Документация на групов проект

Rent a car

Repository - https://github.com/Sveli0/Best_Rent-A-Car

1. Цели:

1.1 Потребители:

В системата могат да бъдат регистрирани много потребители. Достъп до системата получават единствено потребители, които са предоставили потребителско име и парола и са преминали успешно през процес по автентикация. На потребители, които не са преминали автентикационен процес се отказва достъп до функционалността на системата.

Съществуват два вида профили: потребителски и администраторски. Всеки с потребитлески профил може да разглежда предлаганите коли под наем и да прави заявки, докато администраторите профили могат да създават, преглеждат, променят и изтриват (CRUD) потребители, коли от автопарка на компанията, както и да преглеждат и одобряват резервации на коли.

За всеки потребител се пази следната информация:

- потребителско име
- парола
- собствено име
- фамилно име
- ЕГН
- телефонен номер
- адрес на електронна поща

1.2 Автомобили

В системата се предлагат различни автомобили под наем, като по заявките може да се направи справка кога дадена кола ще бъде свободна. Автомобилите в системата се управляват от администраторите, като за всеки автомобил се пази и показва следната информация:

- марка на автомобила
- модел на автомобила
- година на автомобила
- брой пасажерски места
- кратко описание (съдържащо техническа информация), по избор
- цена за наем на автомобила/ден

1.3 Заявки/Резервации

След вход в системата (автентикация) всеки потребител може да посети страницата, на която се създават заявки и да посочи датите, за които иска да наеме автомобил. На база въведените дати потребителя вижда списък с автомобилите, свободни за тези дати и може да направи заявка за наемането на избрания от него автомобил.

Потребители с администраторска роля могат да посетят страницата за управление на заявки, където могат да прегледат всички постъпили заявки за резервация на автомобили. Всяка заявка се състои от:

- избран автомобил
- начална дата
- крайна дата
- потребител, наемащ автомобила

2. Разпределение на работата

Разпределението на работата беше динамично в процес на разработка, гледахме всеки от партньорите успешно да добие опит във всяка част от изграждането му. Често се разделяхме да работим на различни модели за да може комбинирането на функционалностите да е по – лесно.

3. Реализация

При създаването на проекта генерирахме автоматично проект със имплементирана логика и презентация за потребители. Използвайки клас който наследява AspNetCore IdentityUser класа, успяхме да добавим нужните полета към нашите потребители. След написване на контролер и създаване на View за всяка част от логиката бяхме завършени с управлението на потребителите.

След това чрез добавяне на няколко реда в StartUp.Cs имплементирахме роли, които да ограничават достъпа до функционалностите.

```
        // This method gets called by the runtime. Use this method to add
services to the container.
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddDbContext<ApplicationDbContext>(options =>
                options.UseSqlServer(
                    Configuration.GetConnectionString("DefaultConnection")));
            services.AddDatabaseDeveloperPageExceptionFilter();

            services.AddDefaultIdentity<User>(options =>
options.SignIn.RequireConfirmedAccount = false)
                .AddRoles<IdentityRole>()
                .AddEntityFrameworkStores<ApplicationDbContext>();
            services.AddControllersWithViews();
        }

        // This method gets called by the runtime. Use this method to configure
the HTTP request pipeline.
        public async void Configure(IApplicationBuilder app, IWebHostEnvironment
env)
        {
```

