Разработка веб-интерфейса к базе данных

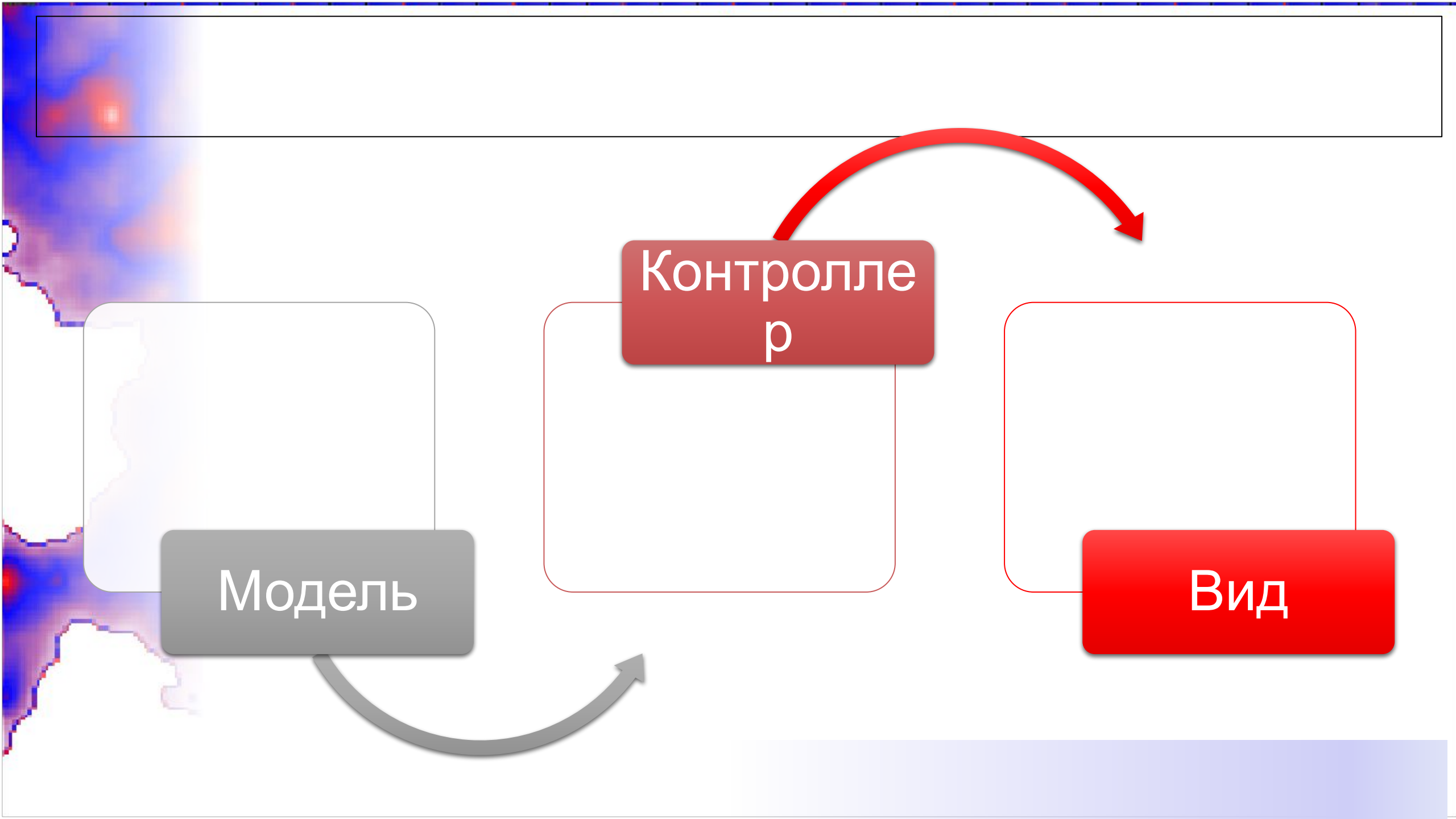
Предварительные условия

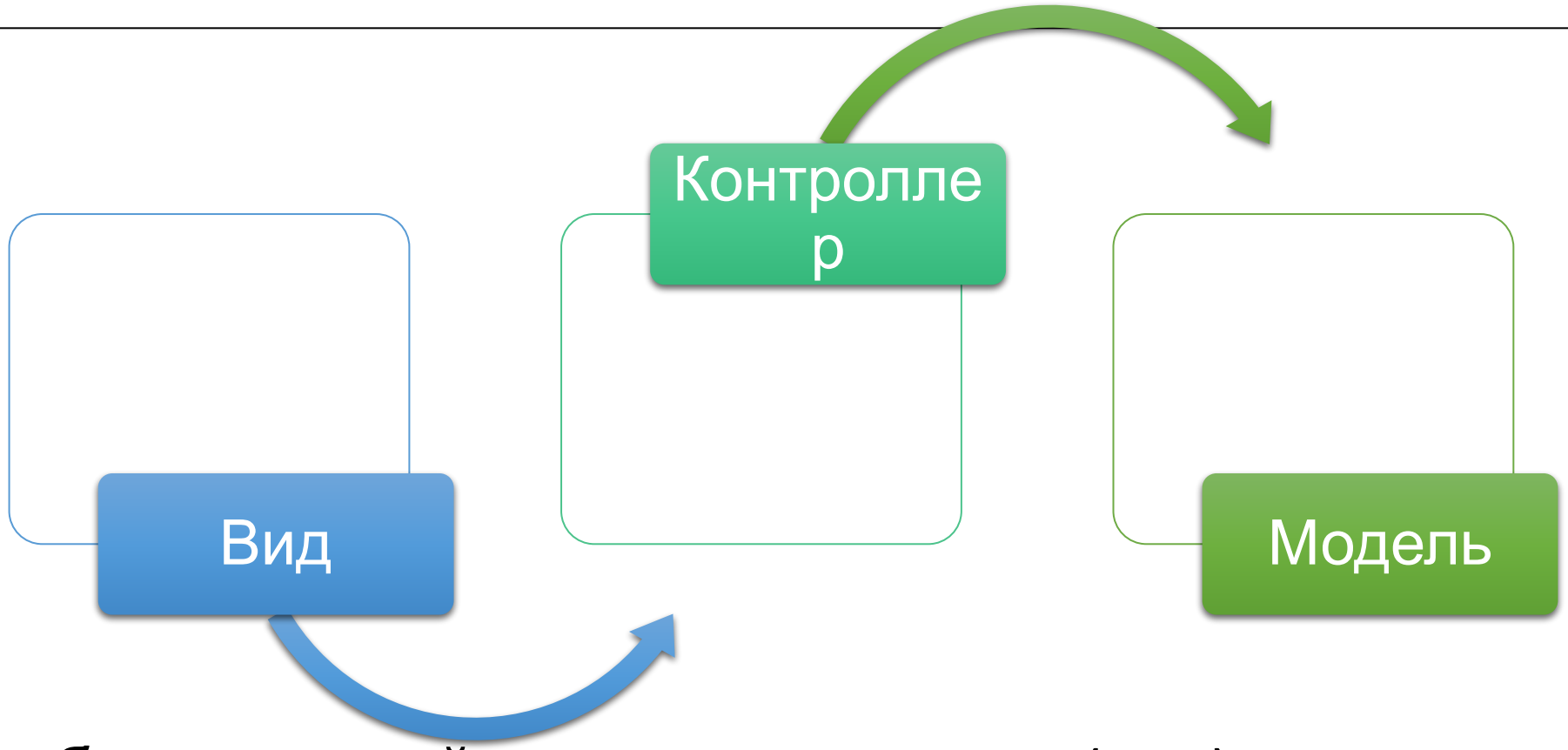
- Установлена среда разработки MS Visual Studio 2017
- Загружены необходимые компоненты среды:
  - Программирования .NET
  - ASP .NET
  - ASP Net Core
- Имеются общие сведения о работе в Visual Studio
- Желательно выполнить работу о создании БД и связи БД с элементами формы

Особенности разработки веб интерфейсов

Введение

- HTML страницы – удобный и универсальный инструмент для отображения на практически любом оборудовании и в любой операционной системе
- С другой стороны, для разработки серьезной информационной системы необходимо владеть не только приемами разработки HTML, но и технологиями поставки данных и интерактивного формирования веб-страниц
- Существует ряд решений для автоматизации рутинных задач в указанной области, основанных на модели MVC – модель, вид, контроллер
- Модель хранит описание данных и их взаимосвязи, вид – отображает данные, контроллер осуществляет логику работы с данными


Контроллер
Модель
Вид



- Верно и обратное – действия пользователя (вид) через логику контроллера влияют на данные модели

Отображение данных в WinForm

- Мы уже знаем, что данные удобно хранить в базе данных
- Для этого надо создать БД
- Подключиться к ней
- Считать как ее структуру, так и данные в соответствии со структурой
- Привязать (спроецировать) таблицы и поля базы данных к визуальным элементам
- Поддерживать установленную связь в направлении от БД к интерфейсу и пользователю и обратно – от действий пользователя к БД

Подход Microsoft

- Далее рассматривается реализация с помощью трех основных технологий фирмы, взаимосвязанных между собой
- Entity Framework – создание моделей
- NET Core – веб взаимодействие
- Razor Pages – шаблоны страниц HTML с реализованной логикой работы пользователя

- Руководство предназначено для студентов-медиков, изучающих дисциплину «Информационные медицинские системы»
- Как следствие, демонстрируется лишь базовая и элементарная функциональность, изложение ведется преимущественно основываясь на примерах.

Отображение простой таблицы БД

Постановка задачи

- Создадим интерактивную страницу, отражающую список студентов
- Список студентов будем хранить в таблице БД
- Список будет выводится в виде страницы HTML, которая обеспечит просмотр данных (списком или пофамильно), редактирование и удаление записей
- Здесь и далее не рассматриваются вопросы защиты информации, ограничения доступа к данным и т.п.
- Практически не затрагиваются действия по редактированию и доработке стандартных страниц

Сведения о студентах

N	Тип	Название	Описание
	int	ID	Первичный ключ
	string	Name1	Фамилия
	string	Name2	Имя
	string	Name3	Отчество
	string	BirthDate	Дата рождения
	string	Phone	Номер телефона
	string	Code	Номер зачетной книжки
	bool	Dorm	Нуждается в общежитии

- Сведения о студентах, которые мы храним в БД будут храниться в колонках таблицы, иначе говоря – в ее полях.
- Любая таблица включает колонку первичного ключа ID
- Мы будем почти всегда хранить данные в строках

