**Лабораторная работа по теме № 20. Создание интранет-портала с авторизацией через Active Directory. Извлечение информации о пользователях из Active Directory.**

**Цель:** получение навыков практического применения Active Directory в веб-приложениях.

**Задачи:**

1. создание каждым студентом веб-приложения на базе шаблона ASP.NET MVC с аутентификацией по данным ОС Windows;
2. развертывание веб-приложений на веб-сервере IIS в домене;
3. демонстрация вывода имени пользователя, указанного при входе в ОС, имени домена и сведений об активных пользователях в домене.

**Теоретическое введение.**

NTLM – это протокол проверки подлинности запроса и ответа, который использует три сообщения для аутентификации клиента в среде, ориентированной на соединение и четвертое дополнительное сообщение, если требуется проверка целостности.

1. Пользователь устанавливает подключение (сетевой путь) к серверу и отправляет NEGOTIATE\_MESSAGE со своими возможностями.

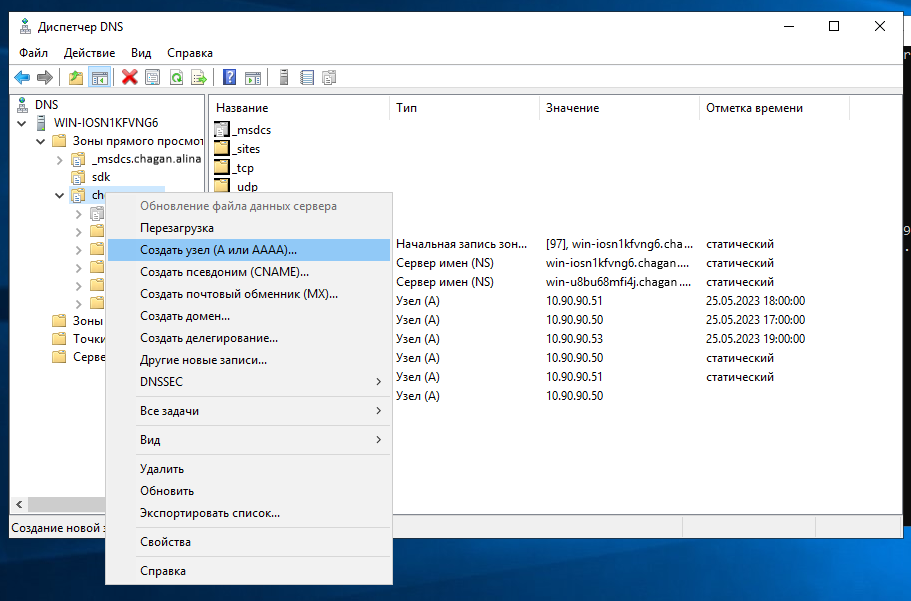
2. Сервер отвечает сообщением CHALLENGE\_MESSAGE, которое используется для идентификации (установления личности) клиента.

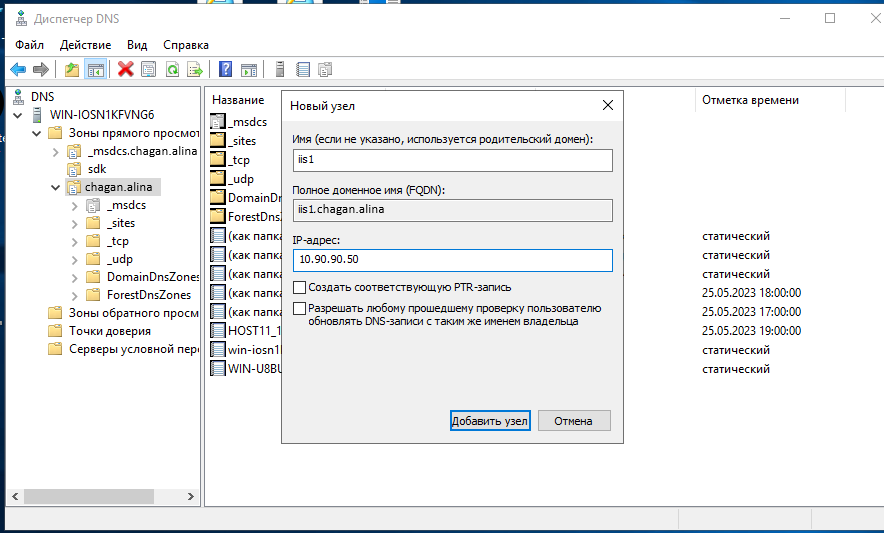
3. Клиент отвечает на сообщение при помощи AUTHENTICATE\_MESSAGE.

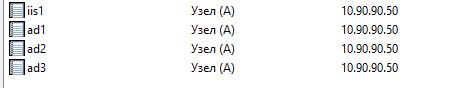
Протокол NTLM использует одно или оба значения хешированных паролей, оба из них хранятся на сервере (или контроллере домена), которые из-за отсутствия привязки эквивалентны паролю. Это означает, что хешированное значение с сервера может быть использовано для аутентификации без фактического знания пароля.

**Порядок выполнения работы.**

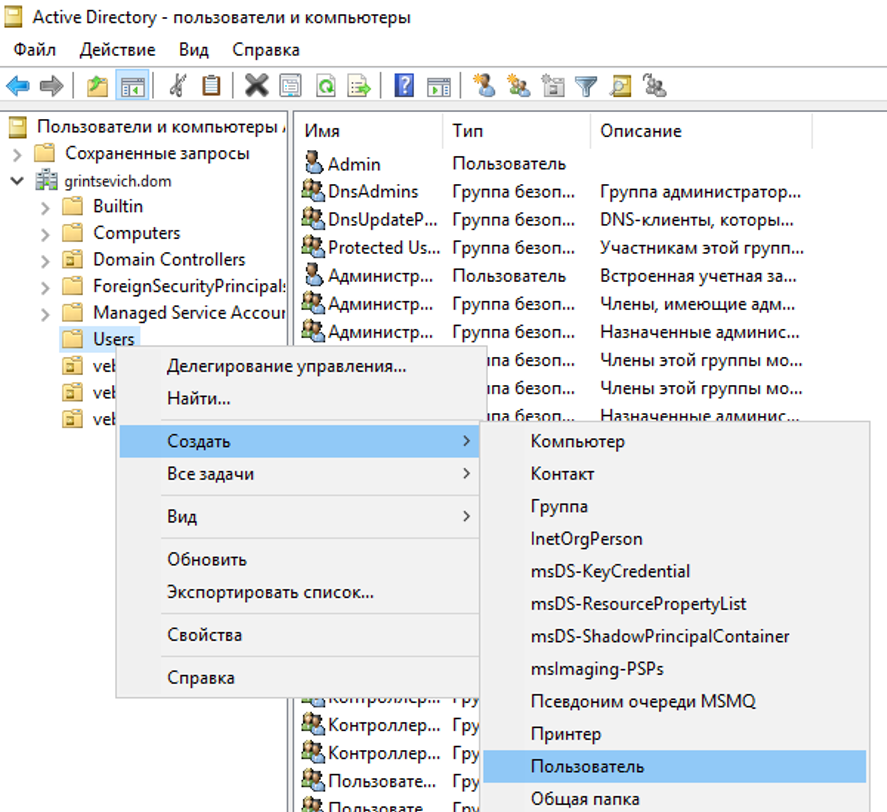
1. Лабораторная работа состоит из двух взаимосвязанных частей – работы администратора и работы разработчика.
2. Работа администратора:
   1. Создайте на DNS-сервере запись типа А для веб-узла разработчика, ссылающуюся на IP-адрес сервера администратора.

