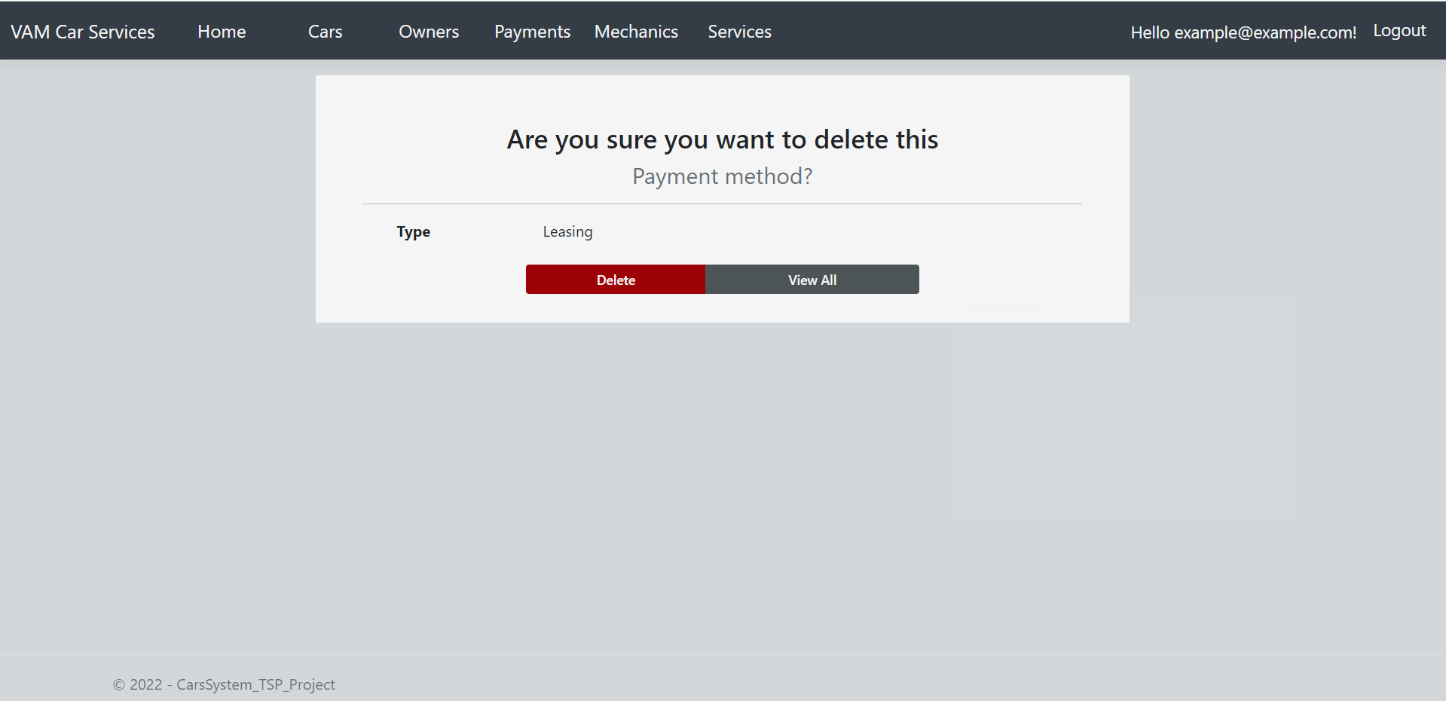
За добавяне на нов запис е необходимо да се натисне зеления бутон „+“. След това се зарежда страница, в която трябва да се попълнят данните за новият вид плащане. По време на въвеждането, при некоректност на формата на попълнените данни, се изобразяват подсказващи съобщения. След успешно въвеждане на данните спрямо критериите, се натиска зеления бутон Submit, с което записът се запазва. За връщане назад по време на въвеждането се натиска бутона View All.