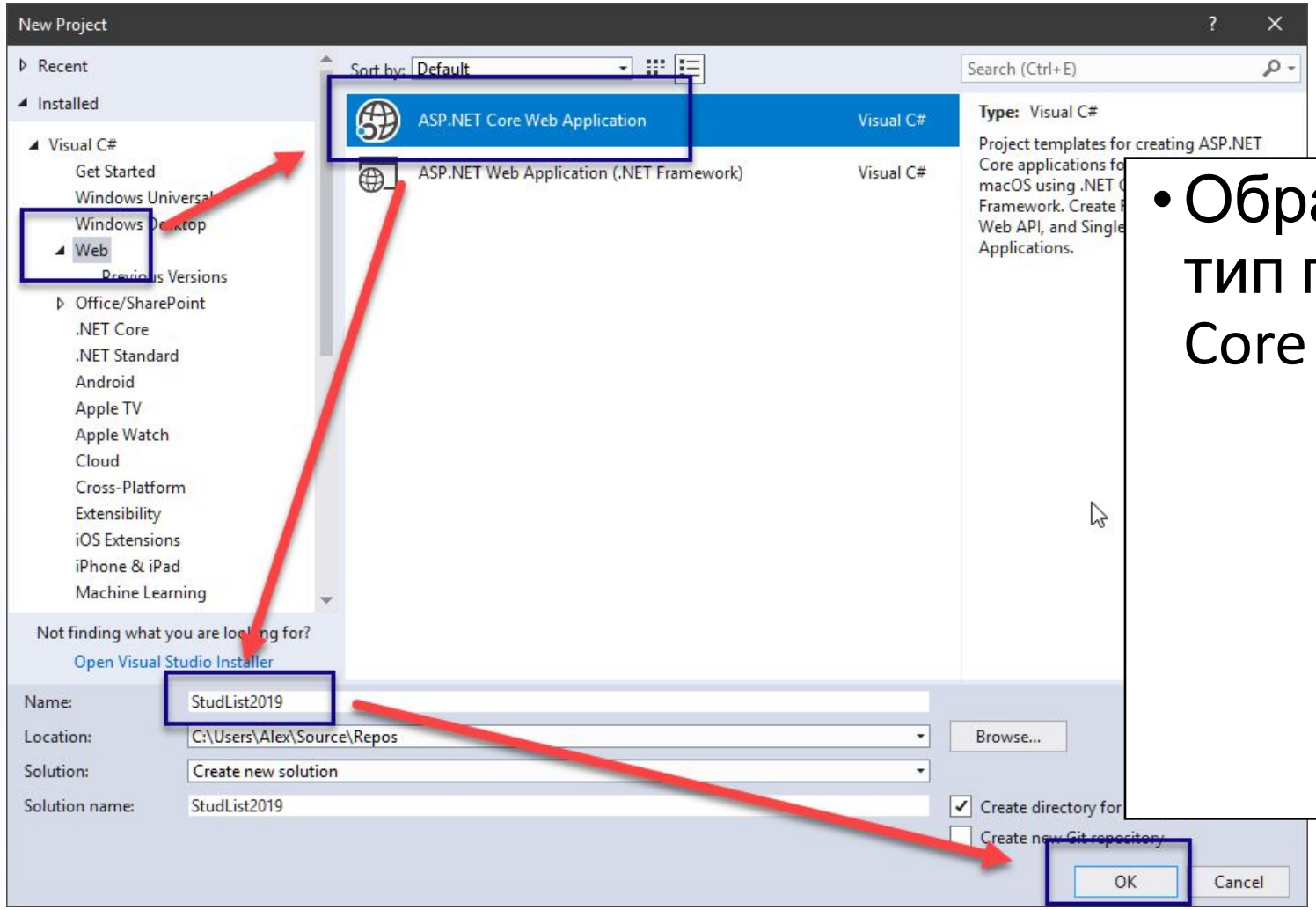
Два подхода

- Мы можем (Data First):
  - Создать БД
  - Подключиться
  - Сгенерировать модель по БД (сделать реверс-инжиниринг БД)
  - Отобразить ее в виде HTML страниц
- Альтернатива (Code First)
  - Создать классы модели
  - Отобразить классы на веб-страницы
  - Отобразить классы на БД (фактически – создать БД и таблицы, соответствующие модели)