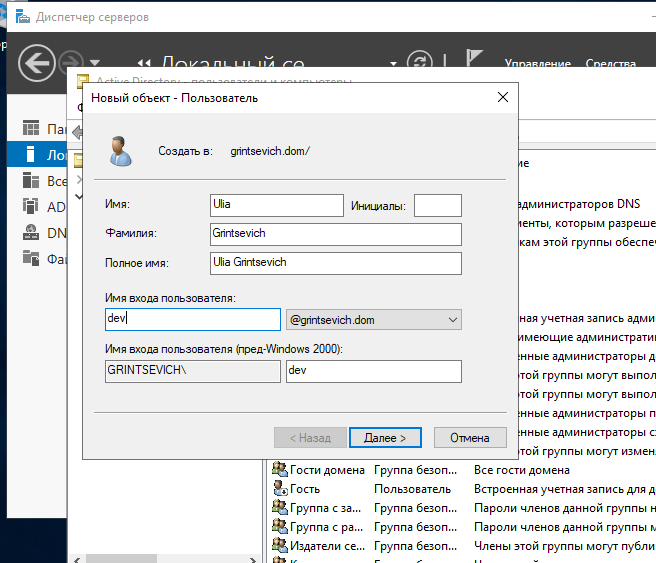


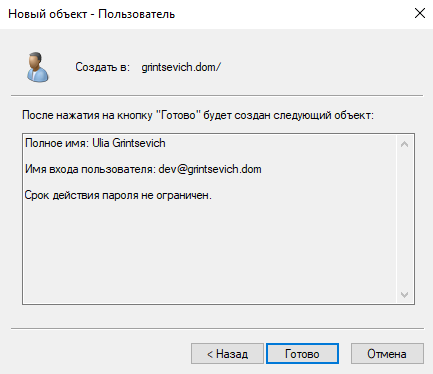
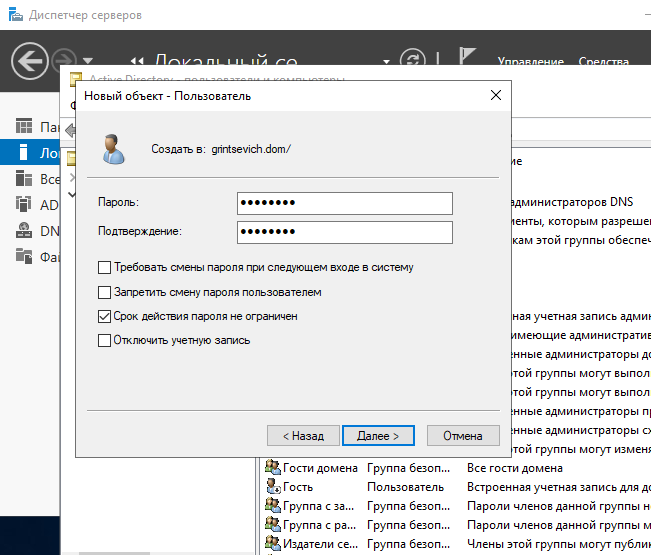




* 1. Создайте в домене отдельную учетную запись для разработчика, заполните в ней кроме обязательного логина также фамилию, имя, телефон, e-mail.