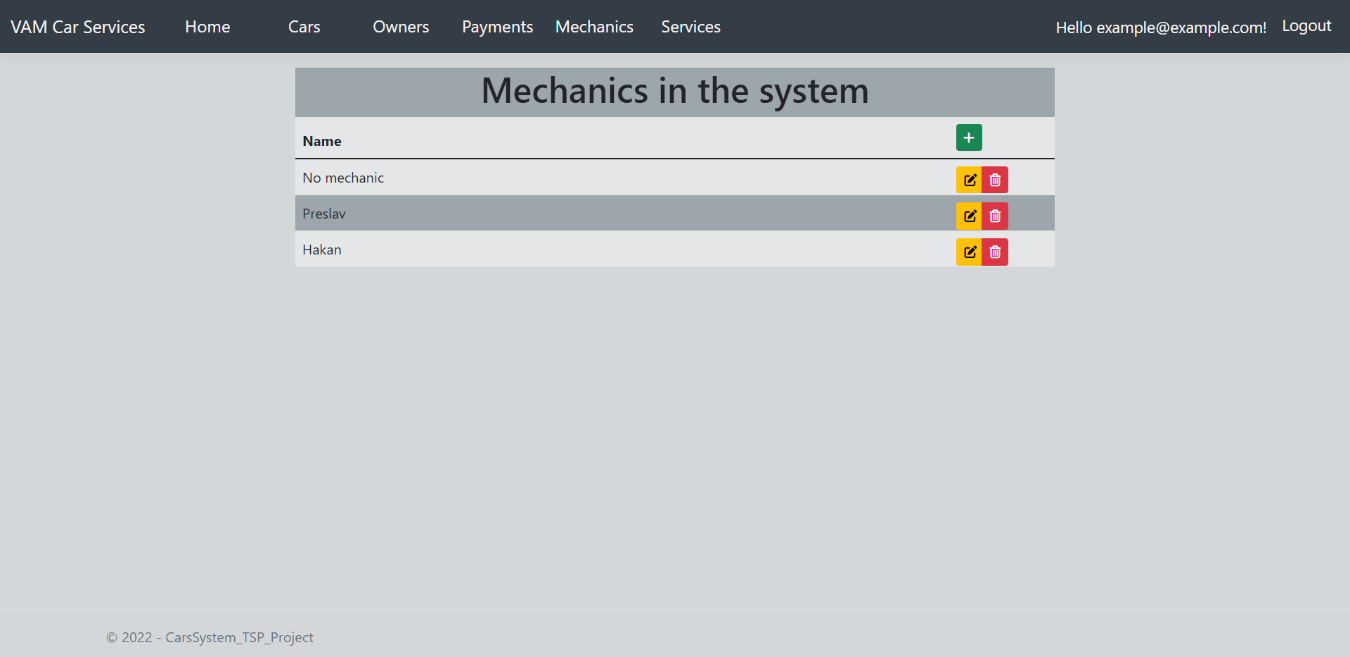
За редактиране на запис е необходимо да се натисне жълтият бутон с форма на молив. Зарежда се страница с информацията на конкретния запис и след направата на необходимите корекции, се натиска бутона Submit за запазване, или бутона View All за връщане назад.