Мы используем подход с созданием кода, так как он несколько проще

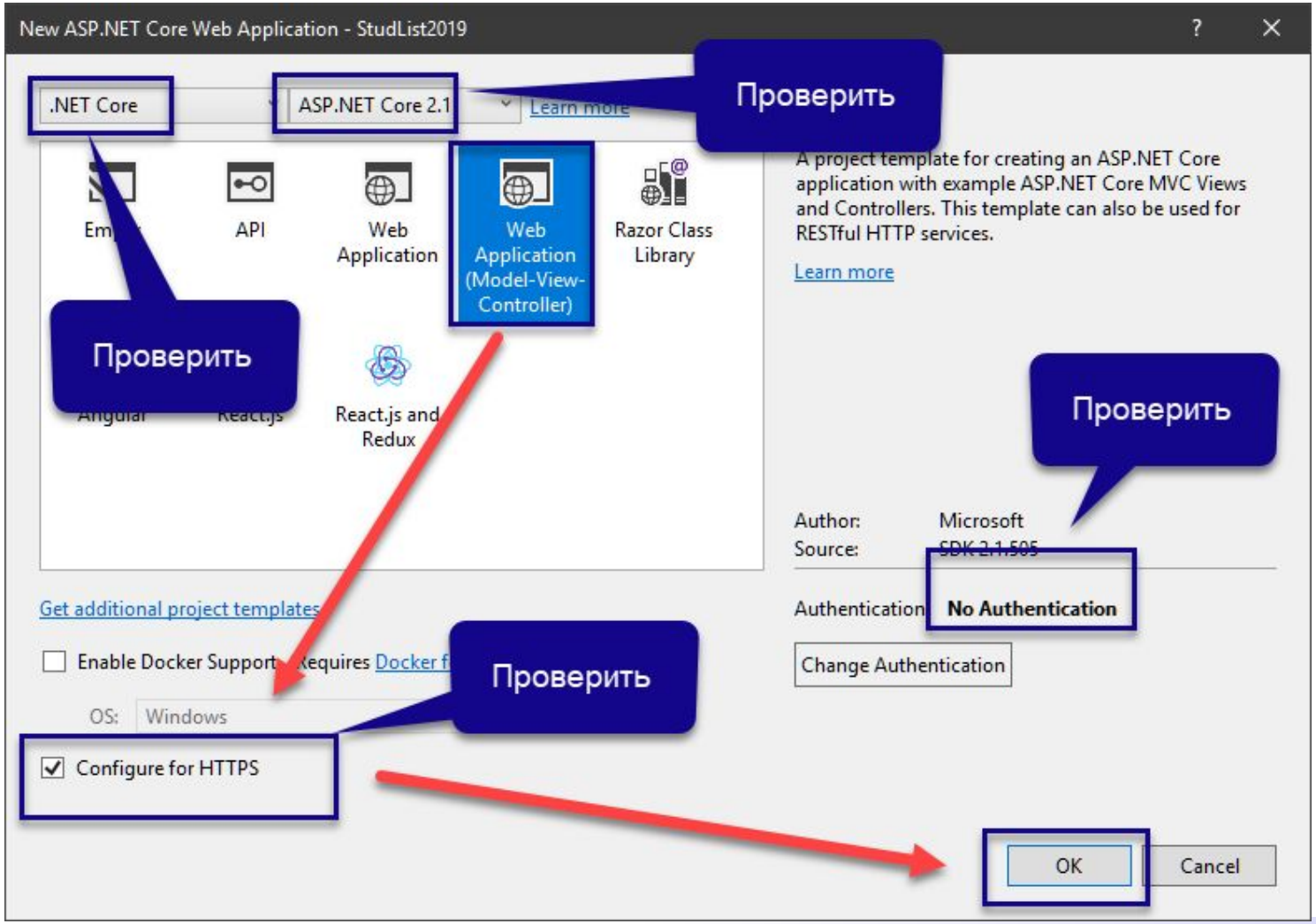
Создание проекта

ASP.NET Core

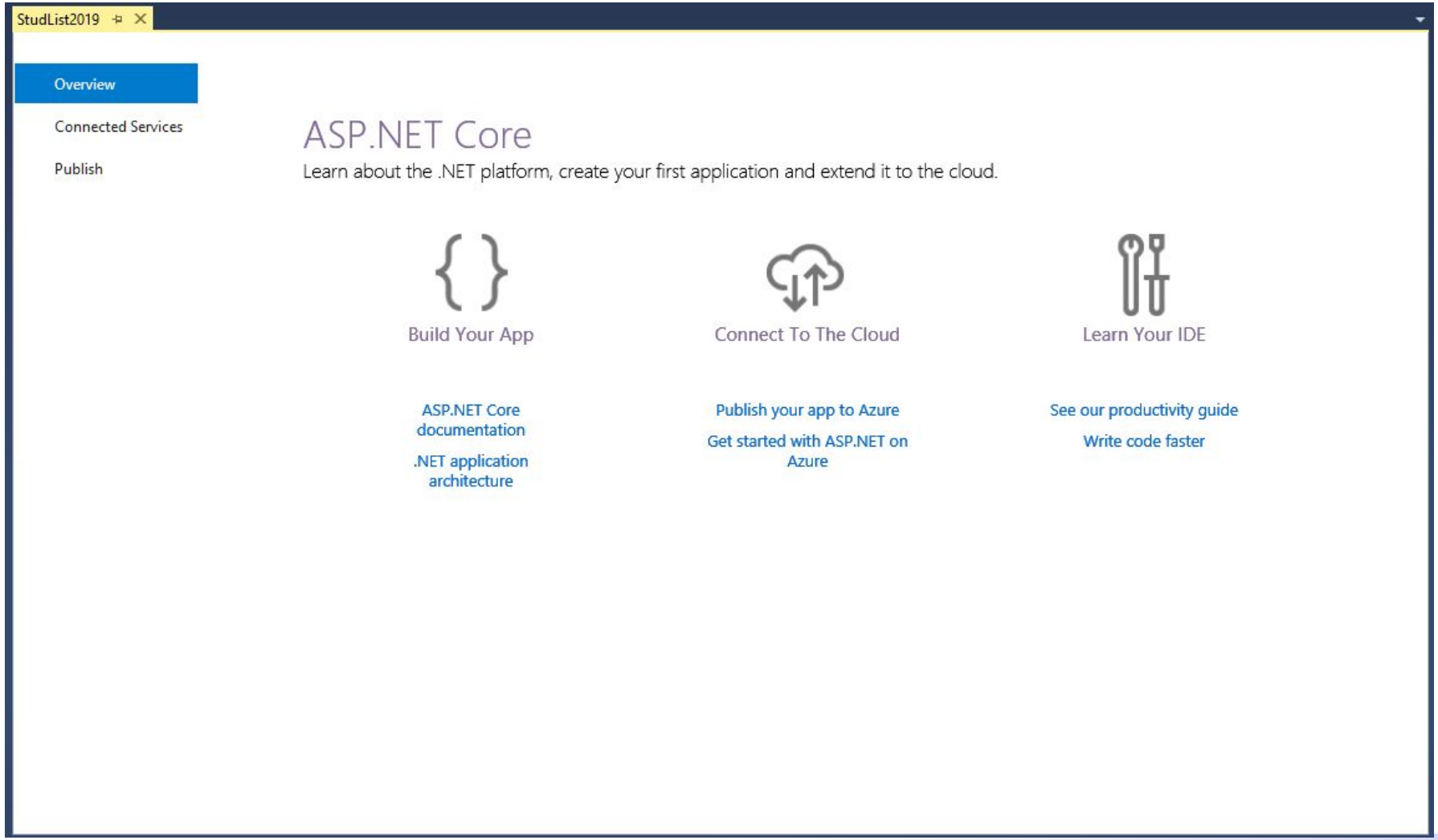


- Обратите внимание на тип проекта – ASP.NET Core

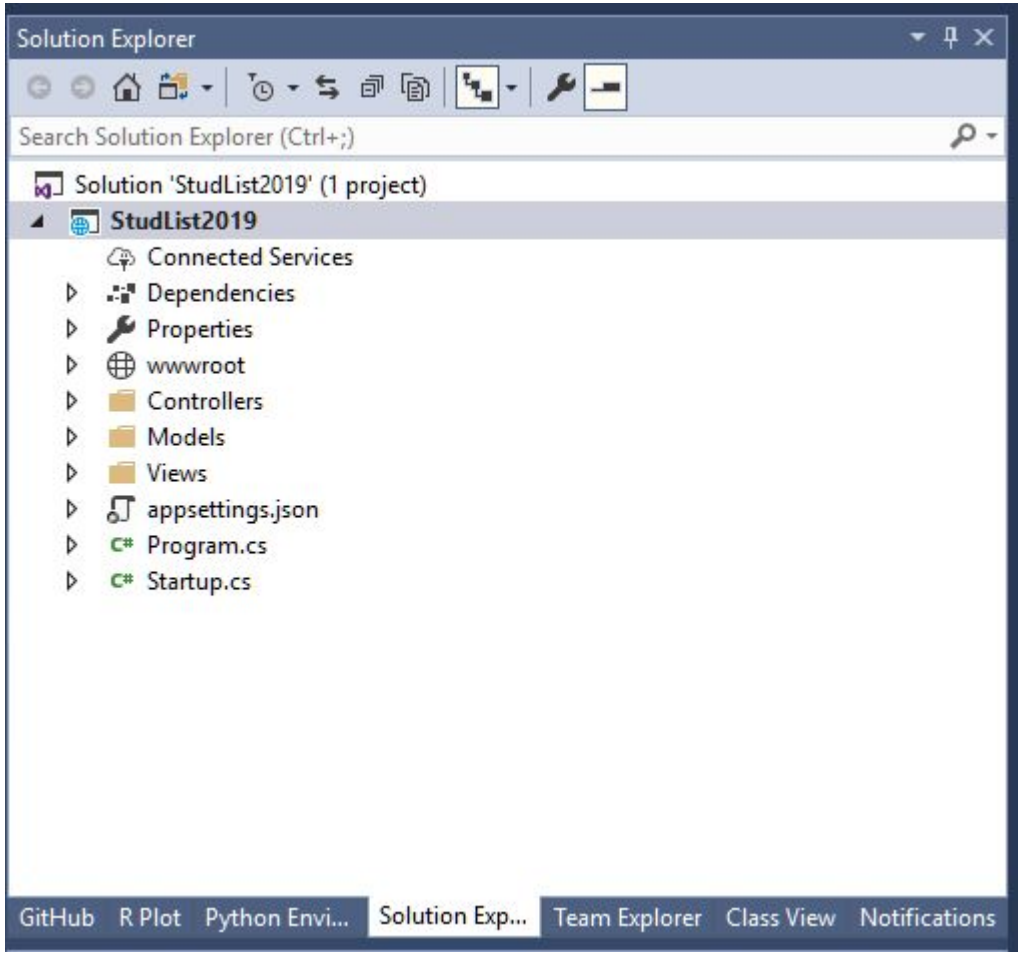
Настройте свойства проекта



Страница проекта



Структура папок



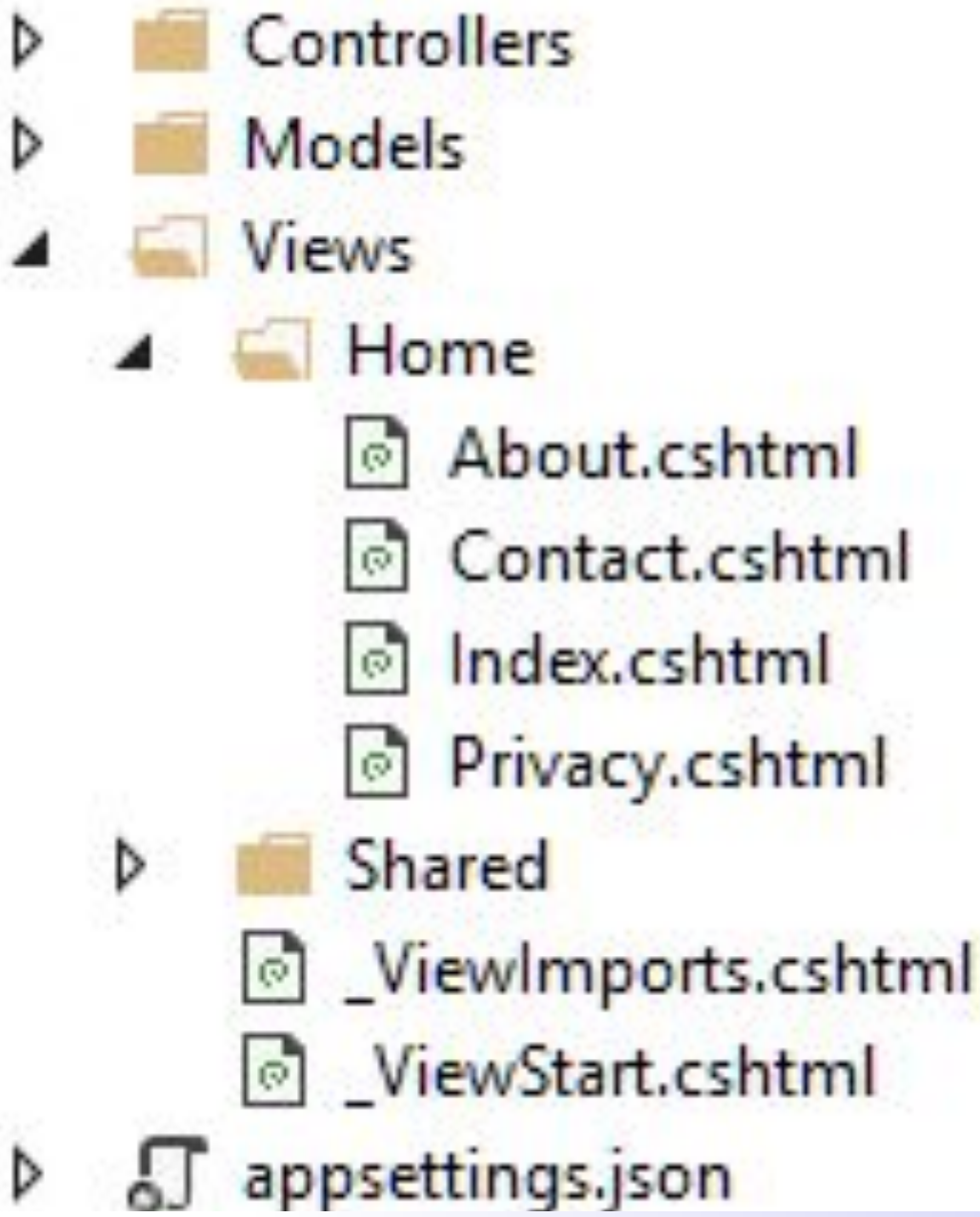
О структуре проекта

- Ожидается, что файлы, которые вы создадите лежат в «правильных» местах, поэтому обращайте внимание на место создания файлов
- Назначение папок:
  - Controllers – контроллеры для классов моделей
  - Models – модели (классы со свойствами)
  - Views – виды (страницы для отображения в браузере, точнее – наборы страниц)

О страницах

- Среда создает специальные страницы, в которых имеются как элементы, единые для всего сайта, так и места под заполнение поступающей информацией.
- За заполнение страниц отвечают связанные с ними программы на C# (их можно посмотреть в Solution Explorer, открыть, но лучше не редактировать ☺)

Папка Home содержит страницы «о компании», контакты, страница для отображения по умолчанию Index и т.д.



Общий шаблон

- Чтобы все страницы выглядели одинаково, они строятся на основании общего шаблона.
- Этот шаблон хранится в папке Shared
- Он, грубо говоря, состоит из двух частей:
  - Навигационное меню вверху страницы и подвал внизу
  - Место между ними для заполнения (RenderBody)
- Остальные страницы получаются так:
  - Сначала применяется шаблон,
  - Затем внутрь шаблона вставляется страница (например, About)
  - Скомпонованная итоговая страница «просчитывается», если это необходимо программой (например, выводятся данные из БД через контроллер)

Общий шаблон

Файл шаблона – Layout
Содержит пункты меню в верхней части экрана – Index, About, Contact

Файл шаблона – Layout
Место под переменные данные

StudList2019 _Layout.cshtml

```
16        <link rel="stylesheet" href="~/css/site.min.css" asp-append-version="true" />
17    </environment>
18 </head>
19 <body>
20     <nav class="navbar navbar-inverse navbar-fixed-top">
                                                    ta-toggle="collapse" data-target=".navbar-collapse">
                                                    pan>
29            <a asp-area="" asp-controller="Home" asp-action="Index" class="navbar-brand">StudList2019</a>
30        </div>
31        <div class="navbar-collapse collapse">
32            <ul class="nav navbar-nav">
33                <li><a asp-area="" asp-controller="Home" asp-action="Index">Home</a></li>
34                <li><a asp-area="" asp-controller="Home" asp-action="About">About</a></li>
35                <li><a asp-area="" asp-controller="Home" asp-action="Contact">Contact</a></li>
36            </ul>
37        </div>
38    </div>
39    </nav>
40
41    <partial name="_CookieConsentPartial" />
42
43    <div class="container body-content">
44        @RenderBody()
45        <hr />
46        <footer>
47            <p>&copy; 2019 - StudList2019</p>
48        </footer>
49    </div>
50
51    <environment include="Development">
52        <script src="~/lib/jquery/dist/jquery.js"></script>
                                                    cript>
                                                    /script>
```

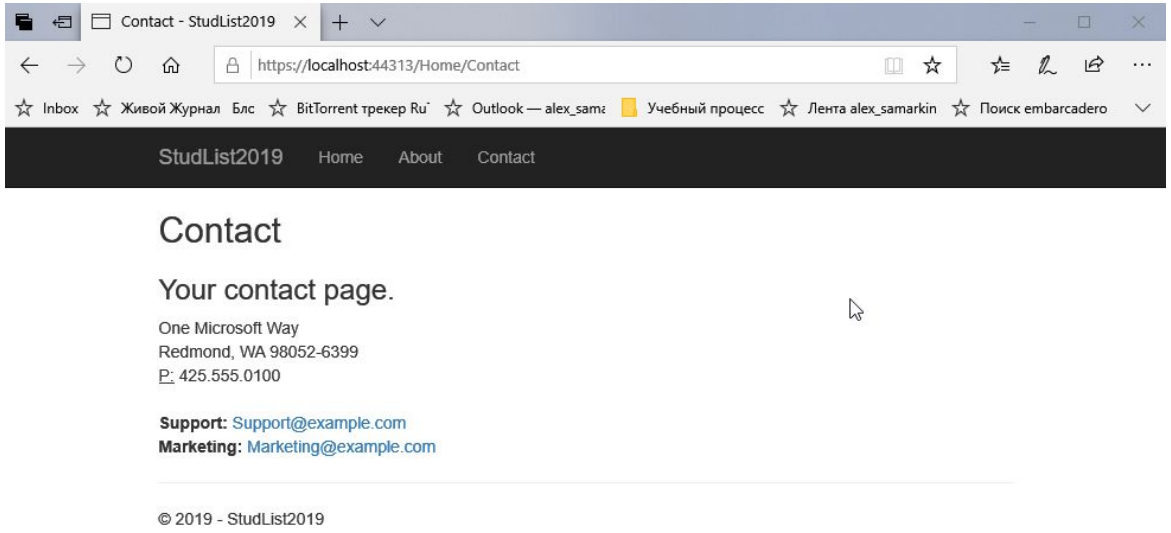
Privacy.cshtml
Shared
_CookieConsentPartial.cshtml
_Layout.cshtml
_ValidationScriptsPartial.cshtml
Error.cshtml
ViewImports.cshtml

- Обратите внимание, следующая строка:
  - `<li><a asp-area="" asp-controller="Home" asp-action="Contact">Contact</a></li>`
- Соответствует папке Home и файлу Contact внутри папки, с одной стороны
- С другой – она отображает пункт меню «Contact» в навигационной панели и создает гиперссылку
- Аналогично работают и два других пункта
- **Если надо добавить новые пункты навигации, то их надо добавлять здесь**
- Поле RenderBody замещается последующими, специализированными страницами

Запуск приложения

- Наше приложение уже работоспособно
- Для его запуска нужен или веб-сервер или встроенный отладочный сервер (в примере – сервер IIS, входящий в состав Windows)
- Чтобы сократить время запуска рекомендуется запускать упрощенную версию отладчика клавишами **Ctrl+F5**
- Приложение откроется в виде страницы в браузере