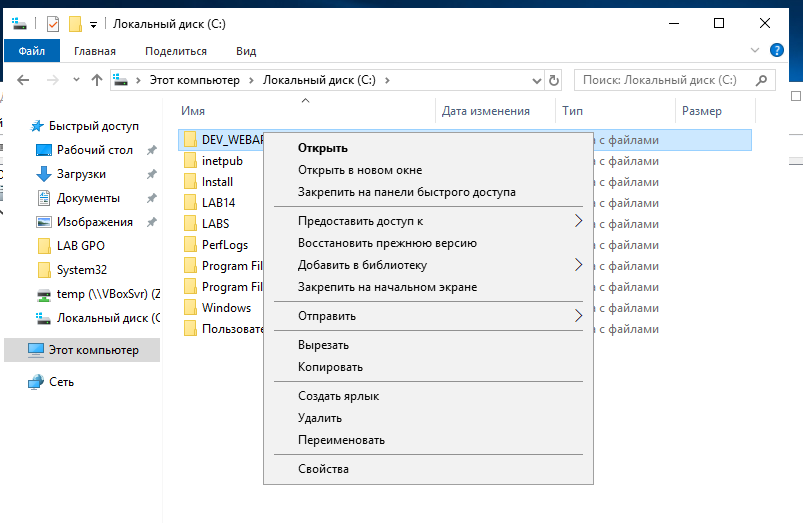


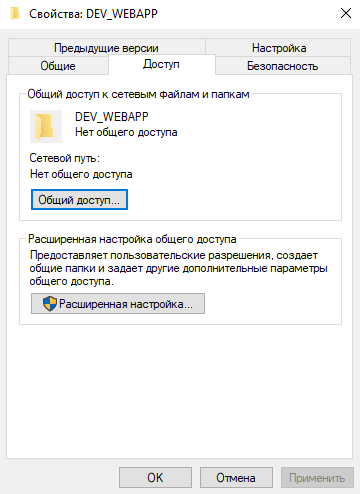
Логин GRINTSEVICH\dev

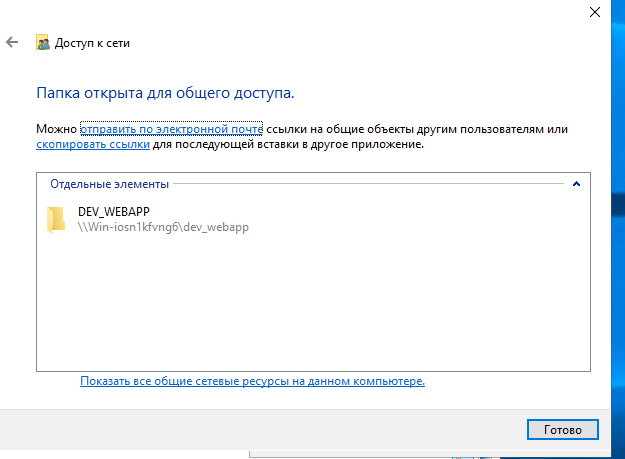
Пароль Ullia2023

ping iis1 - пингуется с сервером на 10.90.90.50

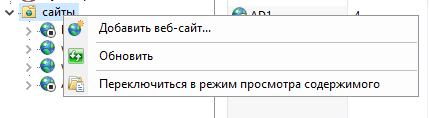
* 1. Создайте на сервере общую папку для загрузки веб-приложения разработчиком, выдайте права на полный доступ к этой папке учетной записи разработчика.

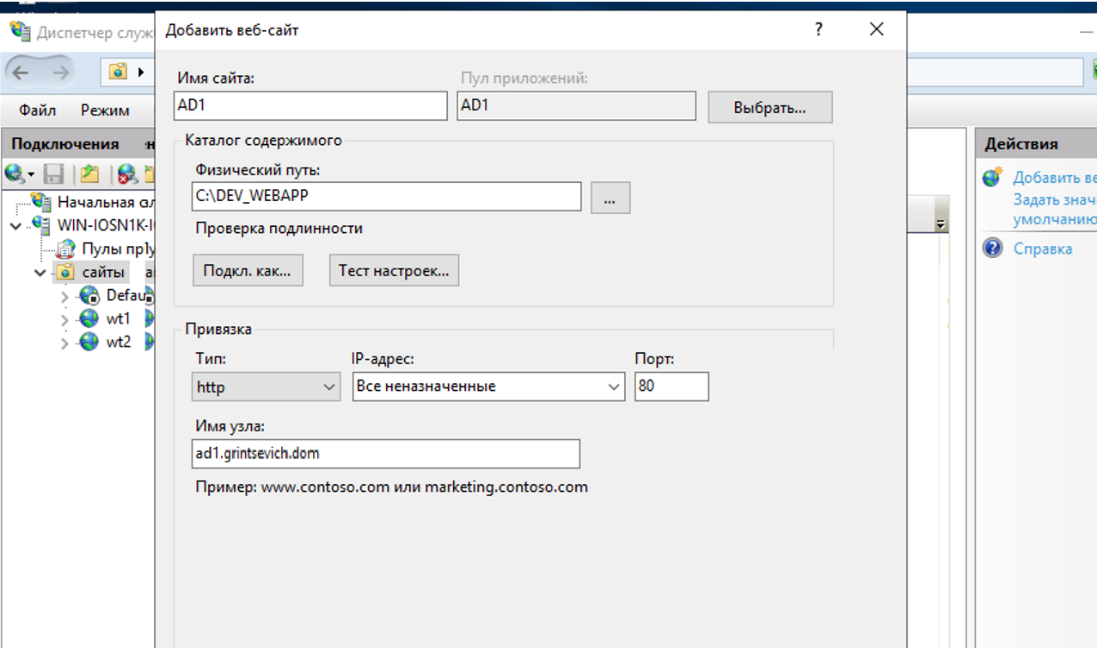


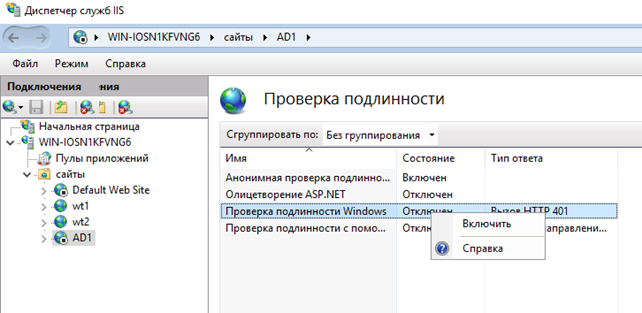


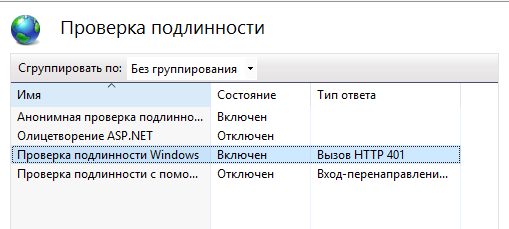


* 1. Из общей папки создайте веб-узел на 80 порту, привязанный к доменному имени.