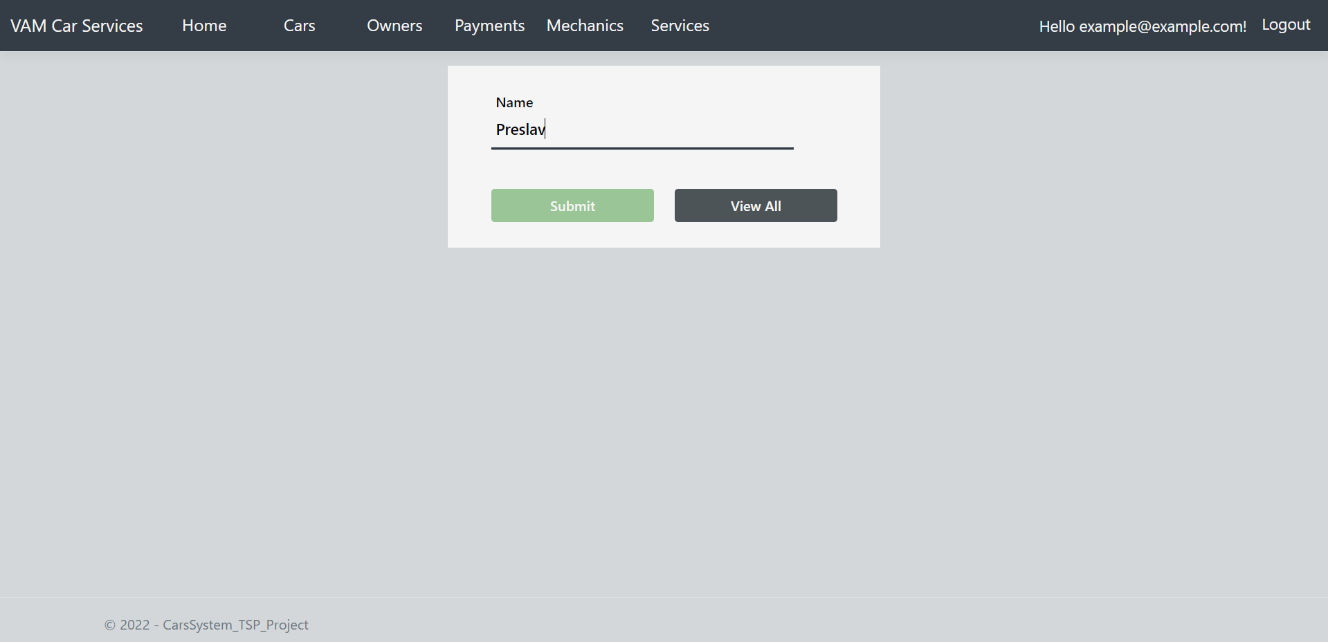
За изтриване на даден запис е необходимо да се натисне върху червения бутон с форма на кошче. След това се зарежда информацията за записа, както и два бутона – за изтриване (Delete) и връщане назад View all.



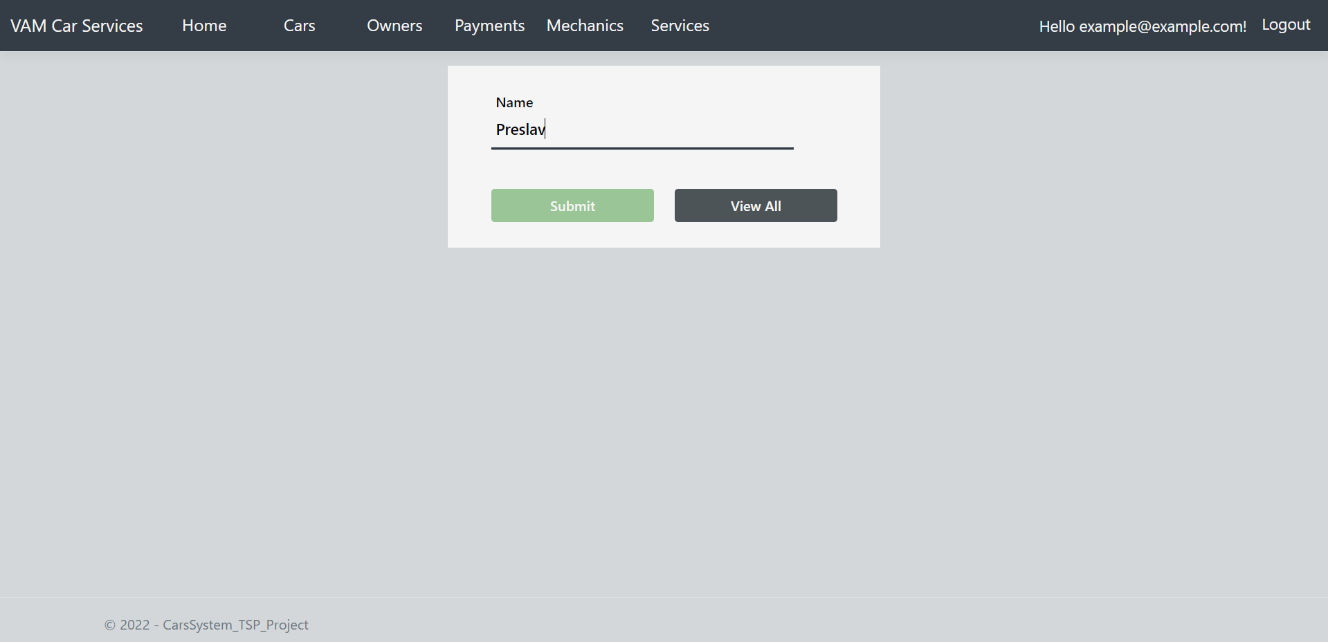
При натискане на секцията Mechanics, се зарежда списък в табличен вид със всички налични записи на механиците, работещи в представителството, в базата данни.