Пример страницы приложения



Что дальше

Алгоритм последующих действий

Создать модель (класс со свойствами) данных

Создать контроллер, сгенерировать контекст и виды

Создать БД

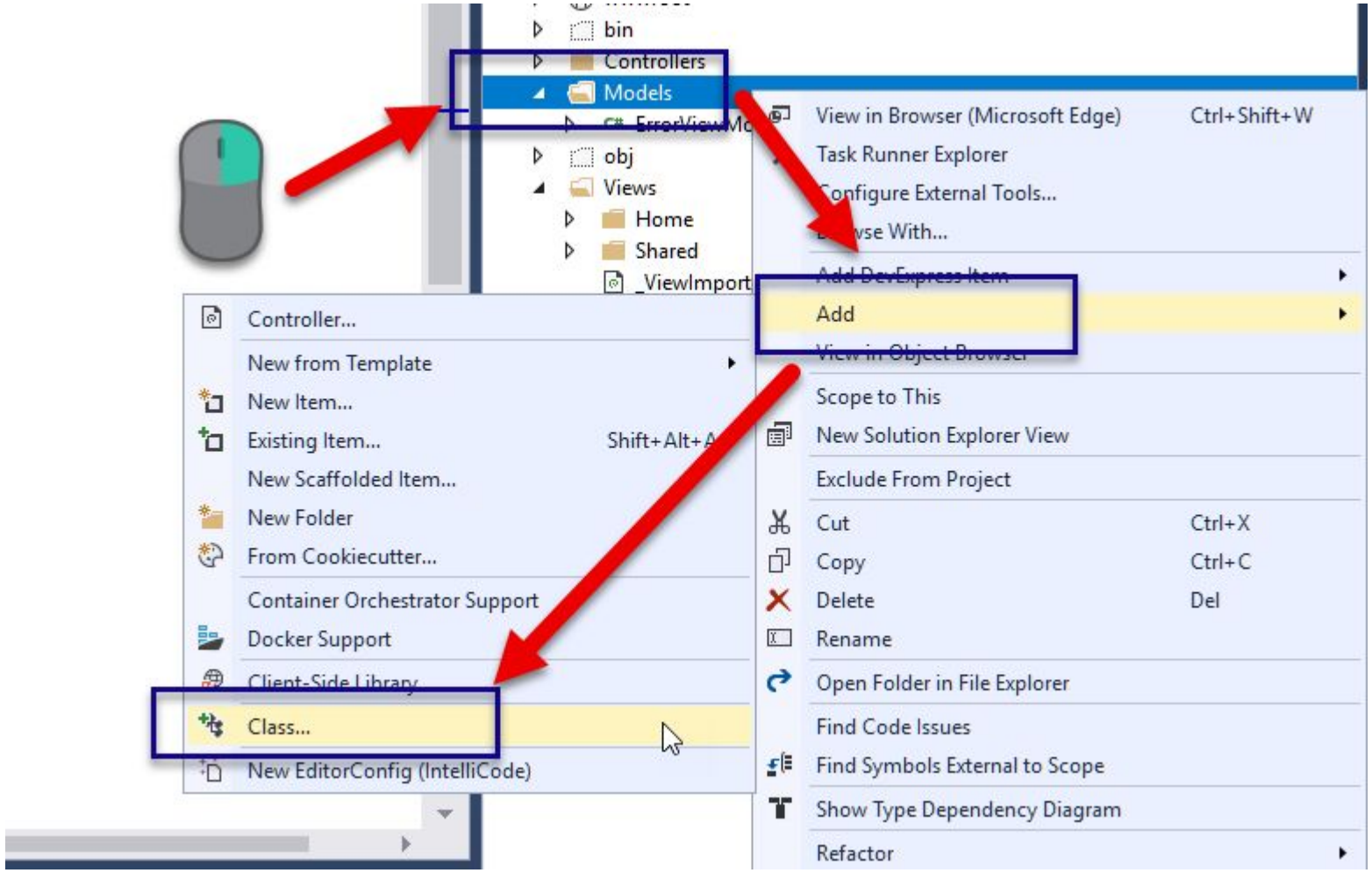
Создать таблицу под данные в БД

Модель, контроллер и виды

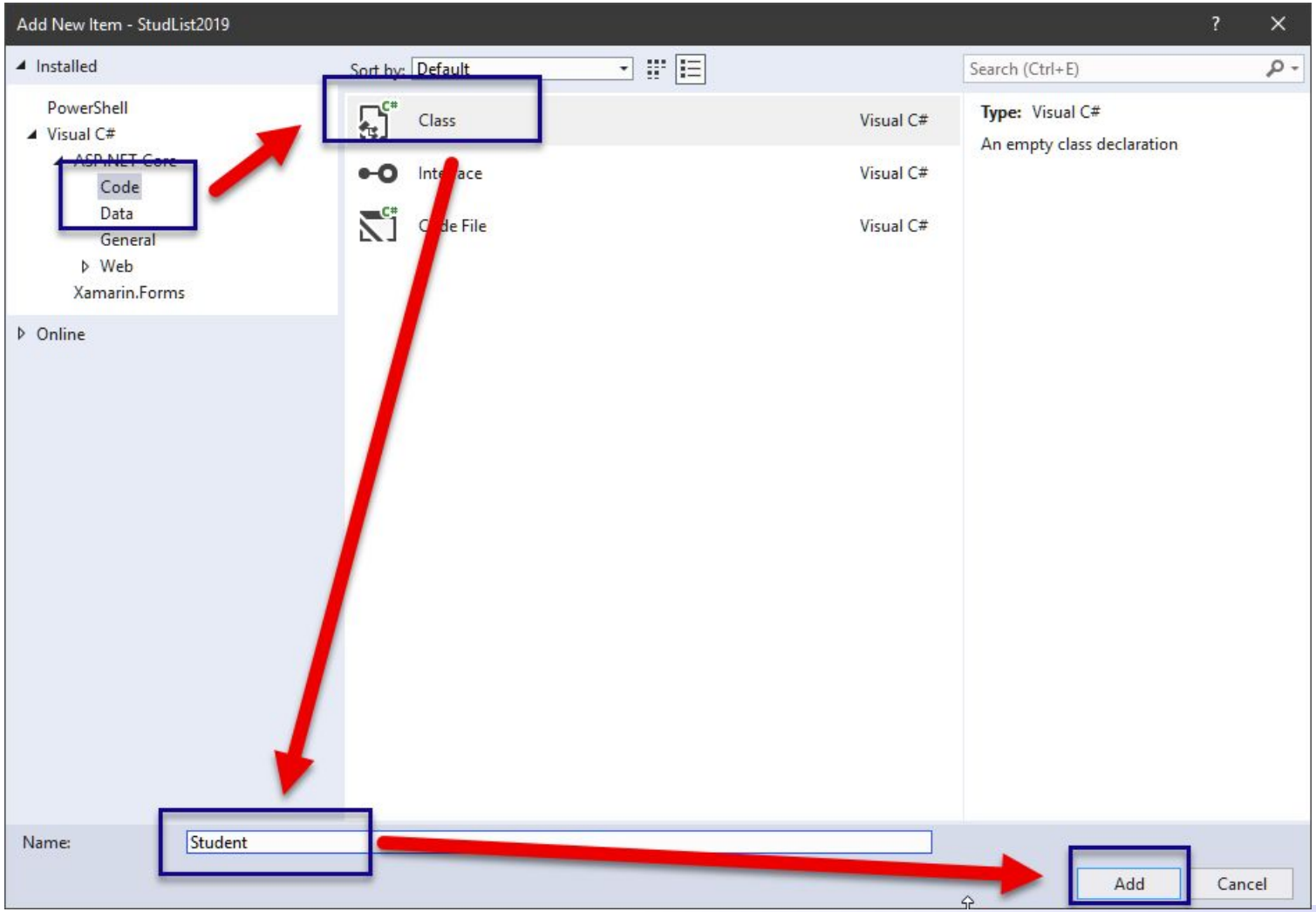
Что и где

- Модель – это public класс со свойствами
- Как минимум, одно из свойств должно быть int и иметь название ID – это поле первичного ключа в БД
- Модели создаются в папке Models
- Принято называть файлы именами в единственном числе (Name а не Names)
- Контроллер создается для каждого класса модели в папке Controllers
- Контроллеры регистрируются в контексте БД
  - Первый контекст генерируется, последующие классы лишь модифицируют его
- При создании контроллера мы автоматически будем создавать виды

Добавление модели



Модель класса Student



Заготовка класса

Debug | Any CPU | IIS Express

Student.cs | StudList2019

StudList2019

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5
6  namespace StudList2019.Models
7  {
8      public class Student
9      {
10     }
11 }
12
```

Первичный ключ (обязательно!!!)

Student.cs* StudList2019

StudList2019

```
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Threading.Tasks;
5
6   namespace StudList2019.Models
7   {
        0 references
8       public class Student
9       {
            0 references | 0 exceptions
10          public int ID { get; set; }
11
12      }
13  }
14
```

Добавьте остальные поля

StudList2019 | StudList2019.Mod

```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Threading.Tasks;
5
6   namespace StudList2019.Models
7   {
        0 references
8       public class Student
9       {
            0 references | 0 exceptions
10          public int ID { get; set; }
11
            0 references | 0 exceptions
12          public string Name1 { get; set; }
            0 references | 0 exceptions
13          public string Name2 { get; set; }
            0 references | 0 exceptions
14          public string Name3 { get; set; }
15
            0 references | 0 exceptions
16          public DateTime BirthDate{ get; set; }
17
            0 references | 0 exceptions
18          public string PhoneNum { get; set; }
            0 references | 0 exceptions
19          public string Code { get; set; }
            0 references | 0 exceptions
20          public bool Dorm { get; set; }
21
22      }
23  }
24
```

Тонкая настройка

- Уточнить свойства поля можно при помощи его атрибутов (небольшой функции в квадратных скобках перед свойством.
- Например, DisplayName задает название поля для отображения на экране (Name1 – Фамилия и т.п.)
- Другой атрибут - [DataType(DataType.Date)] – означает, что надо отобразить только дату из поля типа DateTime

Следует добавить в раздел using строки

```
using System.ComponentModel;
using System.ComponentModel.DataAnnotations;
```

Или убедиться в их наличии

- Рекомендуется задавать хотя бы отображаемые имена полей на русском языке

Полный текст файла модели

Добавлены using для аннотаций

Каждому полю дано отображаемое имя через атрибут

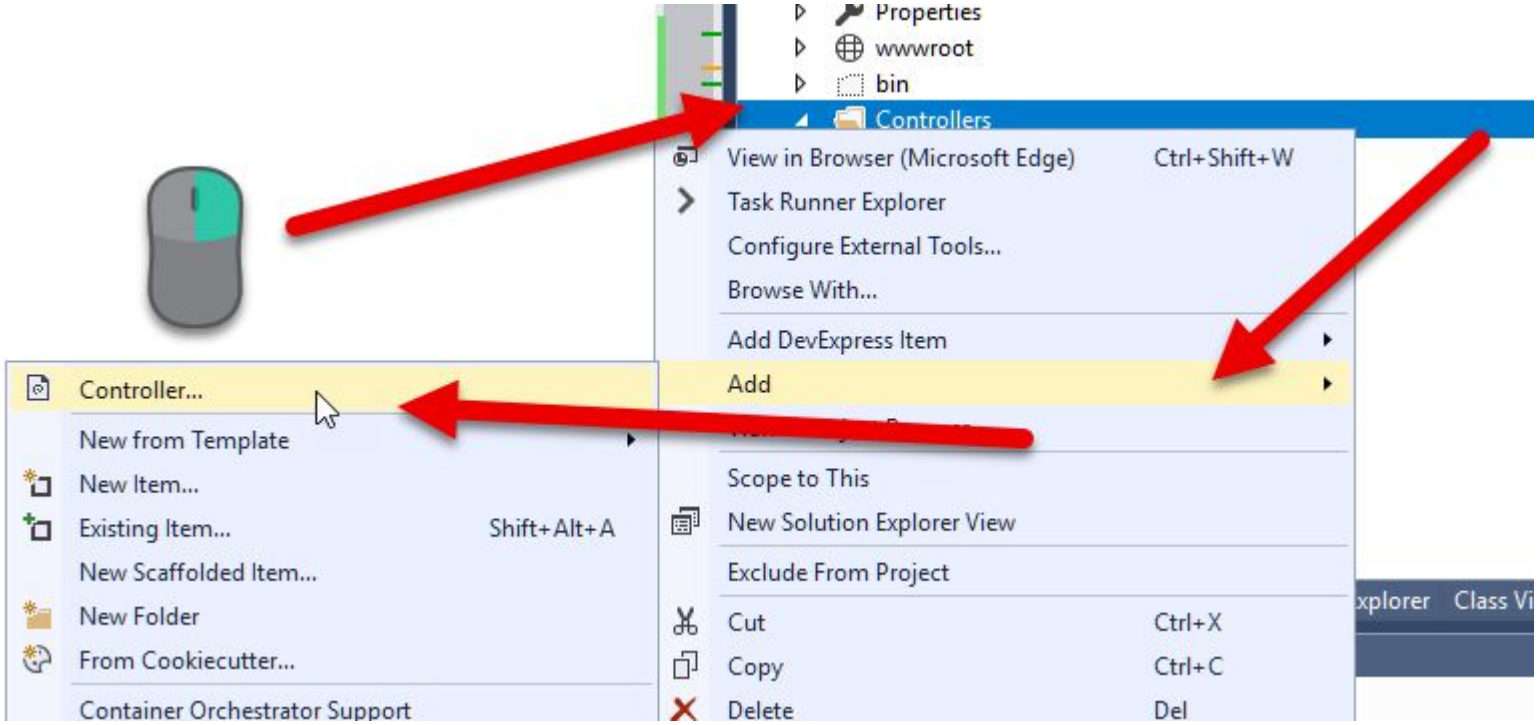
Поле дата рождения, помимо имени, получило еще и атрибут, показывающий, что следует отображать только часть данных с датой

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.ComponentModel.DataAnnotations;
5  using System.Linq;
6  using System.Threading.Tasks;
7
8  namespace StudList2019.Models
9  {
       0 references
10     public class Student
11     {
           0 references | 0 exceptions
12         public int ID { get; set; }
13         [DisplayName("Фамилия")]
           0 references | 0 exceptions
14         public string Name1 { get; set; }
15         [DisplayName("Имя")]
           0 references | 0 exceptions
16         public string Name2 { get; set; }
17         [DisplayName("Отчество")]
           0 references | 0 exceptions
18         public string Name3 { get; set; }
19
20         [DisplayName("Дата рождения")]
21         [DataType("Date")]
           0 references | 0 exceptions
22         public DateTime BirthDate{ get; set; }
23         [DisplayName("Номер телефона")]
           0 references | 0 exceptions
24         public string PhoneNum { get; set; }
25         [DisplayName("Зачетная книжка")]
           0 references | 0 exceptions
26         public string Code { get; set; }
27         [DisplayName("Нужно общежитие")]
           0 references | 0 exceptions
28         public bool Dorm { get; set; }
29
30     }
31 }
32
```