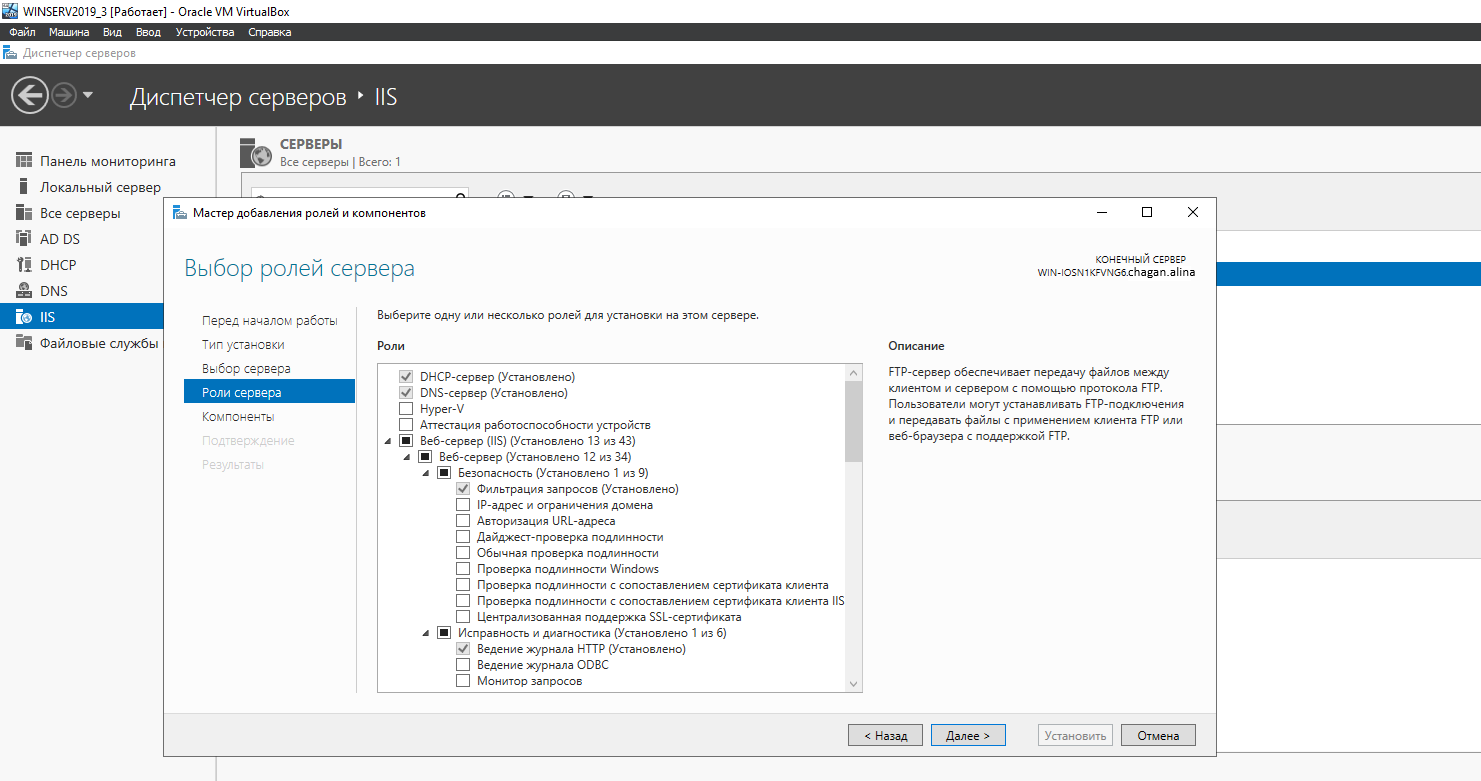


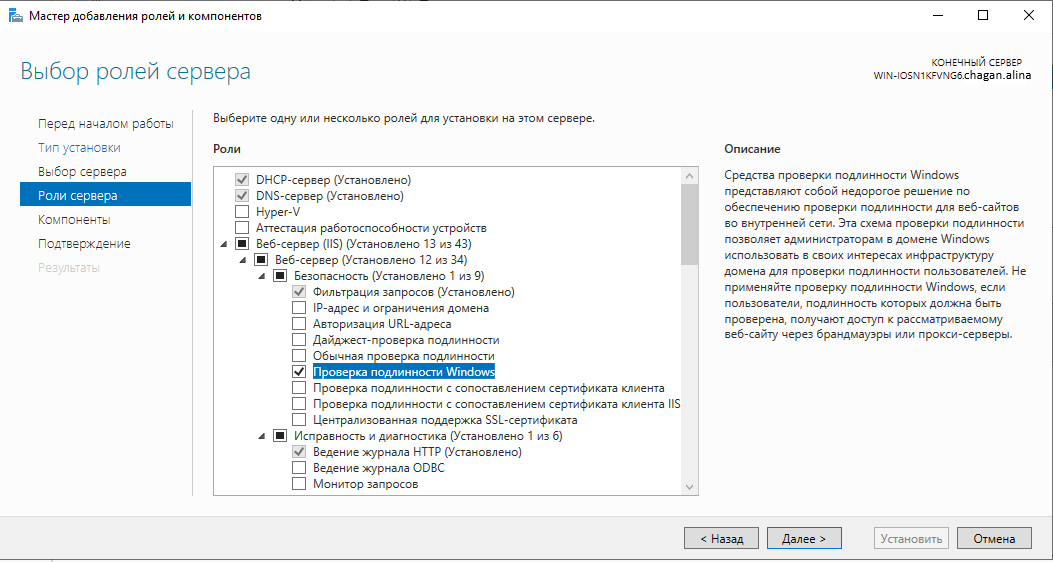


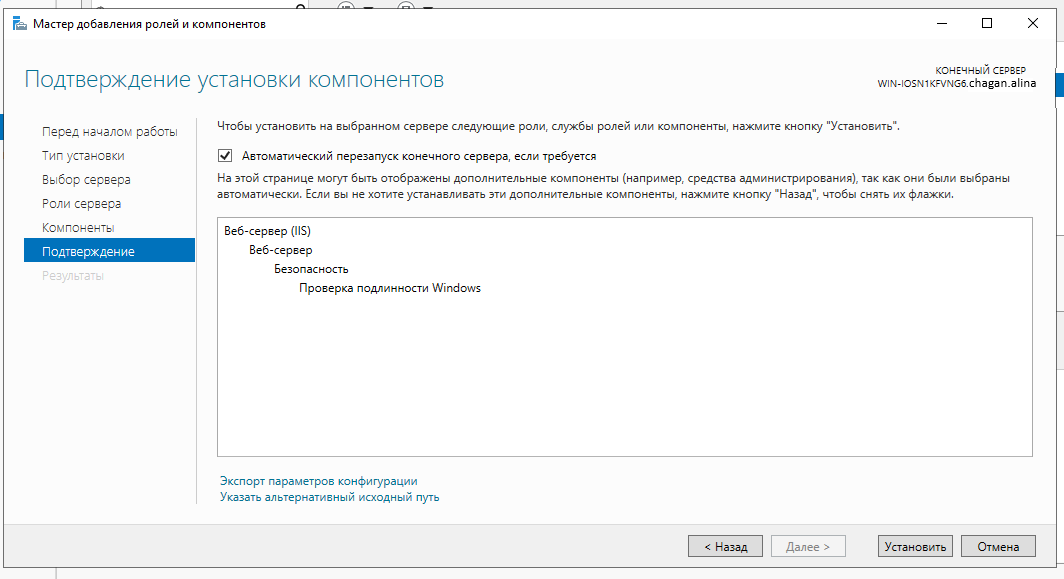




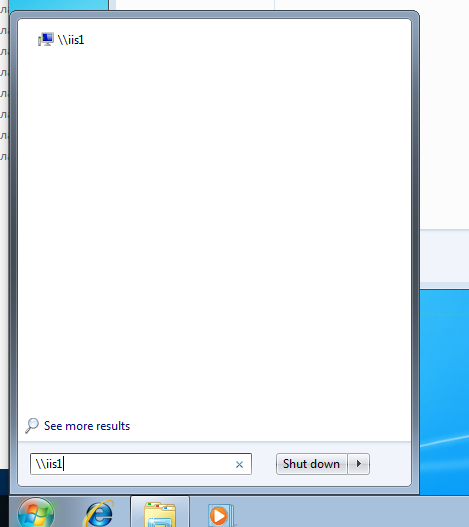
* 1. Для выполнения аутентификации в веб-приложении добавьте компонент «Аутентификация Windows» веб-сервера IIS.