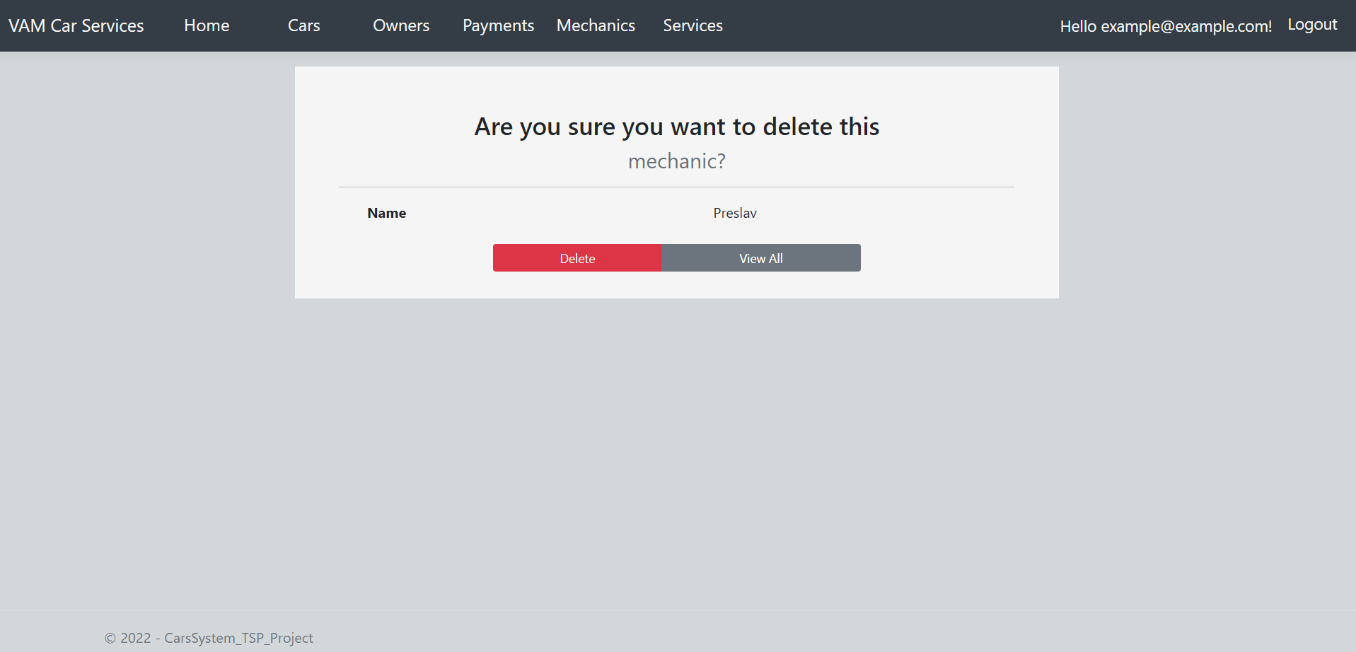
За добавяне на нов запис е необходимо да се натисне зеления бутон „+“. След това се зарежда страница, в която трябва да се попълнят данните на новият механик. По време на въвеждането, при некоректност на формата на попълнените данни, се изобразяват подсказващи съобщения. След успешно въвеждане на данните спрямо критериите, се натиска зеления бутон Submit, с което записът се запазва. За връщане назад по време на въвеждането се натиска бутона View All.