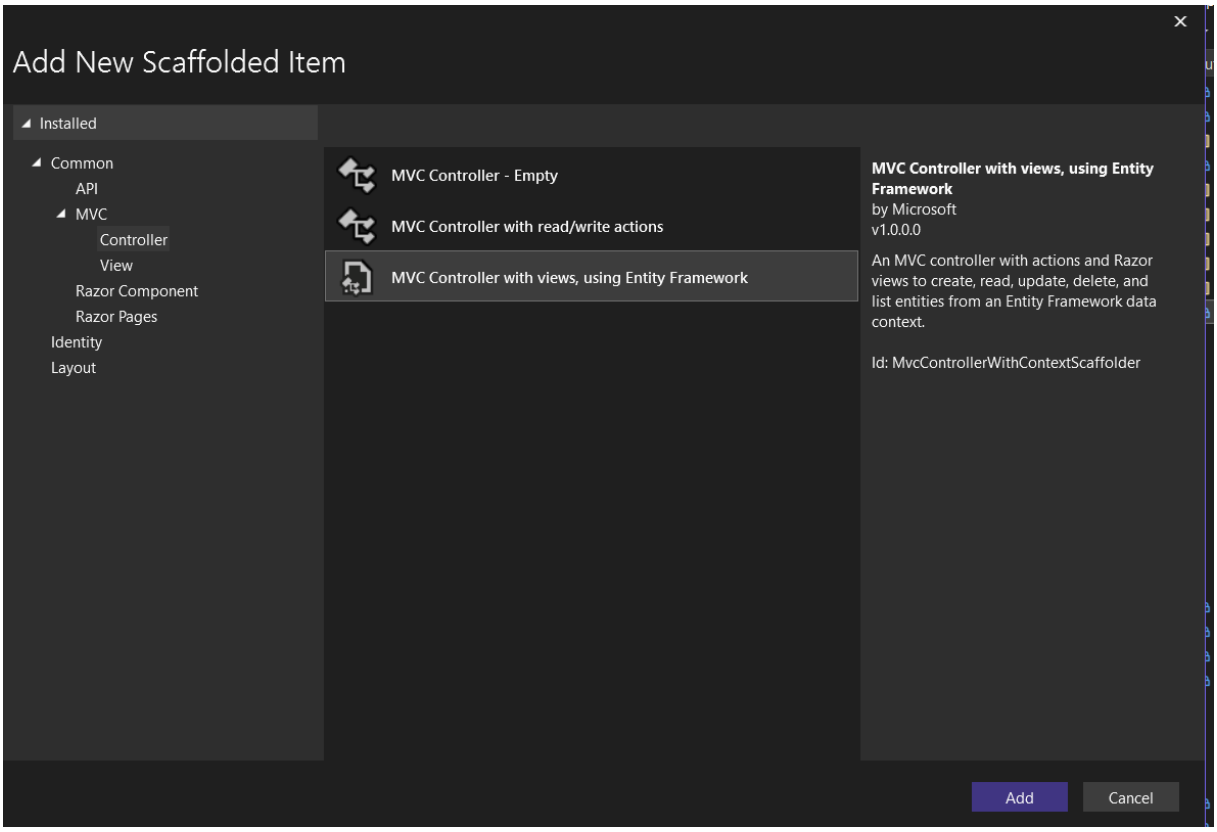
```
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
                app.UseMigrationsEndPoint();
            }
            else
            {
                app.UseExceptionHandler("/Home/Error");
                // The default HSTS value is 30 days. You may want to change this
for production scenarios, see https://aka.ms/aspnetcore-hsts.
                app.UseHsts();
            }
            app.UseHttpsRedirection();
            app.UseStaticFiles();

            app.UseRouting();

            app.UseAuthentication();
            app.UseAuthorization();

            app.UseEndpoints(endpoints =>
            {
                endpoints.MapControllerRoute(
                    name: "default",
                    pattern: "{controller=Home}/{action=Index}/{id?}");
                endpoints.MapRazorPages();
            });

            using (var scope = app.ApplicationServices.CreateScope())
            {
                var userManager =
scope.ServiceProvider.GetRequiredService<UserManager<User>>();
                var roleManager =
scope.ServiceProvider.GetRequiredService<RoleManager<IdentityRole>>();
                string[] roles = new string[] { "Admin", "Customer" };

                foreach (var role in roles)
                {
                    if (!await roleManager.RoleExistsAsync(role))
                    {
                        await roleManager.CreateAsync(new IdentityRole(role));
                    }

                }

                string adminEmail = "admin@rentacar.bg";
                string adminPassword = "Admin1234#";

                if (await userManager.FindByNameAsync(adminEmail)==null)
                {
                    User admin = new User();
                    admin.UserName = adminEmail;
                    admin.FirstName = "Karamfil";
                    admin.LastName = "Karamfilov";
                    admin.EGN = "0000000000";
                    admin.Email = adminEmail;
                    admin.EmailConfirmed = true;
                    admin.PhoneNumber="088maikatanarosen";
                    await userManager.CreateAsync(admin, adminPassword);
                    await userManager.AddToRoleAsync(admin, "Admin");
                }
            }
        }
```

Добавения код осигурява създаването на ролите за потребителите и автоматично се генерира админ със следните полета за достъп:

Username = admin@rentacar.bg

Password = Admin1234#

След това използвахме генерация на MVC Controller with views, using Enitity Framework, за да генерираме CRUD за колите



След което ограничихме достъпа до страниците, до които не трябва потребителите и не автентицизиралите се да имат достъп.

