

Добрый день!) Моё имя Валентина и здесь я оставляю комментарии по выполненному ТЗ)  
Благодарю за прочтение комментариев к выполненному проекту!)

Также прикрепил сжатую

- инструкцию для запуска проекта Instruction\_IlCats и
- схему доменной модели,
- загрузила проект на свой GitHub аккаунт (<https://github.com/valentina25tim>) ,
- ссылка на сам проект (<https://github.com/valentina25tim/ASP.Net> checkout branch -> ParsingSite )
- архивированный проект.

То, что я сделала, заняло 4,5 дня. С не выполненного: не все данные являются истинно взятыми с сайта(далее есть описание какие были выпарарсенны) и не настроена БД для большого объема информации  
Даже если моих знаний не достаточно, благодарю за это задание. В ходе его выполнения я почерпнула новые знания и навыки, а это уже не мало важно) Ранее не занималась парсингом и единственное, для чего использовала виндовсФорм – это калькулятор)

Спасибо Вам)

## Комментрии

1	Для парсинга был выбран сайт <a href="https://www.ilcats.ru/toyota/?function=getModels&amp;market=EU">https://www.ilcats.ru/toyota/?function=getModels&amp;market=EU</a>	
2	<div>Была проанализирована модель для проектировки, значения каждого из полей. Доменная модель в проекте выглядит следующим образом</div> <div><pre>classDiagram     class CarModel {         Name         Id     }     class List_ModelCarType {         Code         DateFrom         DateTo         Completion         Id         ModelCarId     }     class List_CompletionVariable {         CompletionV         DataFrom         DataTo         Engine         Grade         AtmMtm         GirShiftType         DriverPosition         FuelInduction         Id         ModelCarTypeId     }     class List_SparePart {         Name         Id         CompletionVariableId     }     class List_SpareTypeDetail {         Code         ModifiedCode         Quantity         DateFrom         DateTo         Applying         Id         SparePartId     }     CarModel --&gt; List_ModelCarType     List_ModelCarType --&gt; List_CompletionVariable     List_CompletionVariable --&gt; List_SparePart     List_SparePart --&gt; List_SpareTypeDetail</pre></div>	<div>Объяснение взаимосвязи:</div> <div>У CarModel есть Id, по которому выполняется привязка списка ModelCarType к ModelCarId.</div> <div>Таким образом при выборке в БД CarModel будет работать скрипт:</div> <div><pre>SELECT top 100 * from [dbo].[Cars] as car join ModelCarType as cartype on cartype.ModelCarId = car.Id join CompletionVariable as completion on completion.Id = cartype.ModelCarId</pre></div> <div>Аналогично выглядит связь между стальными частями доменной модели.</div>
3	Был создан Solution с папкой src, в которую добавлены папки (API, Core, Infrastructure, WForm). В папку Infrastructure был добавлен Console Application <b>IlCats_Parsing</b> (после будет изменён на Class Library)	
4	В IlCats_Parsing добавлена библиотека для дальнейшего парсинга <b>HtmlAgilityPack</b> , папки HalperLibrary, Parsing	
5	<div>Была проанализирована страничка сайта <a href="https://www.ilcats.ru/toyota/?function=getModels&amp;market=EU">https://www.ilcats.ru/toyota/?function=getModels&amp;market=EU</a></div> <div>И чтоб достать списки</div> <div><pre>List&lt;string&gt; DivNames_n List&lt;List&lt;string&gt;&gt; DivCode_n List&lt;List&lt;string&gt;&gt; DivComplectation List&lt;List&lt;DateTime&gt;&gt; DivDateFrom_n</pre></div>	

Лis<List<DateTime>> DivDateTo\_n

выбраны такие селекторы

html/body/div/div/div[@class='List'<←(list with contents for one name)

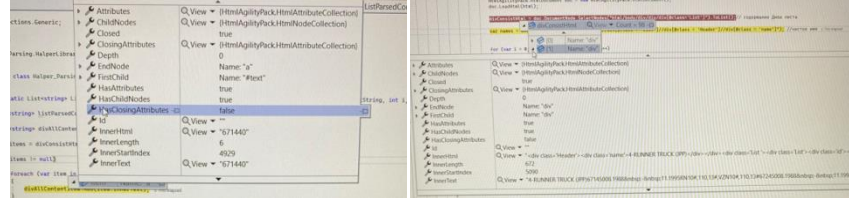
./div[@class = 'List']//div[@class = 'Header']//div[@class = 'name']<←(name list)