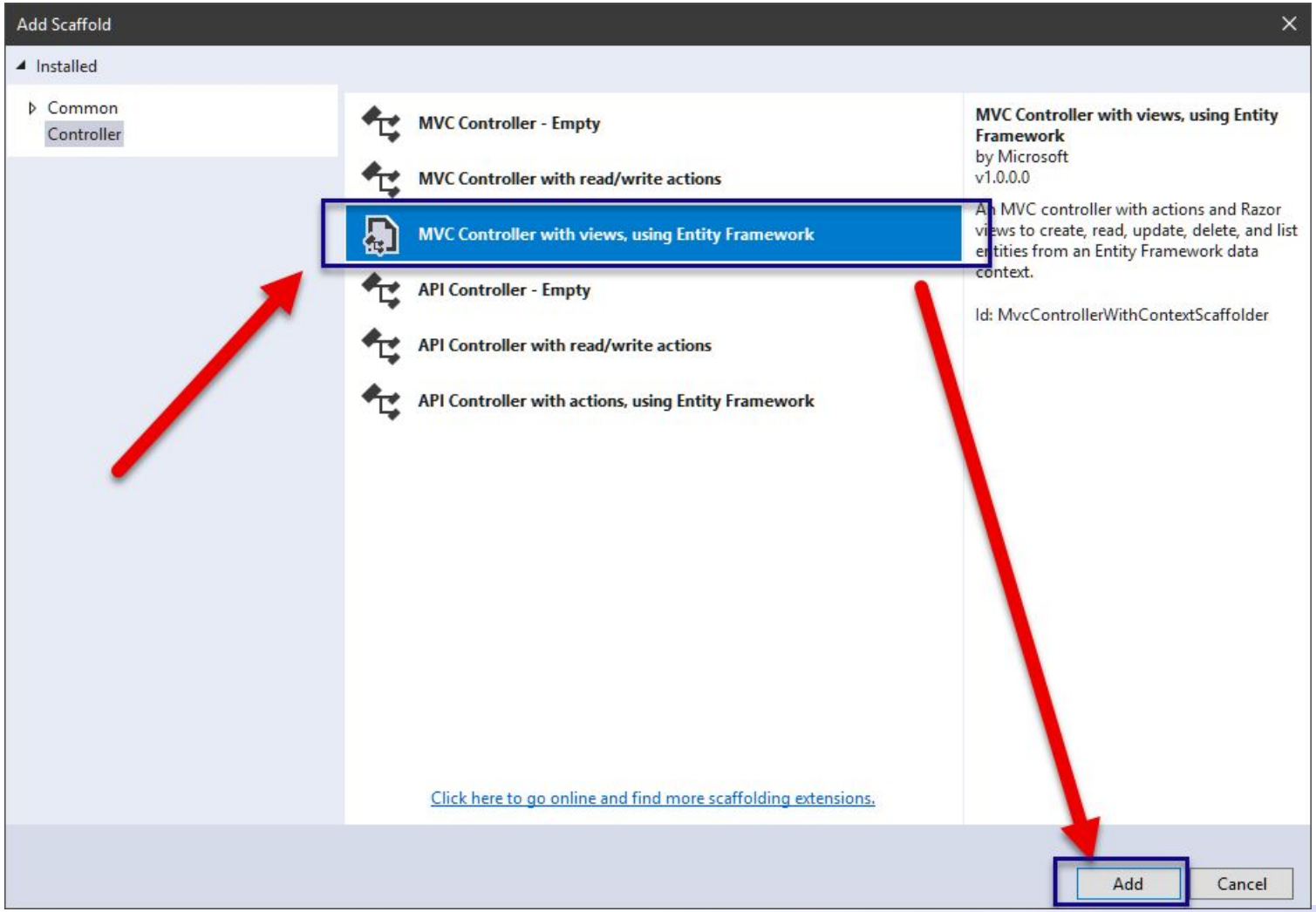
Добавление контроллера

- Напомним, что контроллер создается в папке контроллеров
- Контроллер создается для класса
- Мы создадим контроллер и сгенерируем контекст
  - Если классов больше чем один, то контекст будет создан один раз, а далее будет только корректироваться
- Мы создадим контроллер и сгенерируем необходимые виды (страницы для просмотра, редактирования и т.п.)

Добавление контроллера



Контроллер и автоматически создаваемые виды



Диалог добавления контроллера

Необходимо выбрать класс из списка

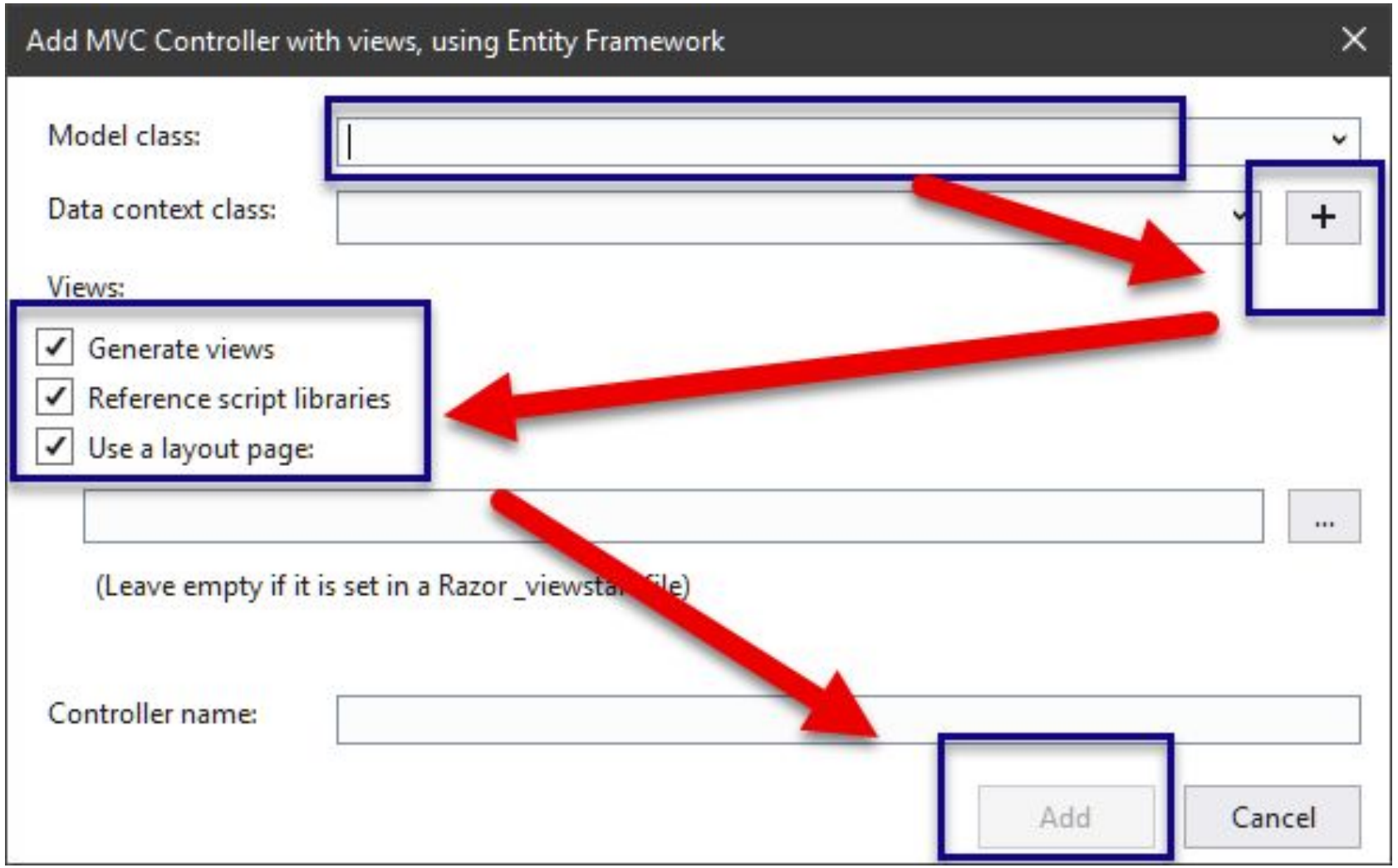
У нас не было создано контекста – его надо добавить

Мы хотим создать виды

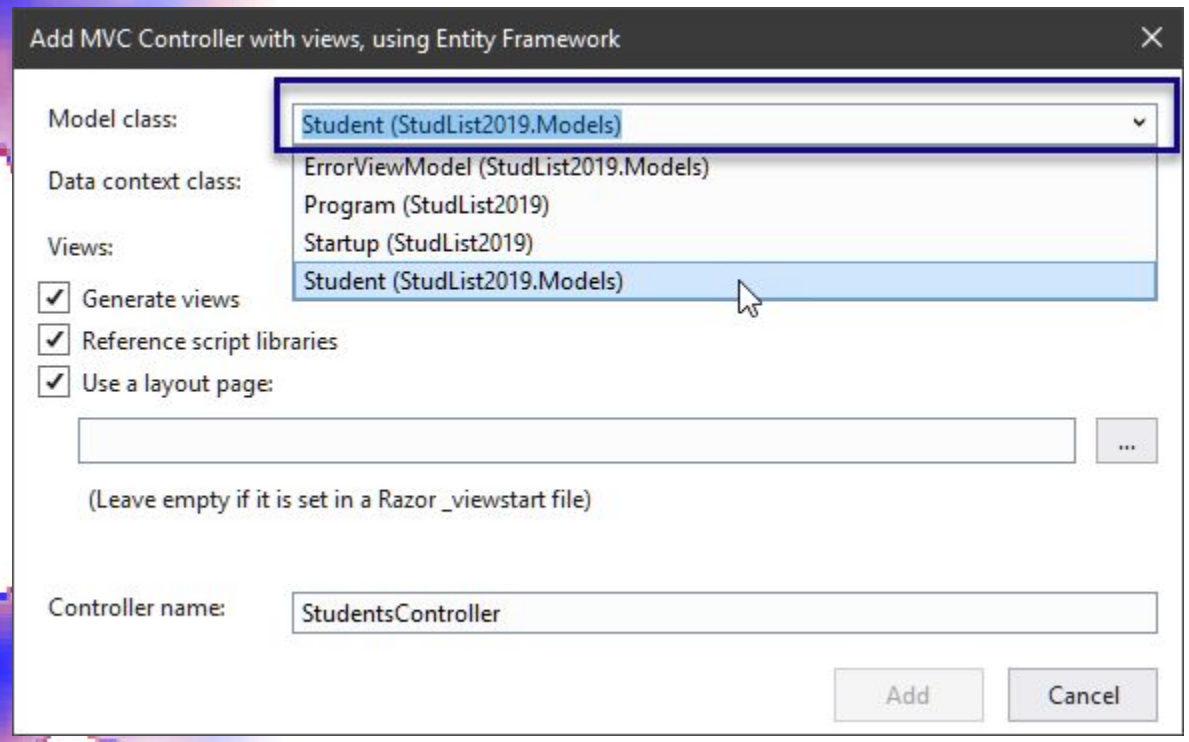
Кнопка Add – добавить – будет доступна по заполнении полей