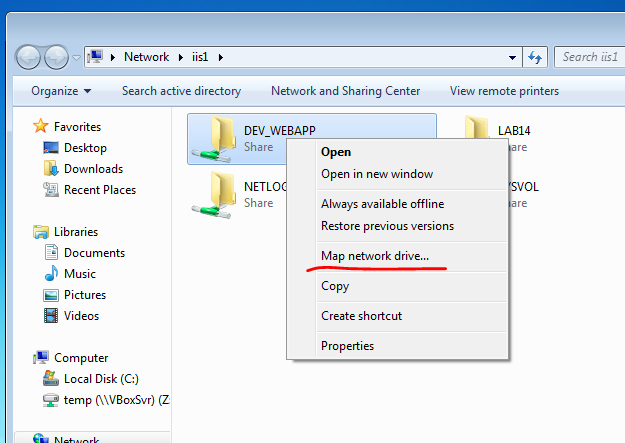


Подключение общей папки

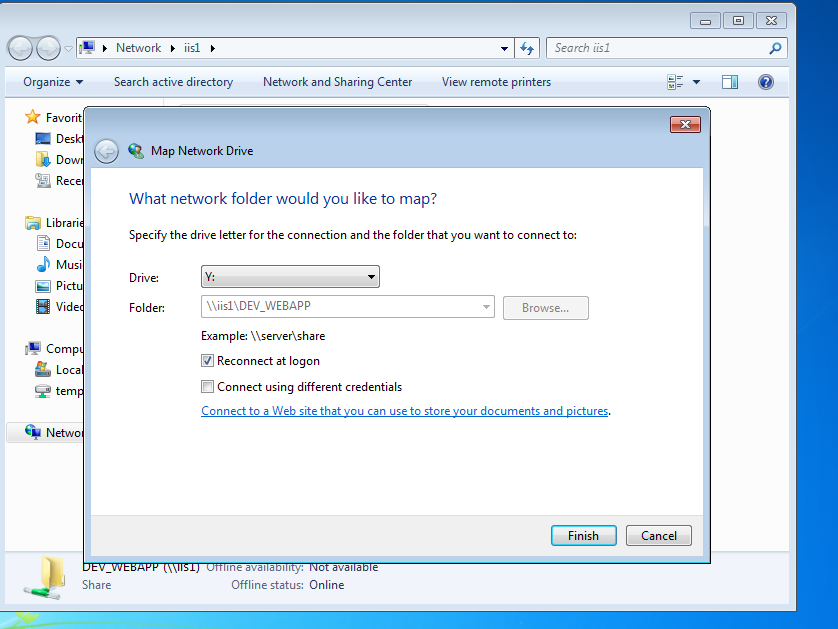


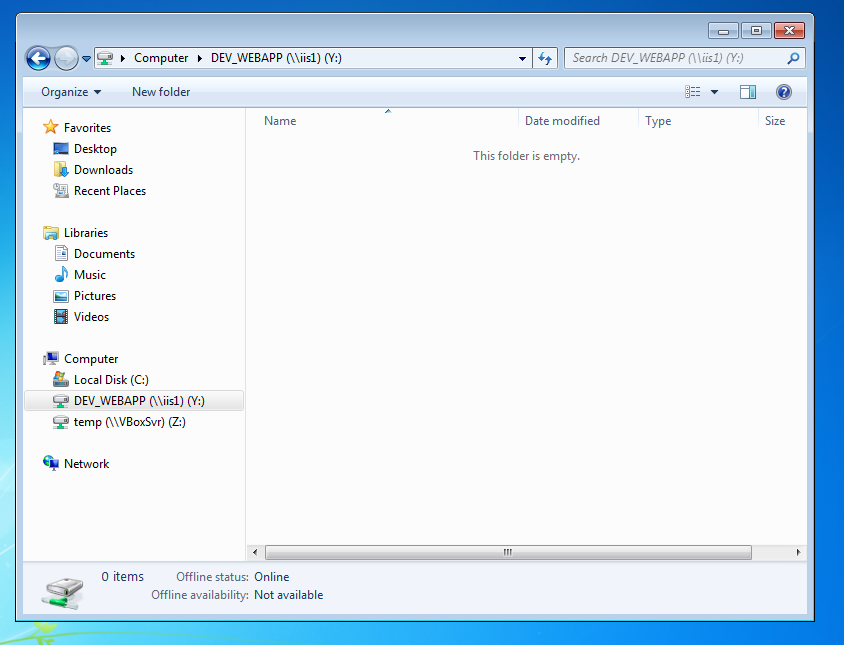
либо указать IP-адрес сервера

либо командой net use через cmd

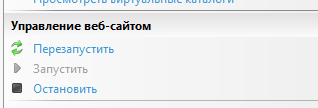


Подключаем сетевой диск

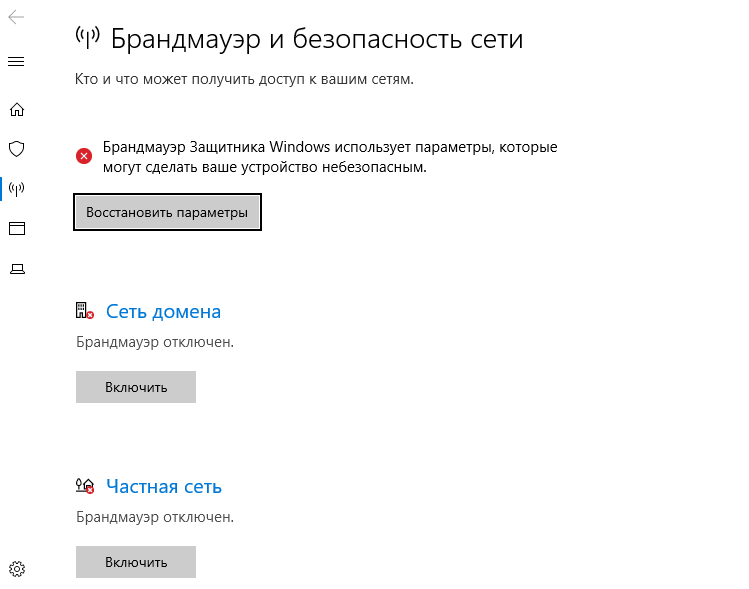




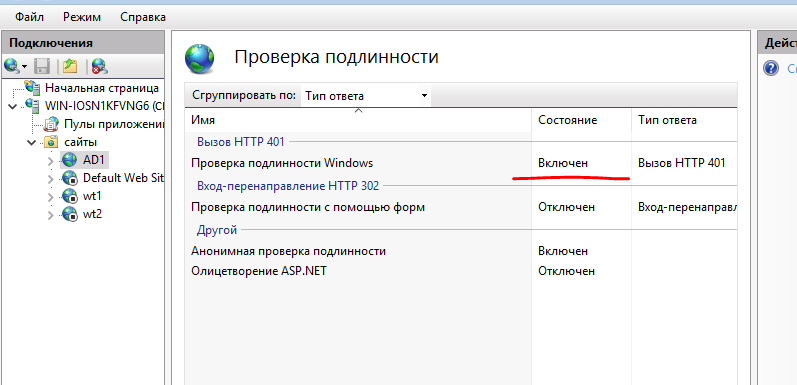
На сервере запускаем сайт в IIS



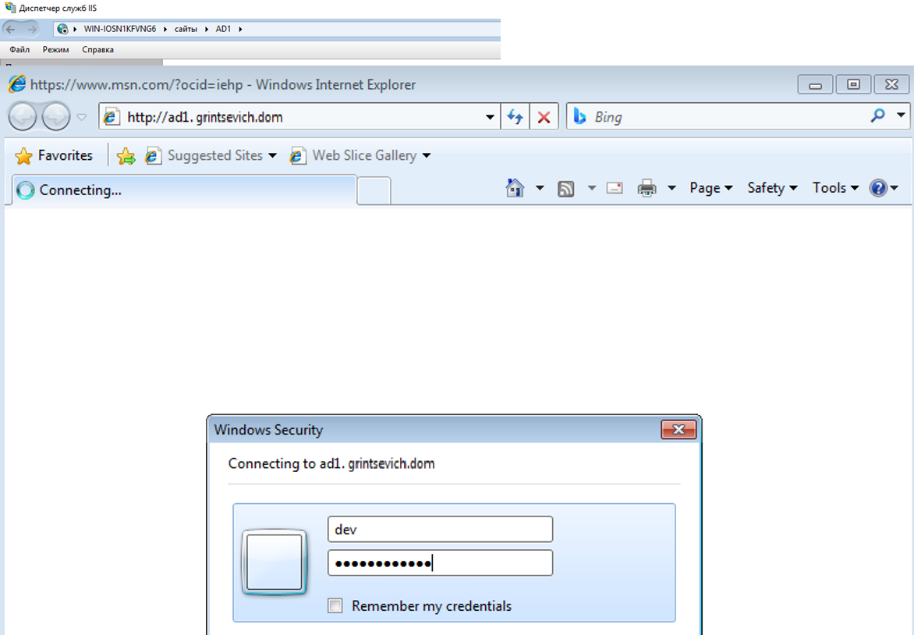