//div[@class='id']/a<←(list with complectations for all names)

//div[@class='modelCode']<←(list with codes for all names)

//div[@class='dateRange']<←(list with dates for all names)

## 6 Полученные данные имеют следующий вид



Если перевести всё в текстовую форму, то дальнейшая работа будет происходить с такими списками взятых по xPath данных:

Список1 → с 98 (имя + коды + даты + комплектация)

Список 2 → с 208 (коды)

Список3 → с 208 (даты)

Список4 → с 208 (комплектация)

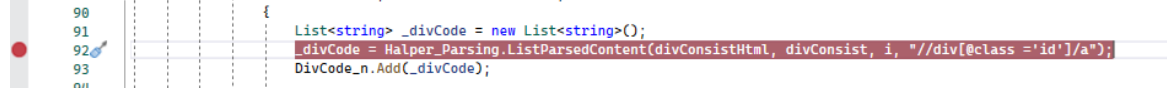
Список5 → с 98 (имена)

В папку HalperLibrary были добавлены файлы с методами расширения для сравнения содержимого

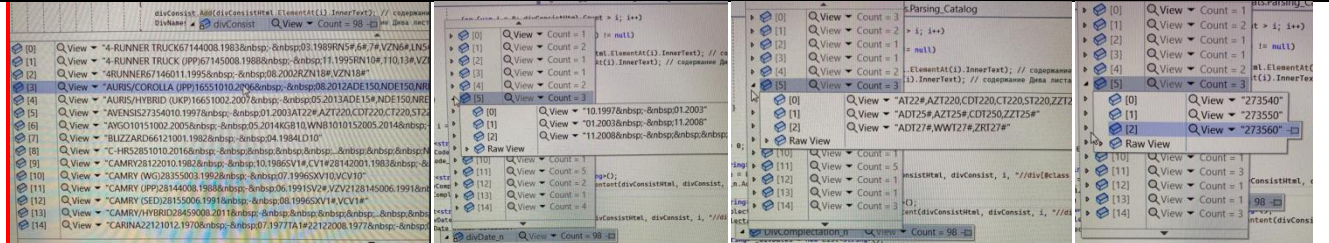
Списка1 с Списком2/Списком3/Списком4.

Если кратко, то строки приводятся к чарам и сравниваются поочередно, оглядываясь на элемент с следующим индексом. Для приведения строки в дату использовано буллевое int32(Parse), определена по кратности дата начала и конца(в случае даты «08.2010 - ...» для второго значения ставится Now).

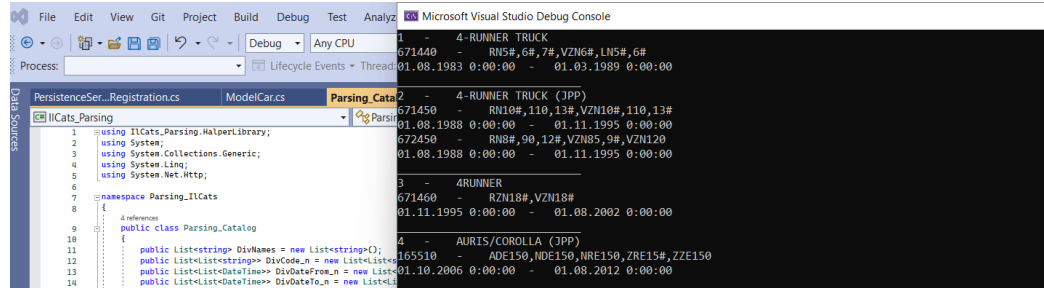
Если интересно как кочают данные, то можно в проекте **ILCats\_Parsing** в файле **Parsing\_Catalog.cs** строка 98 поставить брейкпоинт и посмотреть логику вычитки.



В результате получаются заполненные данными списки с пункта №5.



## 7 Написала метод Print для вывода на консоль



## 8 В проекте **ILCats\_Parsing** изменила Property сменить на ClassLibrary для дальнейшей работы

Output type


Specifies the type of application to build.