След това създадохме UserController-а, който има за цел да индексира променя и изтрива потребители за да могат да бъдат управлявани от администраторите.

След това създадохме CarReservationControllera, където стои основната функционалност на нашия проект, чрез метод за Търсене и проверки на потребителя му се извежда списък със свободните коли за определен период и му се дава възможност да направи заявка за някоя от колите за постигането на тази цел създадохме 4 метода и 2 изгледа:

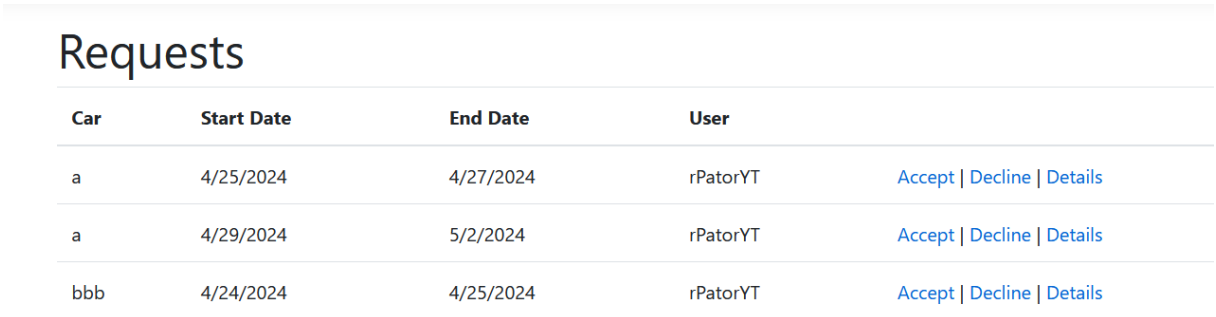
За всеки изглед има HttpGet и HttpPost метод

Search изгледа е с цел да прочете датите, за които е заинтересован потребителя, след което да ги предаде на Reserve изгледа, който спрямо тези дати да набере списък със свободните по това време коли.

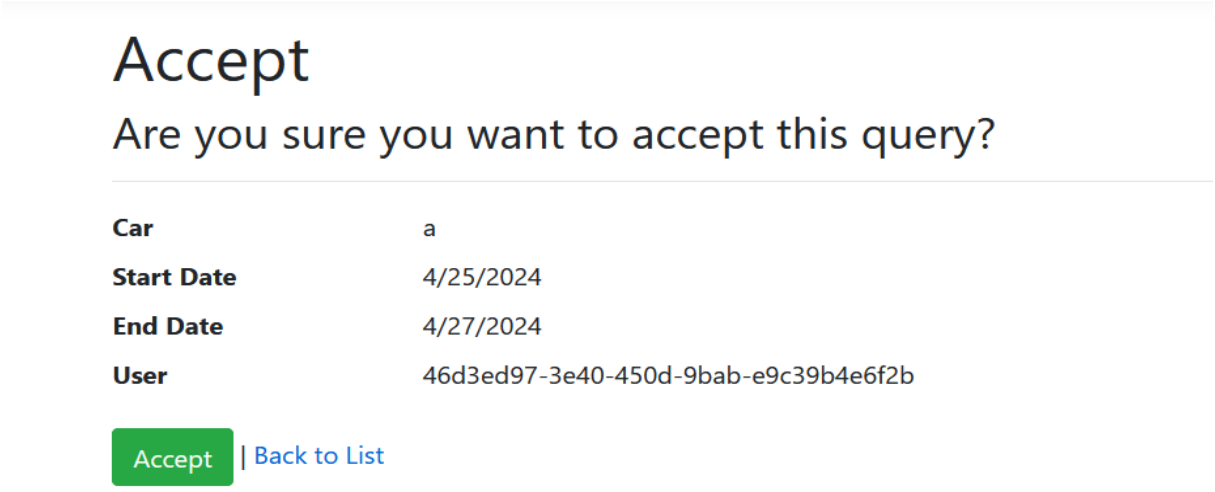
След което чрез кликване на бутон потребителя създава заявки.