Отключаем firewall домена

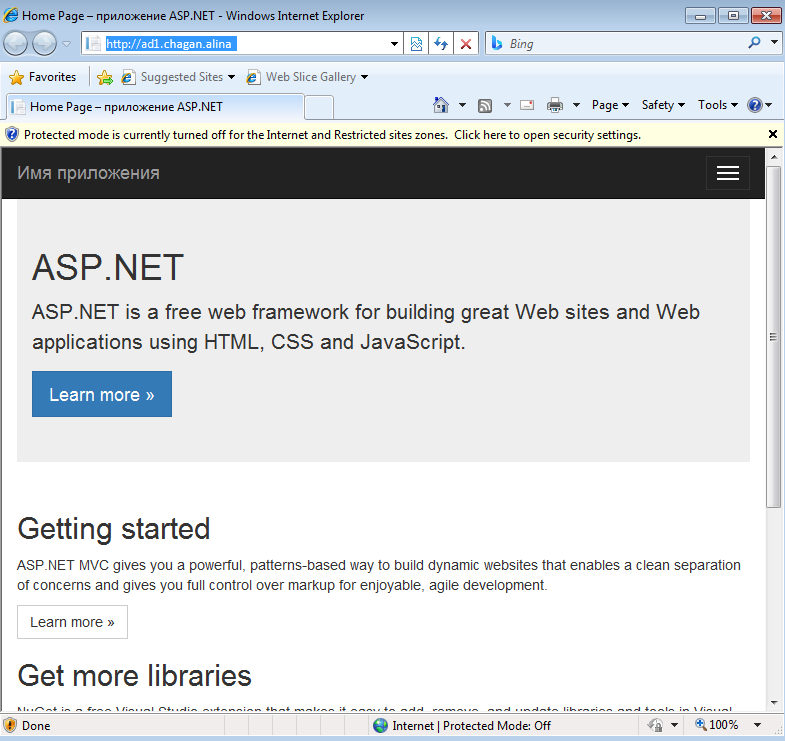


Проверяем, что включена аутентификация Windows для веб-узла



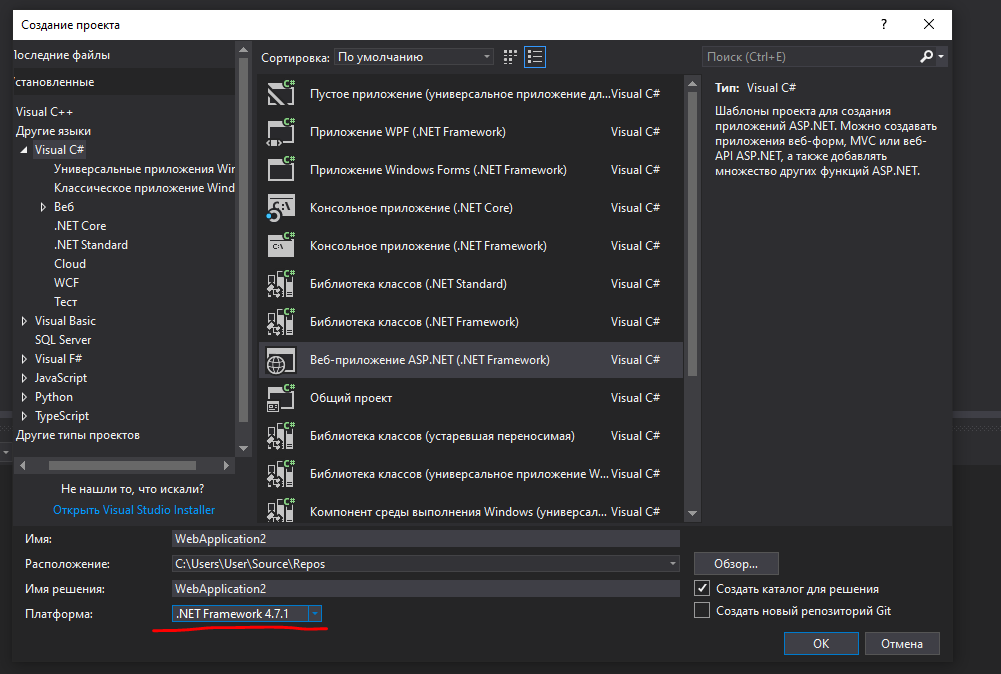
На клиенте

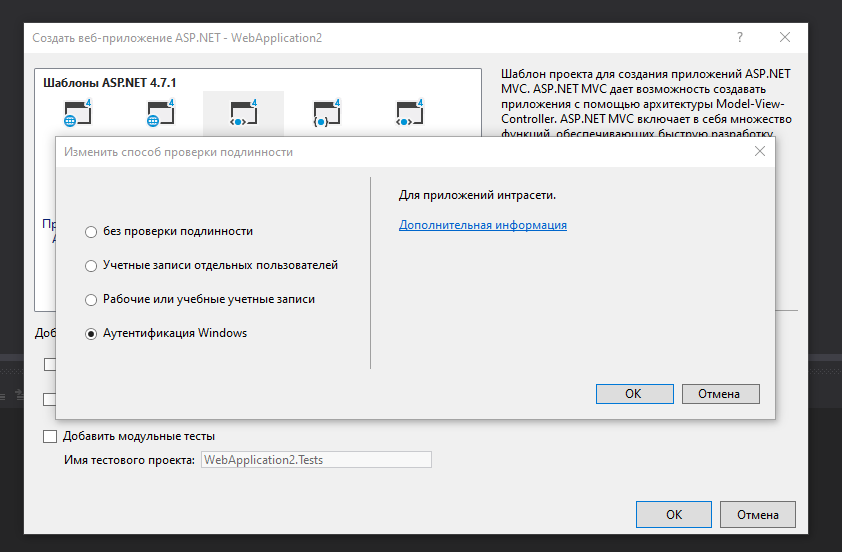




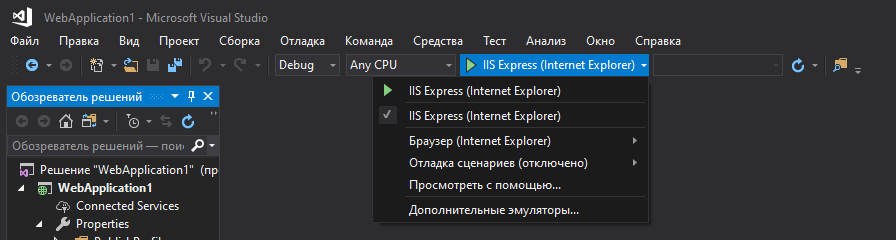
1. Работа разработчика:
   1. Создайте в Visual Studio веб-приложение на основе шаблона «Веб-приложение ASP.NET (.NET Framework)», «MVC». Измените способ проверки подлинности пользователей на «Аутентификация Windows».