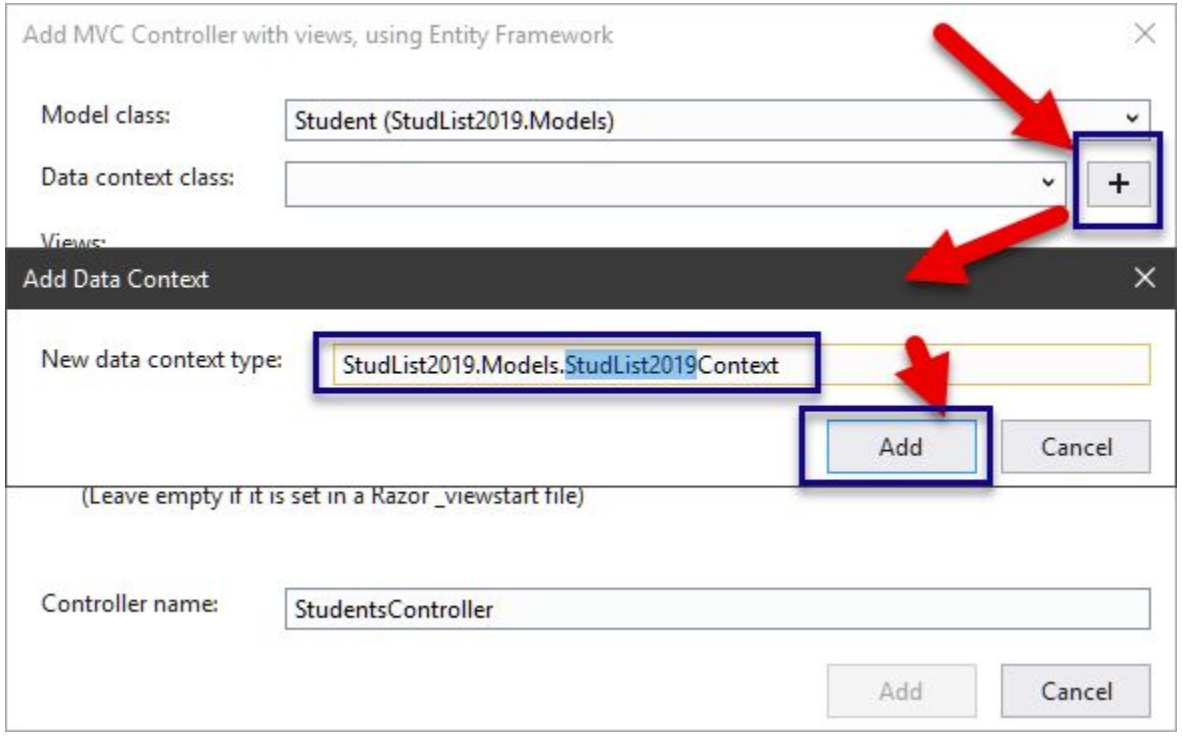
Обратите внимание(!!!):

- Имя модели выбирается из списка
- Название контекста будет сгенерировано автоматически, менять его НЕ НАДО
- Страница шаблона (Layout) у нас уже есть, это поле следует оставить пустым

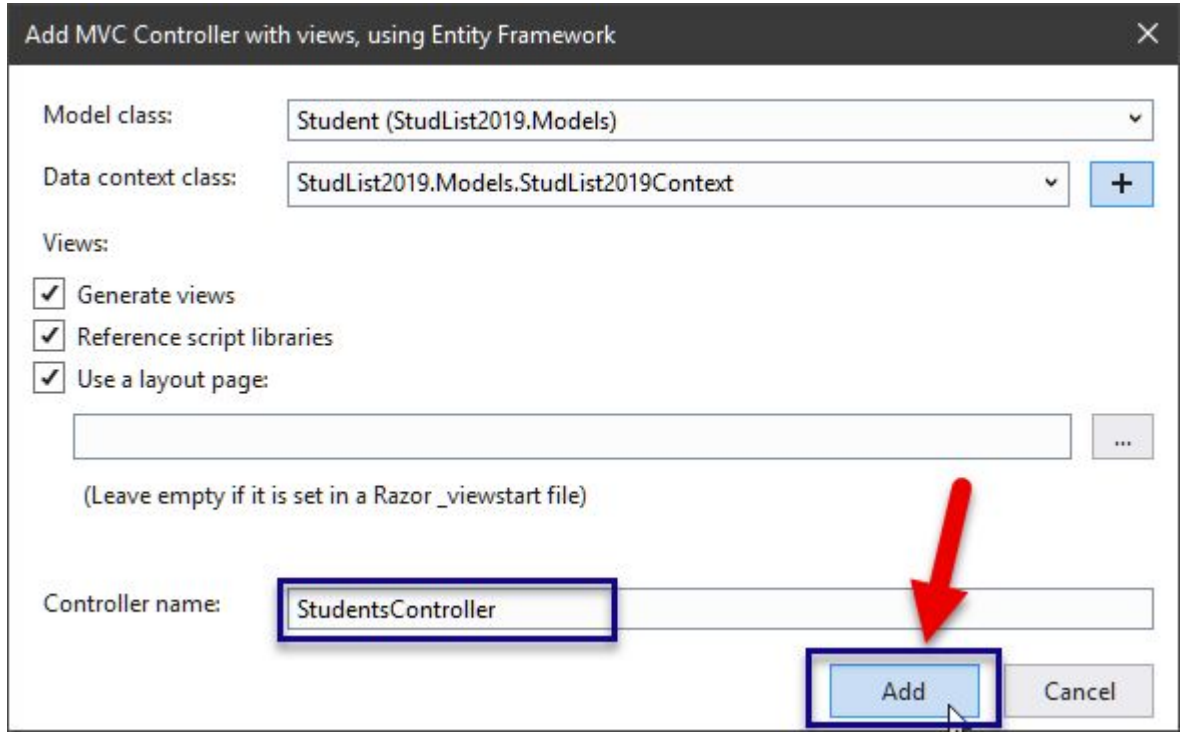


Выбор модели и имени контекста





Контроллер

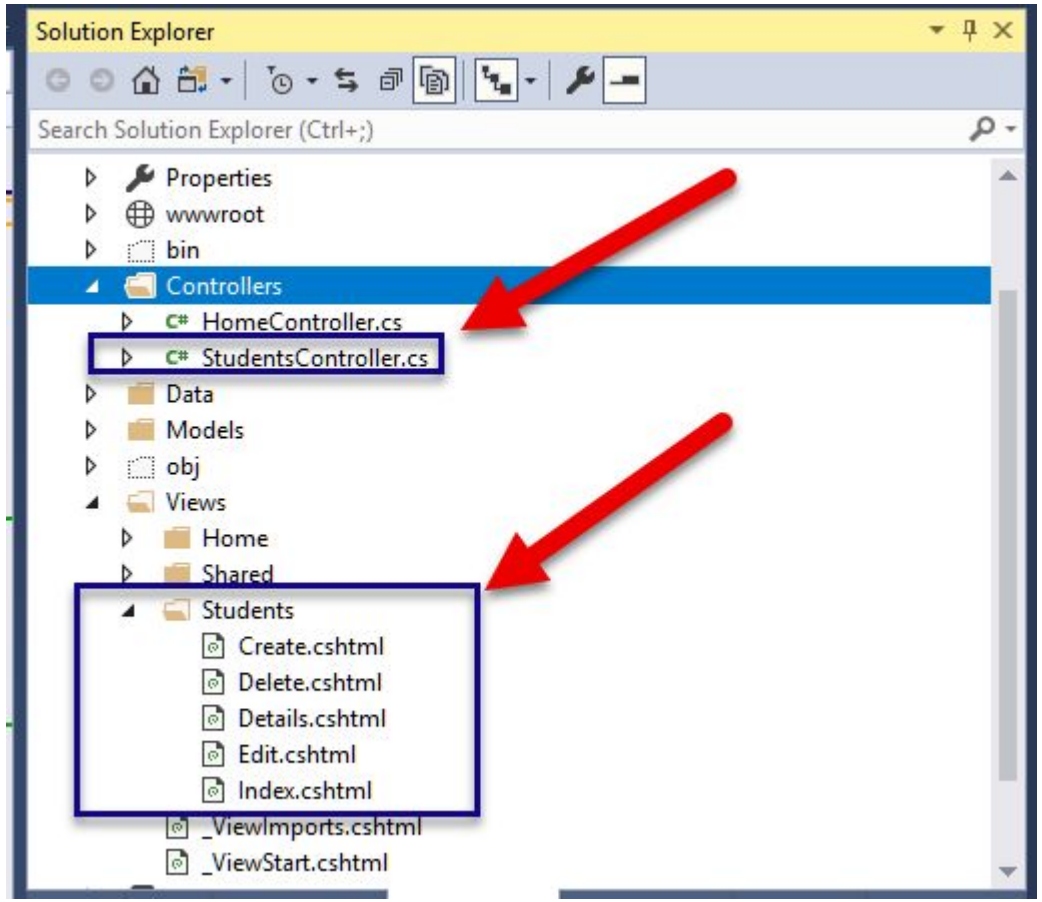

Add MVC Controller with views, using Entity Framework
Model class:
Student (StudList2019.Models)
Data context class:
StudList2019.Models.StudList2019Context
Views:
Generate views
Reference script libraries
Use a layout page:
...
(Leave empty if it is set in a Razor _viewstart file)
Controller name:
StudentsController
Add
Cancel

Проверка

- Просмотрите папку видов и убедитесь, что добавлена подпапка студентов, а ней – файлы для работы с данными

- Должен быть создан контроллер,
- Подпапка для видов
- Внутри подпапки – index, detail, edit, delete, create – страницы для просмотра списка данных, подробных сведений о выбранной записи, форма редактирования, удаления и создания (соответственно)

Убедитесь, что созданы эти файлы



Чего не хватает?

- Нам по-прежнему нужна БД для постоянного хранения данных (в принципе можно и не хранить данные в БД)
- Для создания правильной БД нужно выполнить миграцию.

Миграция

Миграция

- Это процесс создания или обновления БД, изменения структуры данных, таблиц и т.п.
- В процессе миграции также создается подключение к БД (во всяком случае – строка такого подключения).
- Для миграции необходим **провайдер БД.**
- Такие провайдеры предлагает как Microsoft (для собственного SQL Server), так и, например, разработчики PostgresSQL, MySQL и ряда других БД.