Class Library

Class Library

Console Application

Windows Application

9	<p>В папку Core добавила проект ClassLibrary <b>IlCats_Damain</b> с папками Entity, Repositories.</p> <p>В Entity добавлена папка Car с классами, соответствующими схеме с пункта №1. Каждый класс наследует интерфейс с обязательным полем Id.</p> <p>В Repositories добавлен интерфейс, определяющий действия работы с базой данных.</p>
10	<p>В папку Core добавлен проект ClassLibrary <b>IlCats_Application</b> с установленными пакетами автомаппера, валидатора и медиатора для инверсии зависимостей. Добавлена ссылка на проект IlCats_Domain.</p> <p>Созданы папки</p> <p>Contracts (содержит интерфейс для репозитория модели, который определяет методы для задач удалить/создать/получить... )</p> <p>Features(содержит папку с DTO для выбранной модели, команды и запросы)</p> <p>Profiles(маппинг доменной модели на обрезанную модель(DTO) )</p>
11	<p>В папку Interface добавлен проект ClassLibrary <b>IlCats_Persistence</b> с пакетами для работы с EntityFramework и конфигурирования. Добавлены ссылки на проекты <b>IlCats_Application, IlCats_Domain, IlCats_Parsing.</b></p> <p>Добавлены папки :</p> <p>Configuration (конфигурирует доменную модель ModelCar по Name)</p> <p>Repositories(содержит ModelCarRepository.cs, который наследует BaseRepository.cs – по DbContext выполняются методы для работы с БД – получить весь список, получить по айди, удалить, сохранить ...)</p> <p>Добавлены файлы :</p> <p>IlCatsDbContext.cs(создается инстанс с проекта IlCats_Parsing и вызывается метод Start, в котором заполняются публичные листы с списками пункта №5,</p> <p>Создается DbSet&lt;ModelCar&gt; и через modelBuilder по циклам for заполняются поля для доменной модели) . Используется Fluent.</p> <p>PersistenceServiceRegistration.cs( регистрируются строка подключения (далее будет уточнение) , По AddScoped регистрируются и конфигурируются репозитории и доступ к базе данных)</p>
12	<p>В папке API создан API проект <b>IlCats_API</b> с пакетами для логгирования и свэггера.</p> <p>Добавлены ссылки на проекты <b>IlCats_Application, IlCats_Persistence.</b></p> <p>Добавлен файл Nlog.config с настройкой журнала и локацией файлов в папке Logs.</p> <p>В appsetting.json добавлена строка подключения к базе данных</p> <pre> "ConnectionStrings": {   "IlCatsDbConnectionString": "Server = LAPTOP-MFD4FGN1\\SQLEXPRESS07; Initial Catalog = IlCats_db; Integrated Security = True" }, </pre> <p>Вам для запуска проекта нужно поставить свою.</p> <p>Создана папка Filters с файлом CustomExceptionHandlerAttribute.cs , в котором в зависимости от Http ответа обрабатываются исключения.</p> <p>В классе Program добавлено конфигурирование журнала.</p> <p>В классе Startup добавлены сервисы для репозитория, свэггера, подключения к БД, контроллерам с обработкой исключений.</p> <p>Знаю, что это не было необходимо, но раз уж есть БД, то я решила добавить команды/запросы))</p> <p>В папке Controllers создан контроллер</p> <ul style="list-style-type: none"> <li>- CarController с командами(обновить по айди, удалить, создать) и запросами(получить весь список, получить по айди),</li> <li>- CarController_By_Base_Crud (было интересно сделать для сокращения кода при большем числе моделей), который наследует BaseCrudController и содержит один GET запрос.</li> </ul>
13	<p>Далее были выполнены следующие шаги:</p> <div>  </div> <p>13.1 – Выбрать проект для старта <b>IlCats_API</b></p>

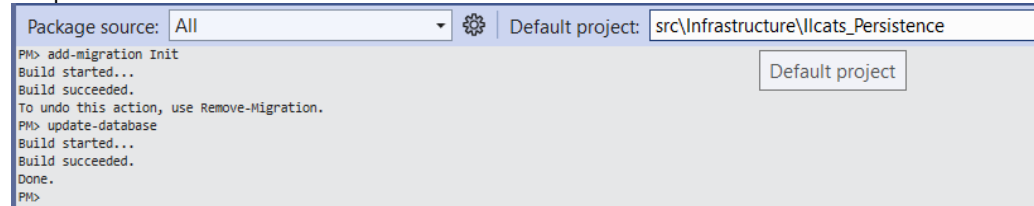
13.2 – В проекте IlCats\_API в файле appsettings.json указать свою строку подключения к базе данных

```
"ConnectionStrings": {  
  "IlCatsDbConnectionString": "Server = LAPTOP-MFD4FGN1\\SQLEXPRESS07; Initial Catalog = IlCats_db; Integrated Security = True"
```

13.3 – В Package Manager Console выбрать проект IlCats\_Persistence и выполнить последовательно команды:

-> Add-Migration Initial (может занять какое-то время)

-> Update-Database



13.4 – Проверить наличие созданной базы данных и заполненные в ней поля.