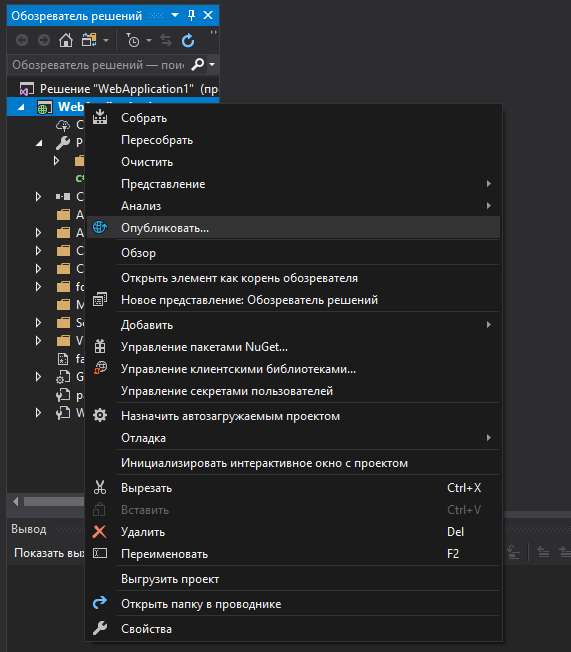
Нужно проверять совместимость .NET Framework с тем, что на сервере для IIS установлен в компонентах



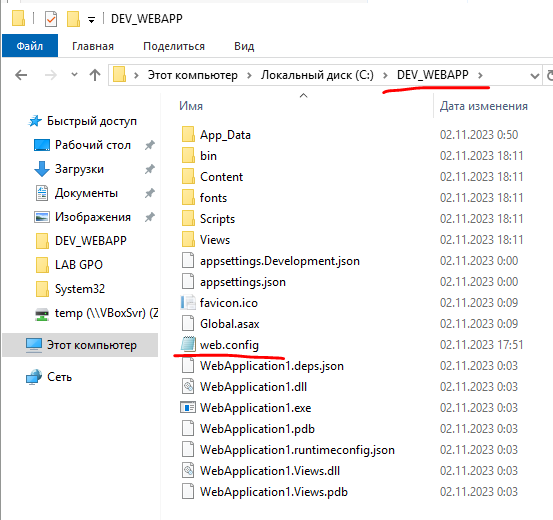


* 1. Опубликуйте приложение в отдельную папку (в окне «Обозреватель решений» нажмите правой кнопкой по проекту, выберите пункт «Опубликовать...», выберите целевой объект публикации «Папка»).