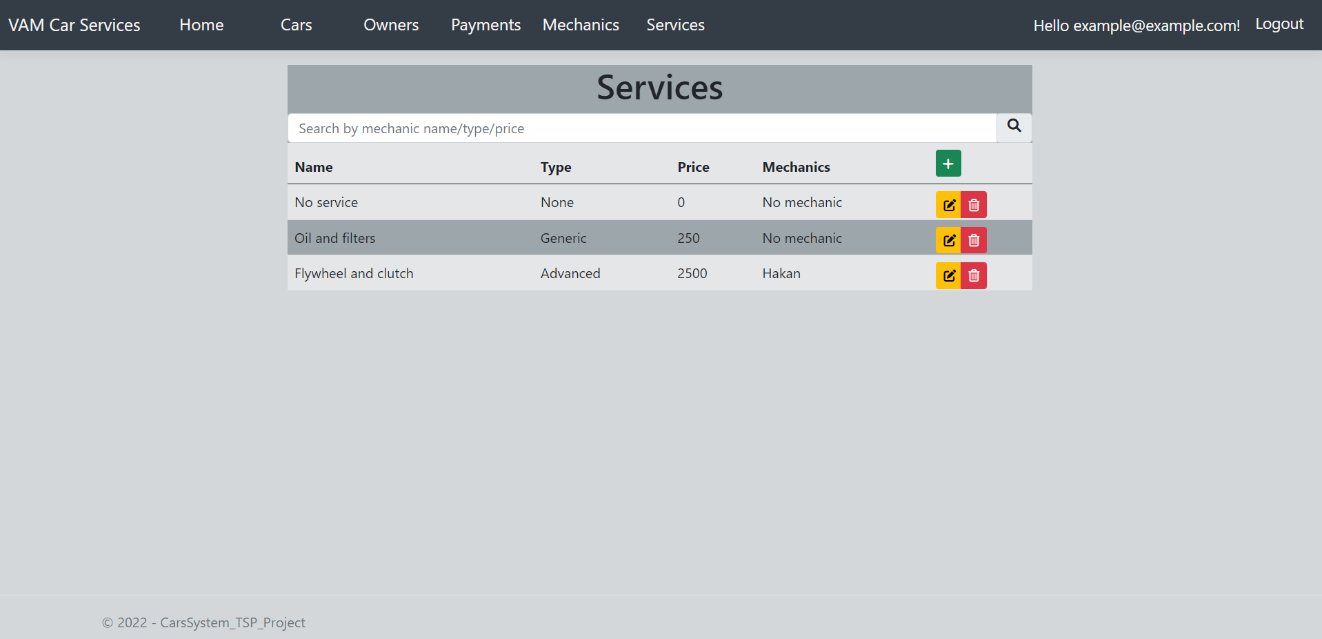
За редактиране на запис е необходимо да се натисне жълтият бутон с форма на молив. Зарежда се страница с информацията на конкретния запис и след направата на необходимите корекции, се натиска бутона Submit за запазване, или бутона View All за връщане назад.