\*Можно воспользоваться скриптом для запроса выборки 20 полных моделей :

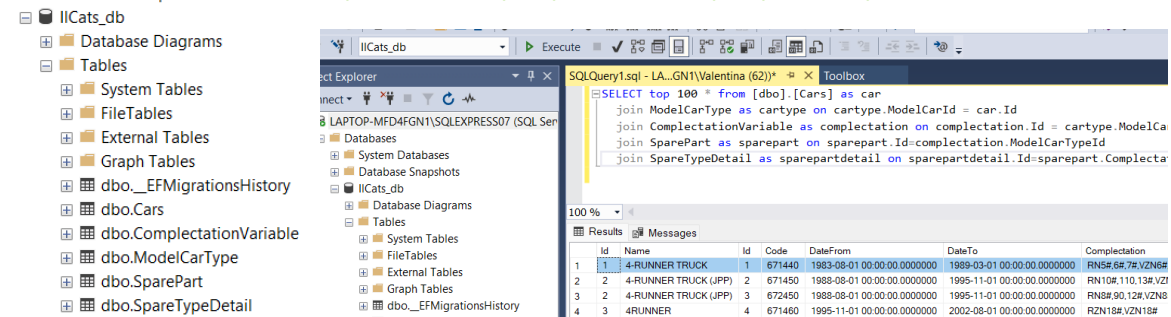
SELECT top 20 \* from [dbo].[Cars] as car

join ModelCarType as cartype on cartype.ModelCarId = car.Id

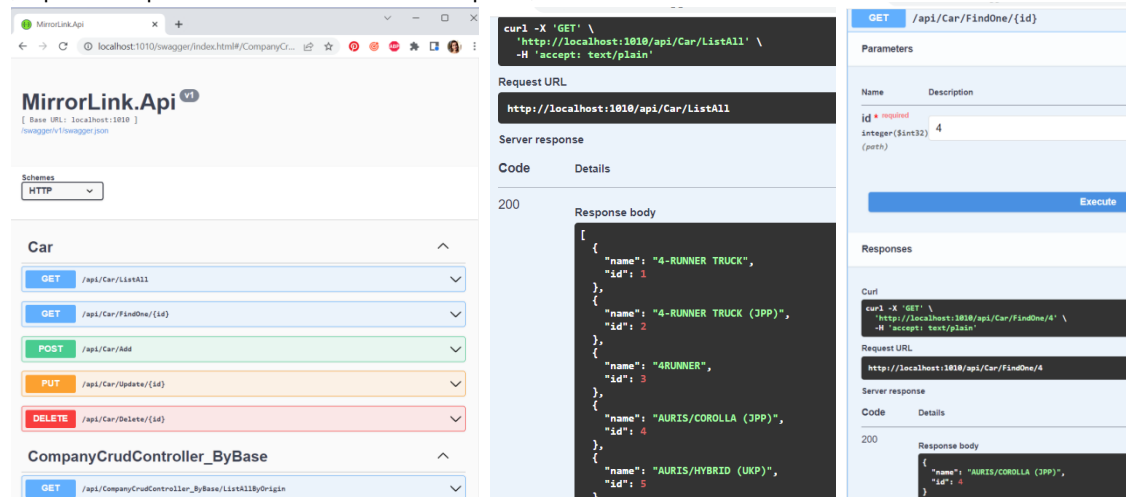
join ComplectationVariable as complectation on complectation.Id = cartype.ModelCarId

join SparePart as sparepart on sparepart.Id=complectation.ModelCarTypeId

join SpareTypeDetail as sparepartdetail on sparepartdetail.Id=sparepart.ComplectationVariableId



13.5 – Запустить IlCats\_API (<http://localhost:1010/swagger/index.html>) и протестировать контроллеры (первый запрос может занять какое-то время)



Комментарии по пункту №13 :

Создаётся миграция данных в базу данных и заполняются поля модели ИСТИННЫМИ значениями (CarModel.dbo(Name), ModelCarType( Code, Complectation, DateFrom, DateTo)) и формируется moack data для всех остальных полей так как я не распарсила все остальные страницы.