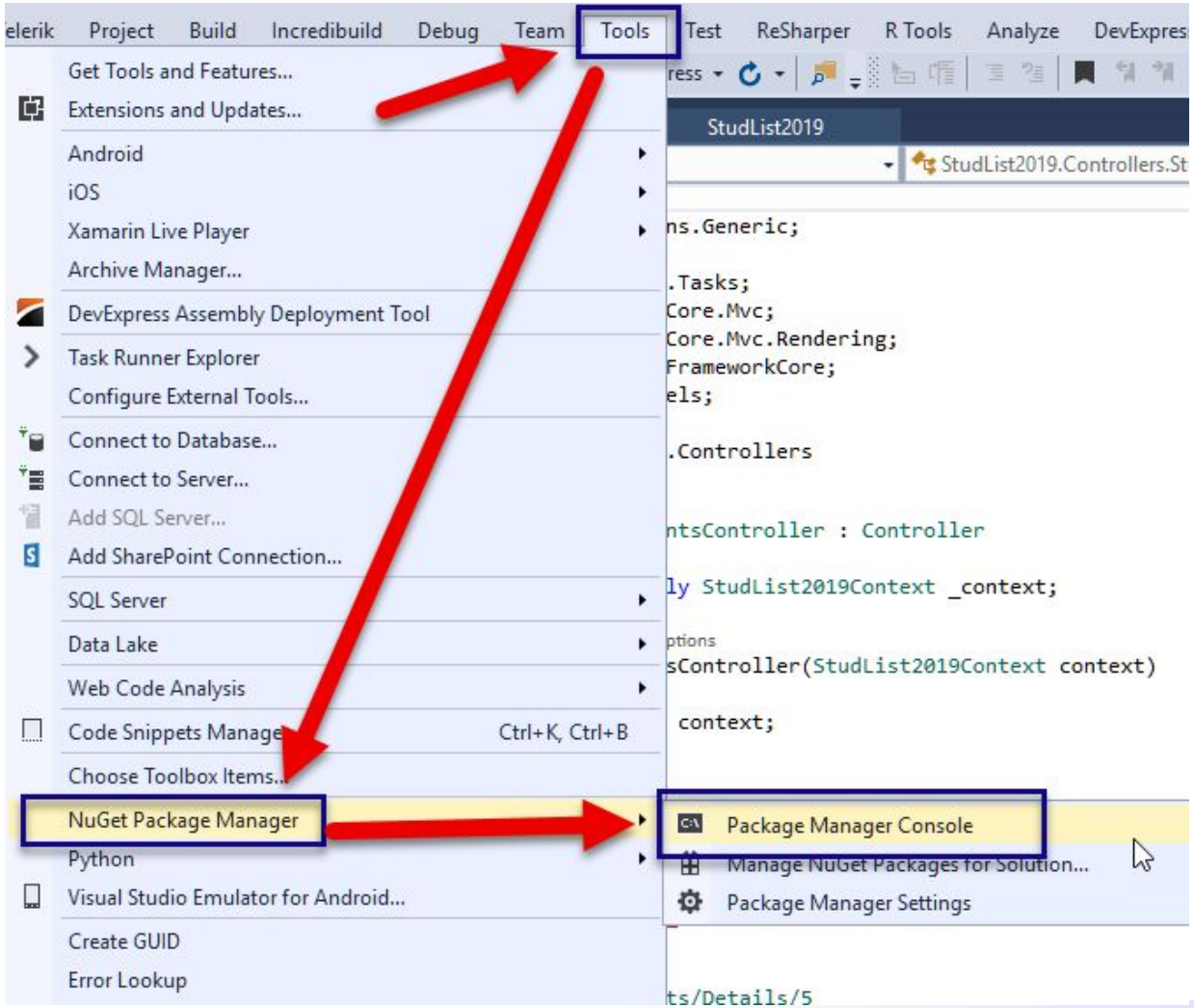
Миграцию делают командами в консоли

- Основных команд, связанных с миграцией, всего три:
  - Add-Migration InitialMigration – первая миграция. Создает БД и подключение, если есть модели – добавляет таблицы в БД в соответствии с моделями и связями между ними
  - Add-Migration Student – добавляет к существующей БД новую таблицу (после добавления модели) или обновляет определение для существующей модели при ее существенных изменениях
- Можно рассматривать миграцию как слепок с текущего состояния модели, который нужно применить к БД
- Несколько миграций записываются в виде последовательности файлов
  - Remove-Migration – удаляет последнюю не примененную к БД миграцию

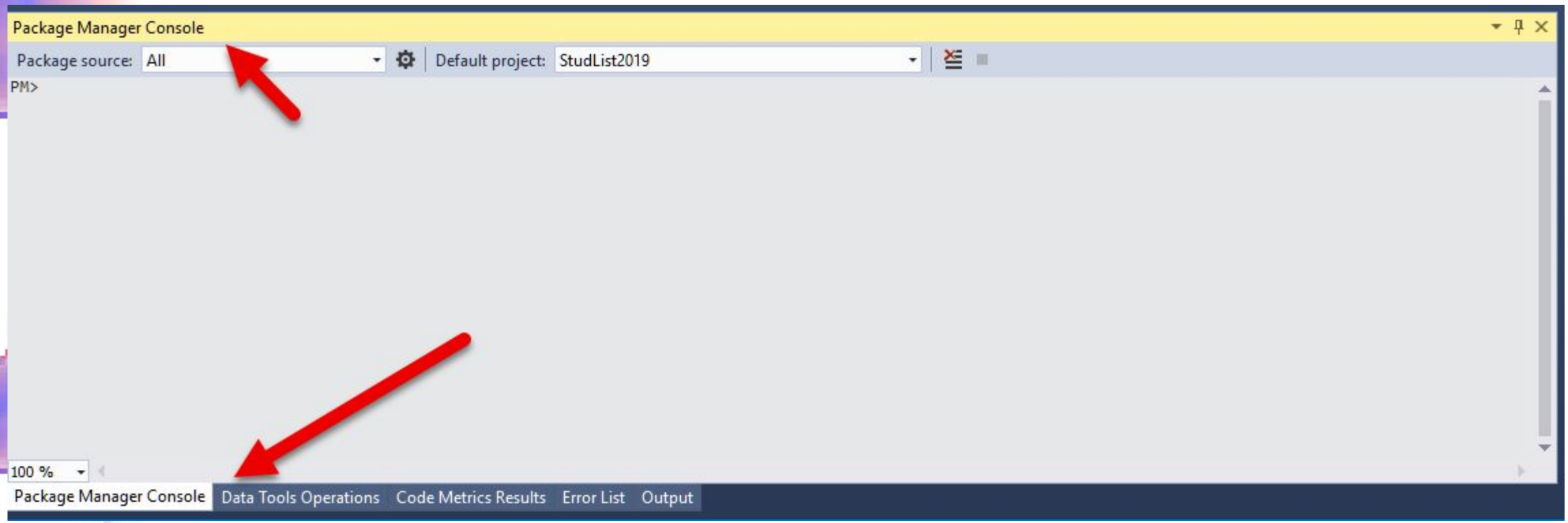
Обновление БД

- Обновление БД делает еще одна команда:
  - Update-Database
- Эта команда ФАКТИЧЕСКИ выполняет миграцию
- Таким образом, вы можете скорректировать действия по миграции, добавить или удалить некоторые из них, и лишь затем применить их к БД командой Update-Database
- База создается в каталоге пользователя C:\Users\<UserName>, где UserName – имя текущего пользователя Windows

Вызов консоли



Консоль ввода команд



Начальная миграция

Package Manager Console

Package source: All

```
PM> Add-Migration InitialMigration
```

Package Manager Console

Package source: All Default project: StudList2019

```
PM> Add-Migration InitialMigration
The EF Core tools version '2.1.1-rtm-30846' is older than that of the runtime '2.1.8-servicing-32085'. Update the tools for the latest features and bug fixes.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 2.1.8-servicing-32085 initialized 'StudList2019Context' using provider 'Microsoft.EntityFrameworkCore.SqlServer' with options: None
To undo this action, use Remove-Migration.
PM>
```

- После ввода команды (факультативно) просмотрите папку Migrations

Применение миграции

Package Manager Console

Package source: All

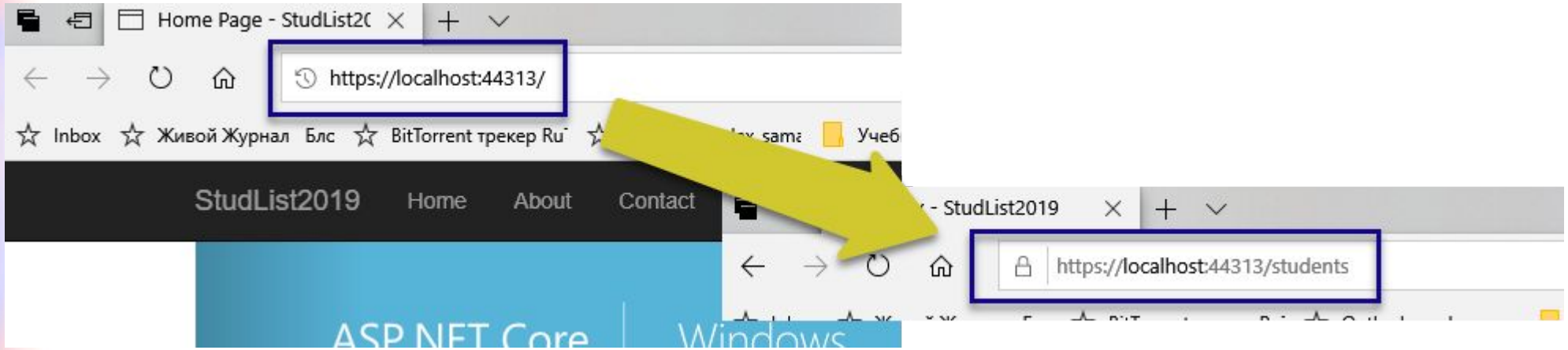
```
PM> Update-Database
```

Package Manager Console

Package source: All Default project: StudList2019

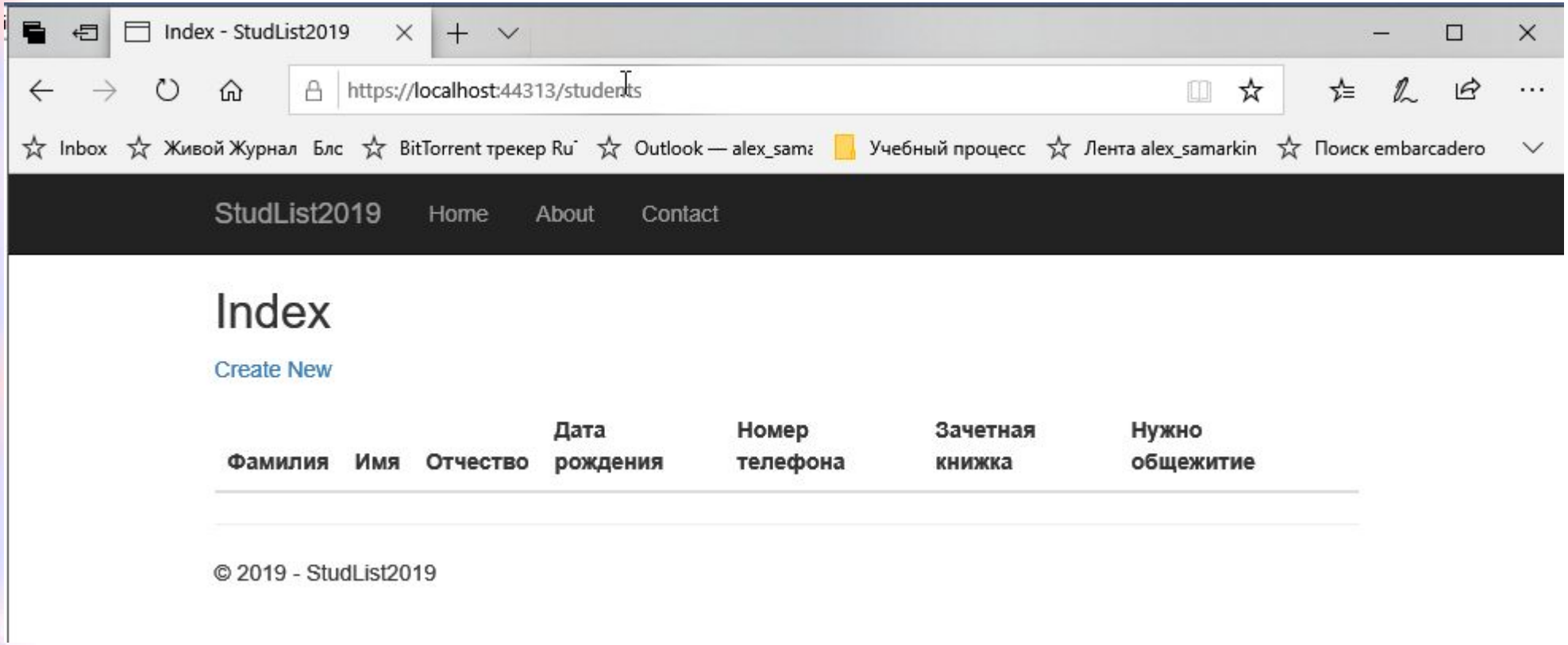
```
PM> Update-Database
The EF Core tools version '2.1.1-rtm-30846' is older than that of the runtime '2.1.8-servicing-32085'. Update the tools
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 2.1.8-servicing-32085 initialized 'StudList2019Context' using provider 'Microsoft.EntityFram
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (1,231ms) [Parameters=[], CommandType='Text', CommandTimeout='60']
      CREATE DATABASE [StudList2019Context-307888f0-5971-4998-a2e1-83da102de6ee];
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (611ms) [Parameters=[], CommandType='Text', CommandTimeout='60']
      IF SERVERPROPERTY('EngineEdition') <> 5
      BEGIN
          ALTER DATABASE [StudList2019Context-307888f0-5971-4998-a2e1-83da102de6ee] SET READ_COMMITTED_SNAPSHOT ON;
      END;
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (82ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      CREATE TABLE [__EFMigrationsHistory] (
          [MigrationId] nvarchar(150) NOT NULL,
          [ProductVersion] nvarchar(32) NOT NULL,
```

Первичное тестирование приложения



- Запустите приложение командой Ctrl+F5
- Откройте в браузере добавьте к адресу приложения папку Students

Убедитесь, что страница открывается (пока без данных)



Добавление пункта меню для доступа к списку студентов

```
<div class="navbar-collapse collapse">
    <ul class="nav navbar-nav">
        <li><a asp-area="" asp-controller="Home" asp-action="Index">Home</a></li>
        <li><a asp-area="" asp-controller="Home" asp-action="About">About</a></li>
        <li><a asp-area="" asp-controller="Home" asp-action="Contact">Contact</a></li>

        <li><a asp-area="" asp-controller="Students" asp-action="Index">Список  студентов</a></li>
    </ul>
</div>
```