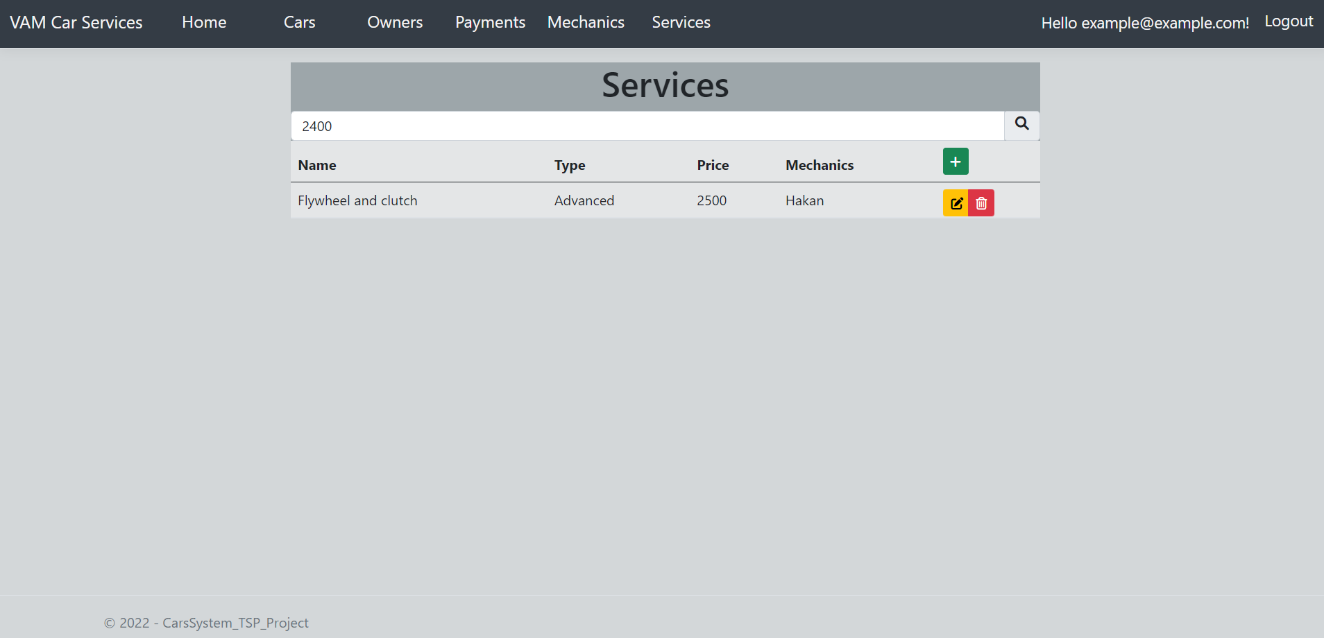


За изтриване на даден запис е необходимо да се натисне върху червения бутон с форма на кошче. След това се зарежда информацията за записа, както и два бутона – за изтриване (Delete) и връщане назад View all.