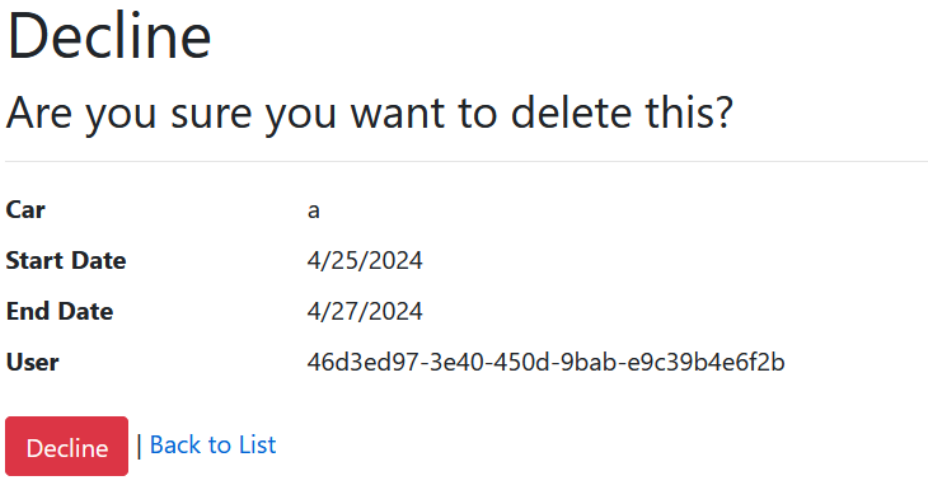
При управлението на заявки администраторите имат възможноста да разглеждат резервациите(одобрените заявки) и неодобрените заявки, в индекса за неодобрените заявки те имат възможността да приемат и отказват заявки.

Индексиране на неодобрените заявки:



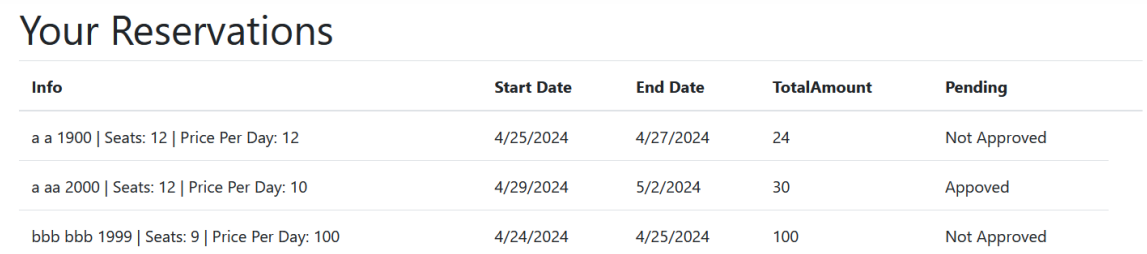
Изглед при приемане (бутона е зелен 😁)





Изглед при отказване

Заявките имат и направена основна CRUD функционалност, чрез която се достигат тези функционалности.



Your Reservations

Info	Start Date	End Date	TotalAmount	Pending
a a 1900 | Seats: 12 | Price Per Day: 12	4/25/2024	4/27/2024	24	Not Approved
a aa 2000 | Seats: 12 | Price Per Day: 10	4/29/2024	5/2/2024	30	Appoved
bbb bbb 1999 | Seats: 9 | Price Per Day: 100	4/24/2024	4/25/2024	100	Not Approved

5. Развитие и нововъведения

В проекта има малки спънки, които в бъдеще бихме желали да подобрим:

При валидиране на датите при търсене валидацията е успешно но поради препращането не се изписва съобщението за валидация.

Моделите ни се намират в главния проект, а не в отделена библиотека поради проблеми със свързването на проектите и начина, по който взаимодействат връзките в GitHub.

6.Заключение

Извършвайки осново поставената задачи ние добихме много нови знания и опит в разработката на software в Asp.Net

7.Изпозлзвани технологии и литература

- модела за езикова обработка базиран на изкуствен интелект, ChatGPT на OpenAI

- StackOverflow

- лекциите на Shad Sluiter, качени в Youtube

- https://youtu.be/Y6DCP-yH-9Q?si=IKokSBDsBdh8hjVd – за авторизация