- Откройте файл _Layout.cshtml (папка Shared)
- Скопируйте строку и отредактируйте ее как показано

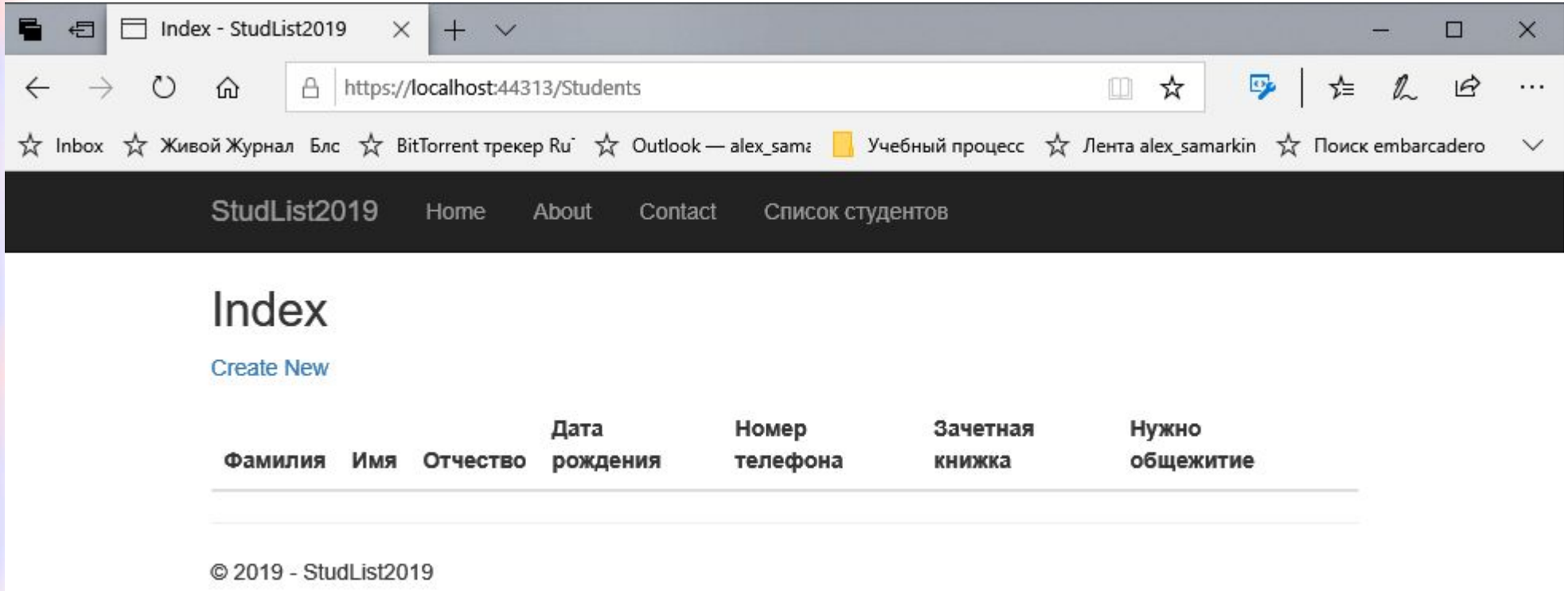
Почему именно так?

- Система ожидает определенного порядка размещения файлов, в соответствии с логикой приложения
- В частности:
  - Мы создали контроллер для студентов и папку для хранения видов
  - Если мы хотим показать некие данные, то мы должны указать ее контроллер (по соглашению это еще и папка с html файлами) и конкретное действие над данными (например, просмотр).
  - Действию соответствует вид – файл с тем же именем, что и действие типа cshtml.
- Студенты связаны с контроллером/папкой Students
- Базовое действие – просмотр списка – Index (на странице Index есть переход к другим действиям).

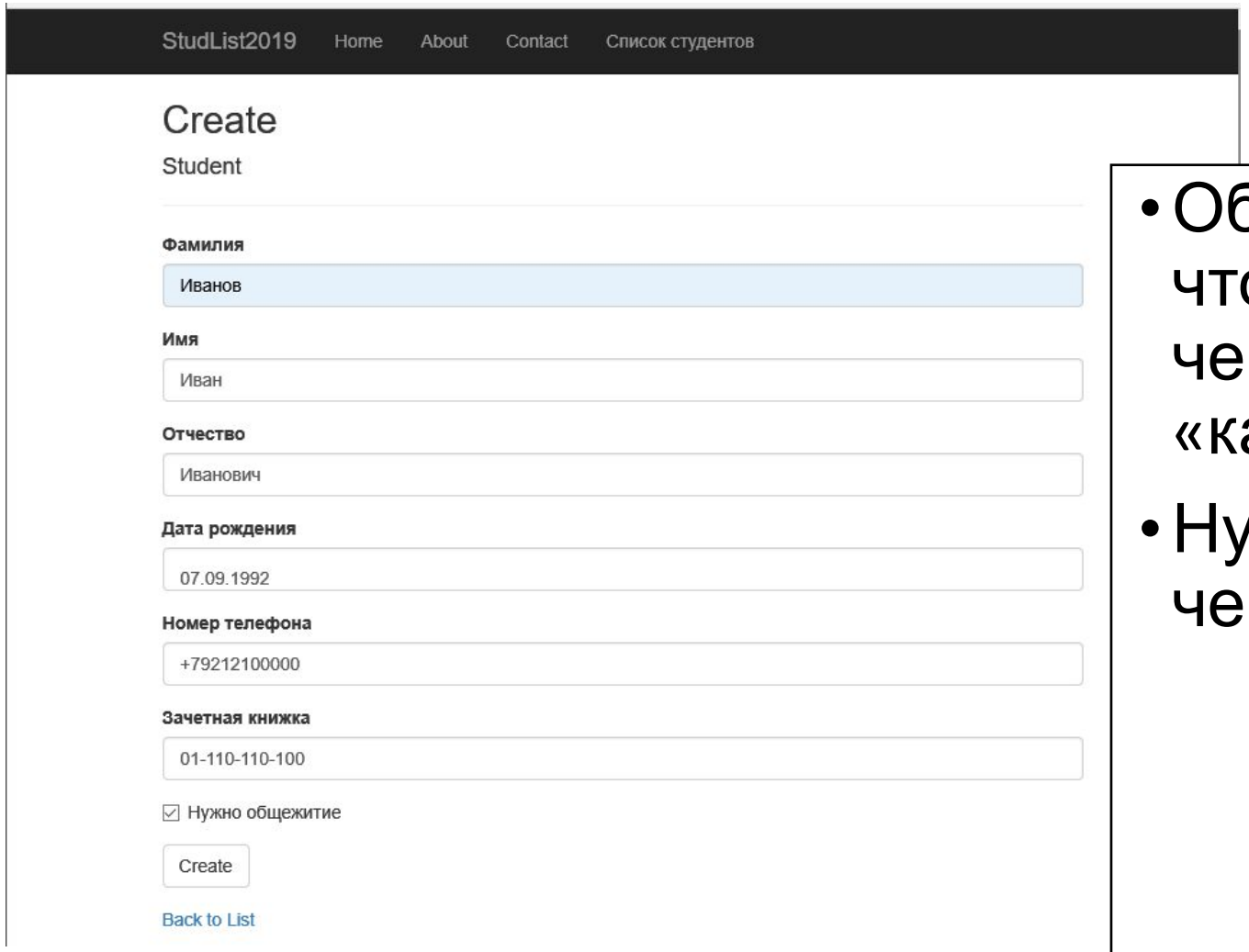
Завершение

- Запустите приложение
- Создайте 5-6 студентов
- Удалите 2-го
- Отредактируйте фамилию последнего

Страница списка

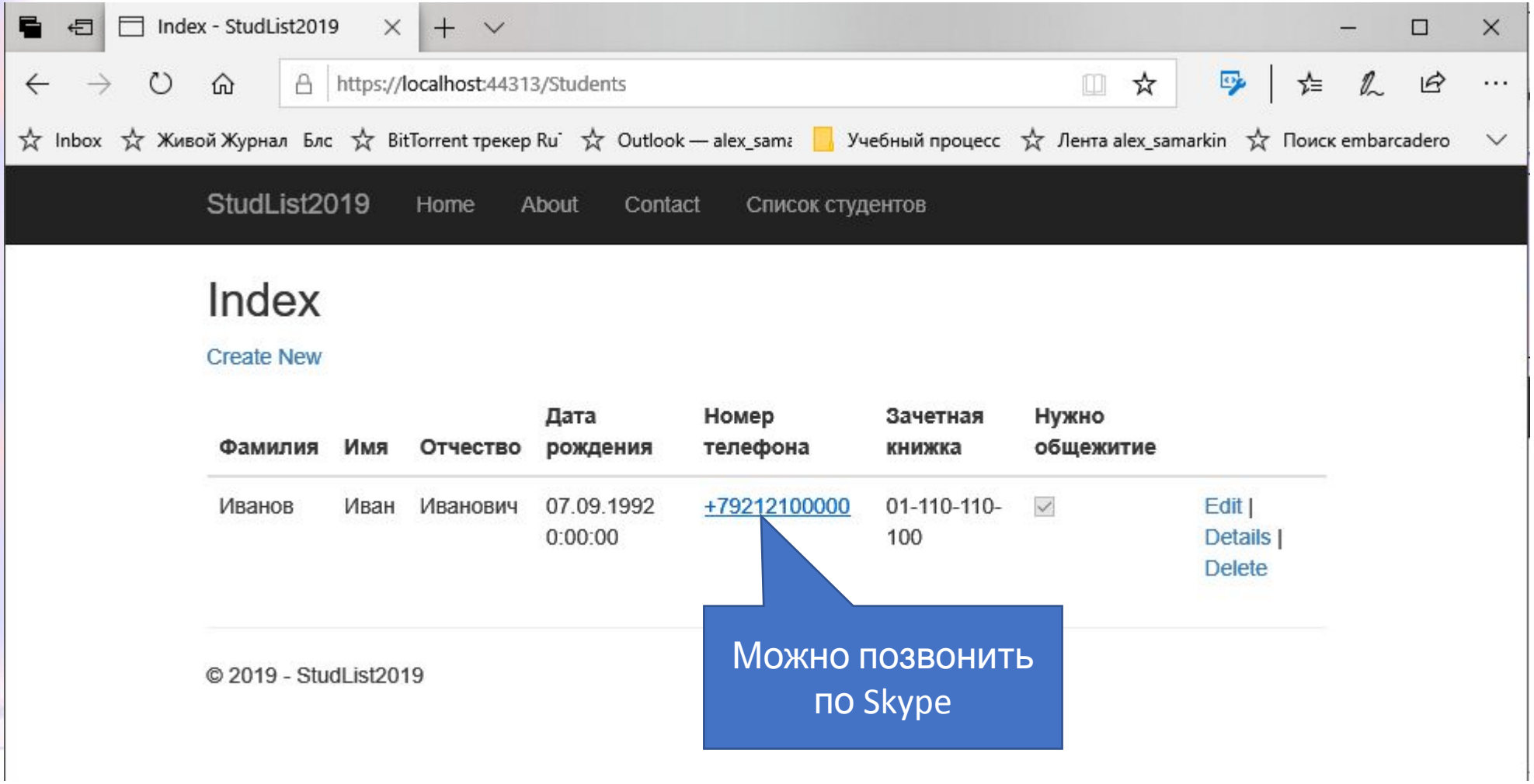


Форма ввода даннных



- Обратите внимание, что дата вводится через элемент «календарь»
- Нужно ли общежитие - чекбокс

Результат


Index - StudList2019
https://localhost:44313/Students
Inbox
Живой Журнал
Блс
BitTorrent трекер Ru
Outlook — alex_sama
Учебный процесс
Лента alex_samarkin
Поиск embarcadero
StudList2019
Home
About
Contact
Список студентов
Index
Create New
Фамилия
Имя
Отчество
Дата рождения
Номер телефона
Зачетная книжка
Нужно общежитие
Иванов
Иван
Иванович
07.09.1992 0:00:00
+79212100000
01-110-110-100
Edit | Details | Delete
© 2019 - StudList2019
Можно позвонить по Skype