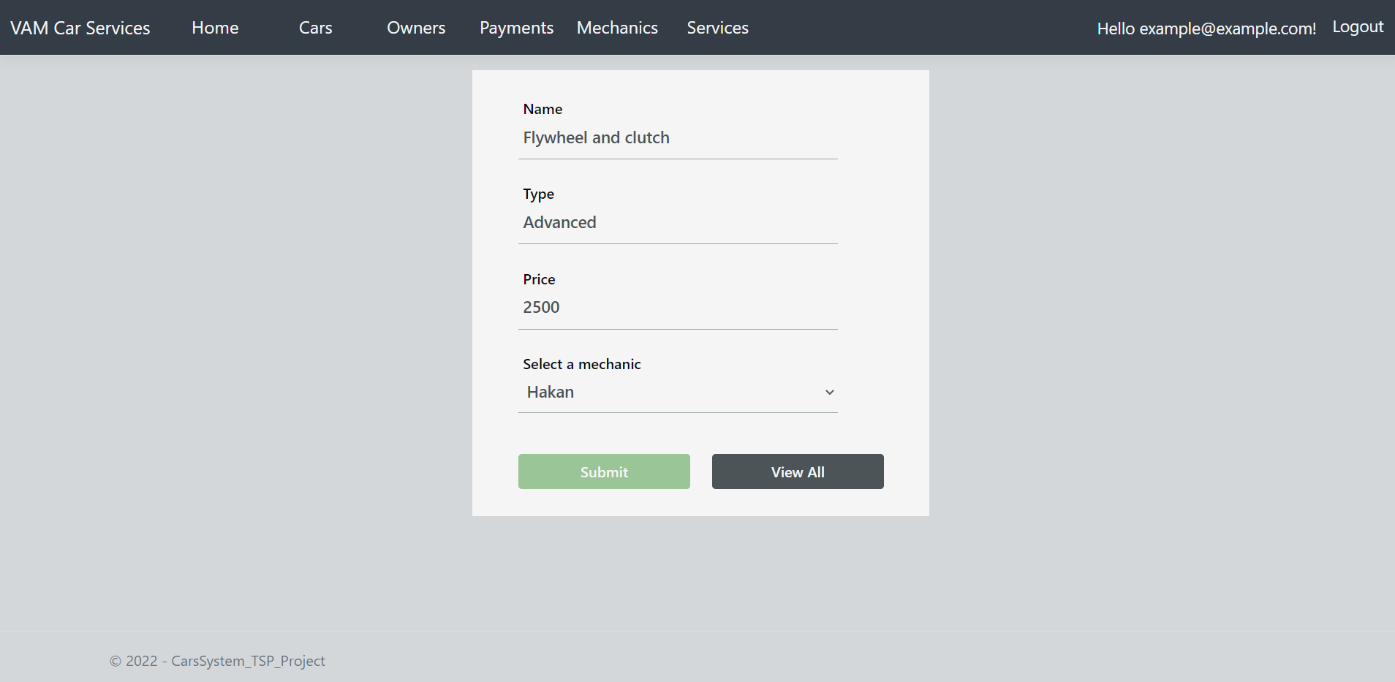


При натискане на секцията Services, се зарежда списък в табличен вид със всички налични записи за видовете обслужване и техните цени в базата данни. Над тях е поместено поле за търсене, чрез което може да се филтрират показаните записи по съответните критерии: цена/вид/механик, извършил обслужването. След въвеждане и натискане на enter бутон от клавиатурата или лупата до полето за търсене, резултатите се изобразяват. При повторно натискане (когато полето е празно), на екрана отново се извеждат всички налични записи в базата данни.