Учитывая созданную связь между данными, для того чтоб селектить 20 ModelCar будет работать такой скрипт для запроса в sql

SELECT top 20 \* from [dbo].[Cars] as car

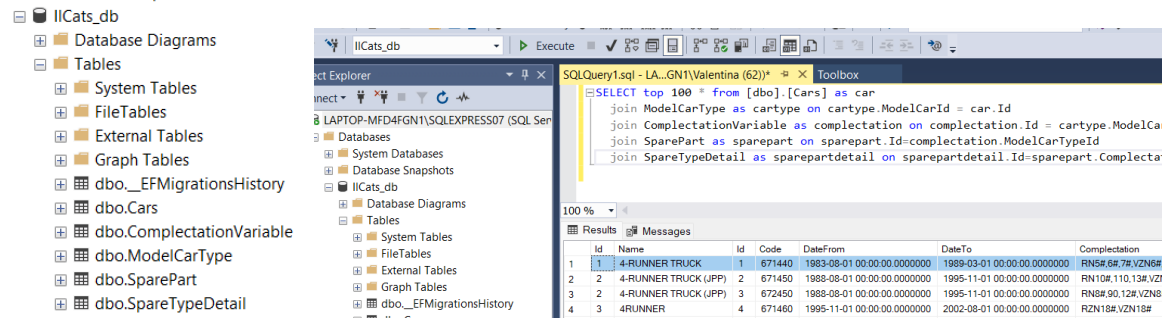
join ModelCarType as cartype on cartype.ModelCarId = car.Id

join ComplectationVariable as complectation on complectation.Id = cartype.ModelCarId

```

join SparePart as sparepart on sparepart.Id=complectation.ModelCarTypeId
join SpareTypeDetail as sparepartdetail on sparepartdetail.Id=sparepart.ComplectationVariableId

```



The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Database Explorer' pane displays the structure of the 'ILcats\_db' database, including tables like 'dbo.Cars', 'dbo.ComplectationVariable', 'dbo.ModelCarType', 'dbo.SparePart', and 'dbo.SpareTypeDetail'. The main window shows a SQL query window with the following query:

```

SELECT top 100 * from [dbo].[Cars] as car
join ModelCarType as cartype on cartype.ModelCarId = car.Id
join ComplectationVariable as complectation on complectation.Id = cartype.ModelCa
join SparePart as sparepart on sparepart.Id=complectation.ModelCarTypeId
join SpareTypeDetail as sparepartdetail on sparepartdetail.Id=sparepart.Complecta

```

The 'Results' pane shows the following data:

	Id	Name	Id	Code	DateFrom	DateTo	Complectation
1	1	4-RUNNER TRUCK	1	671440	1983-08-01 00:00:00.0000000	1989-03-01 00:00:00.0000000	RN5# 6# 7# VZN6#
2	2	4-RUNNER TRUCK (JPP)	2	671450	1988-08-01 00:00:00.0000000	1995-11-01 00:00:00.0000000	RN10# 110.13# VZ1
3	2	4-RUNNER TRUCK (JPP)	3	672450	1988-08-01 00:00:00.0000000	1995-11-01 00:00:00.0000000	RN8# 90.12# VZN8
4	3	4RUNNER	4	671460	1995-11-01 00:00:00.0000000	2002-08-01 00:00:00.0000000	RZN18# VZN18#

При запуске ILcats\_API есть возможность выполнить простые crud операции ВКЛЮЧИЛА ДЛЯ ПРОСТОТЫ ТОЛЬКО NAME.

- 14 В папку WForm был добавлен проект WindowsForm **ILcats\_WiForm**  
Используя библиотеку System.Data.SqlClient, в файле Database.cs создано подключение к базе данных (Вам необходимо указать своё)

```

4 references
public class Database
{
    SqlConnection sqlConnection = new SqlConnection("Server = LAPTOP-MFD4FGN1\\SQLEXPRESS07; Initial Catalog = ILcats_db; Integrated Security = True");
}
2 references

```

И методы, открывающие и закрывающие доступ к БД.

Созданы две формы:

- Первая при запуске проекта выводит айди и имя, выполняет вывод при нажатии на поле имени или нужного айди в текстовые поля внизу формы, позволяет обновить данные в таблице и создать новую запись.
- Вторая выполняет запись в базу данных.