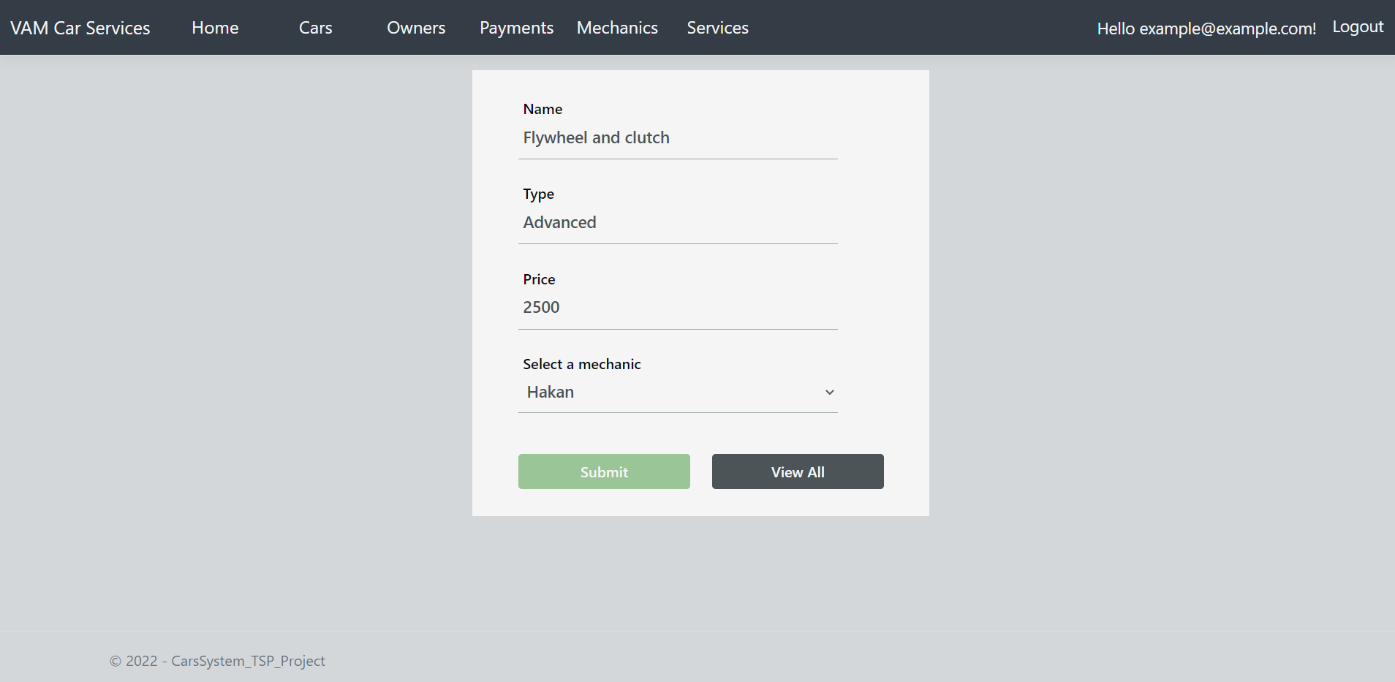




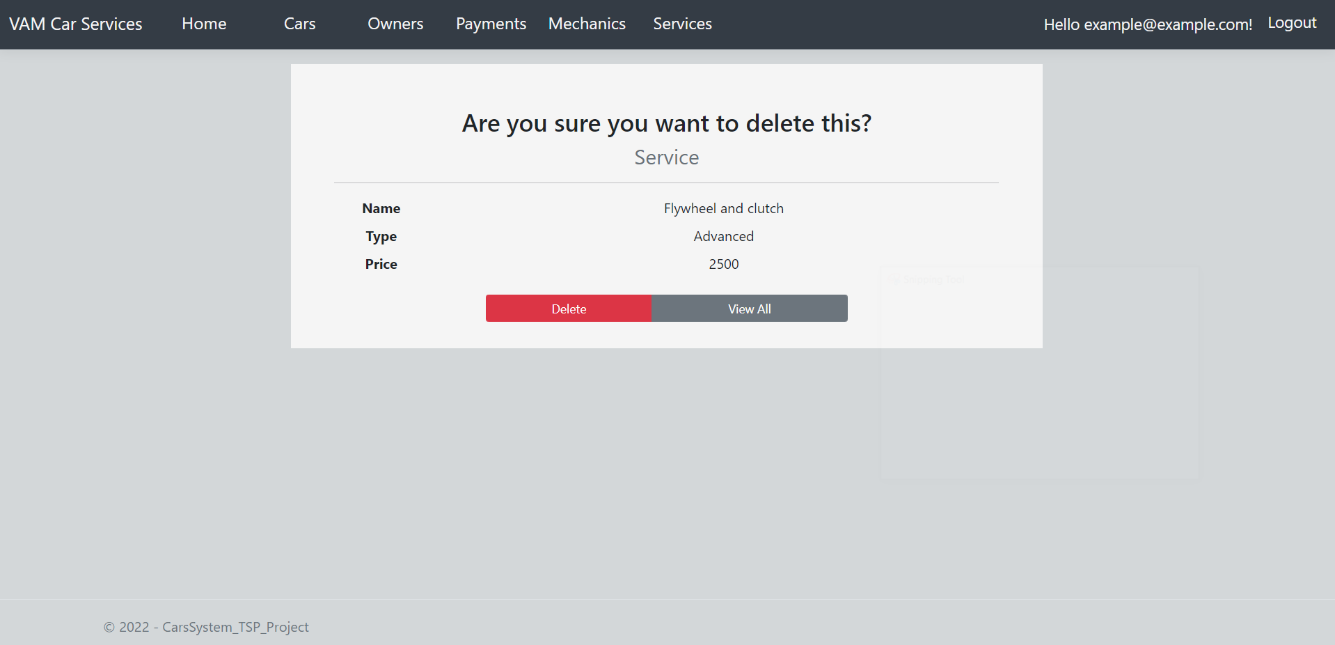
За добавяне на нов запис е необходимо да се натисне зеления бутон „+“. След това се зарежда страница, в която трябва да се попълнят данните на новото обслужване. По време на въвеждането, при некоректност на формата на попълнените данни, се изобразяват подсказващи съобщения. След успешно въвеждане на данните спрямо критериите, се натиска зеления бутон Submit, с което записът се запазва. За връщане назад по време на въвеждането се натиска бутона View All.



За редактиране на запис е необходимо да се натисне жълтият бутон с форма на молив. Зарежда се страница с информацията на конкретния запис и след направата на необходимите корекции, се натиска бутона Submit за запазване, или бутона View All за връщане назад.



За изтриване на даден запис е необходимо да се натисне върху червения бутон с форма на кошче. След това се зарежда информацията за записа, както и два бутона – за изтриване (Delete) и връщане назад View all.



1. **Изводи и възможности за развитие.**

**Могат да се реализират различни типове потребители, всеки с определена роля. Има възможност да се конфигурира външна аутентикация. Също така системата може сравнително лесно да се преправи, за да поддържа история на обслужванията на дадено превозно средство, както и да се реализират допълнителни критерии за търсенията. Възможно е да се създаде секция за изчисляване на лизингови вноски, лихви, модел на плащане и т.н.**