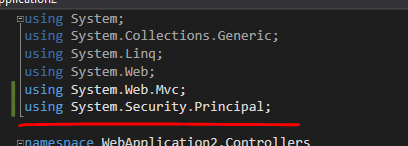


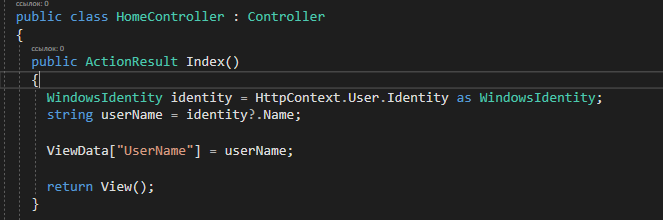
Публикуем на своем хосте, перенося в общую папку пробрасываем в виртуалку на сервер. Обязательно проверять наличие web.config файла

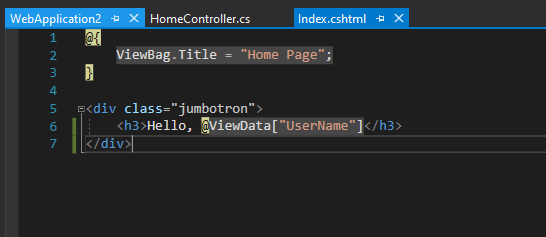


* 1. Передайте папку с опубликованным проектом на веб-сервер IIS (через сеть в общую папку на виртуальной машине-сервере или/и через общую папку VirtualBox).
  2. Войдите на введенной в домен операционной системе под Вашей учетной записью, откройте в Internet Explorer URL с доменным именем, созданным для вас администратором, убедитесь, что в правом верхнем углу отображается «Hello <домен>\<пользователь>».

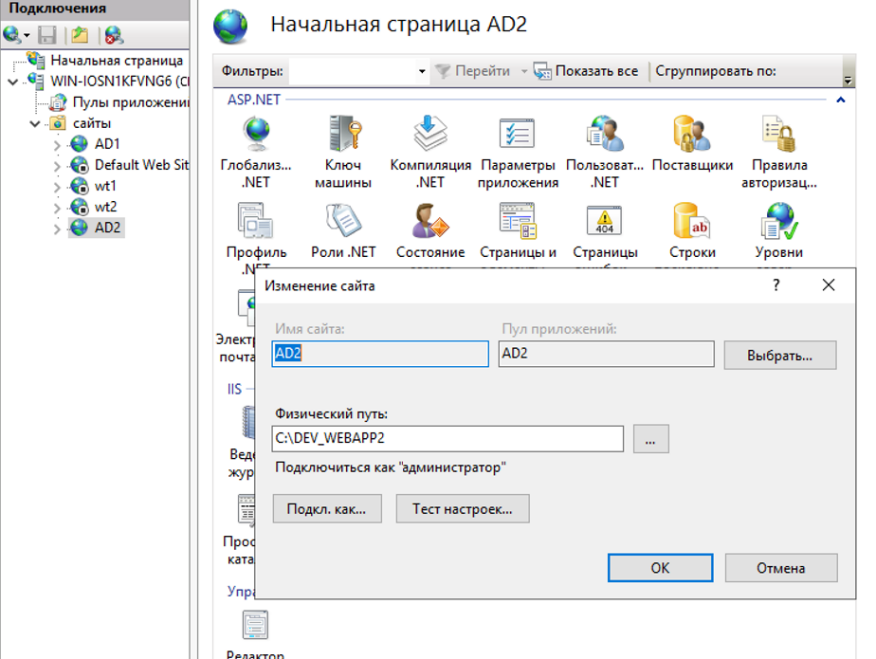
Создаем проект шаблона MVC с аутентификацией Windows, после создания проекта меняем контроллер и метод Index()



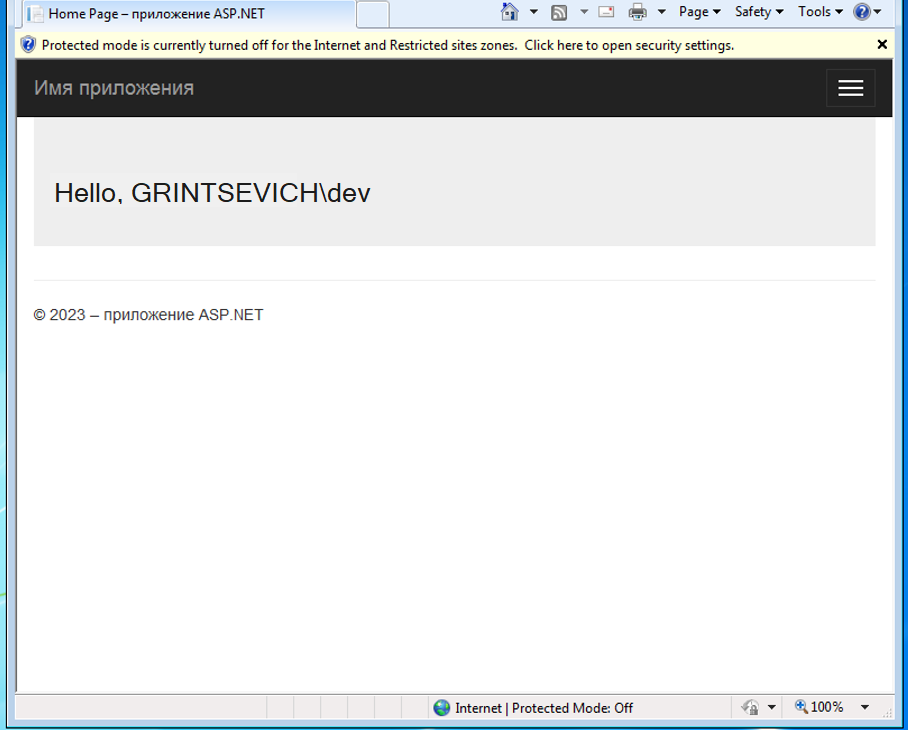




Публикуем, пробрасываем на сервер и создаем веб-узел

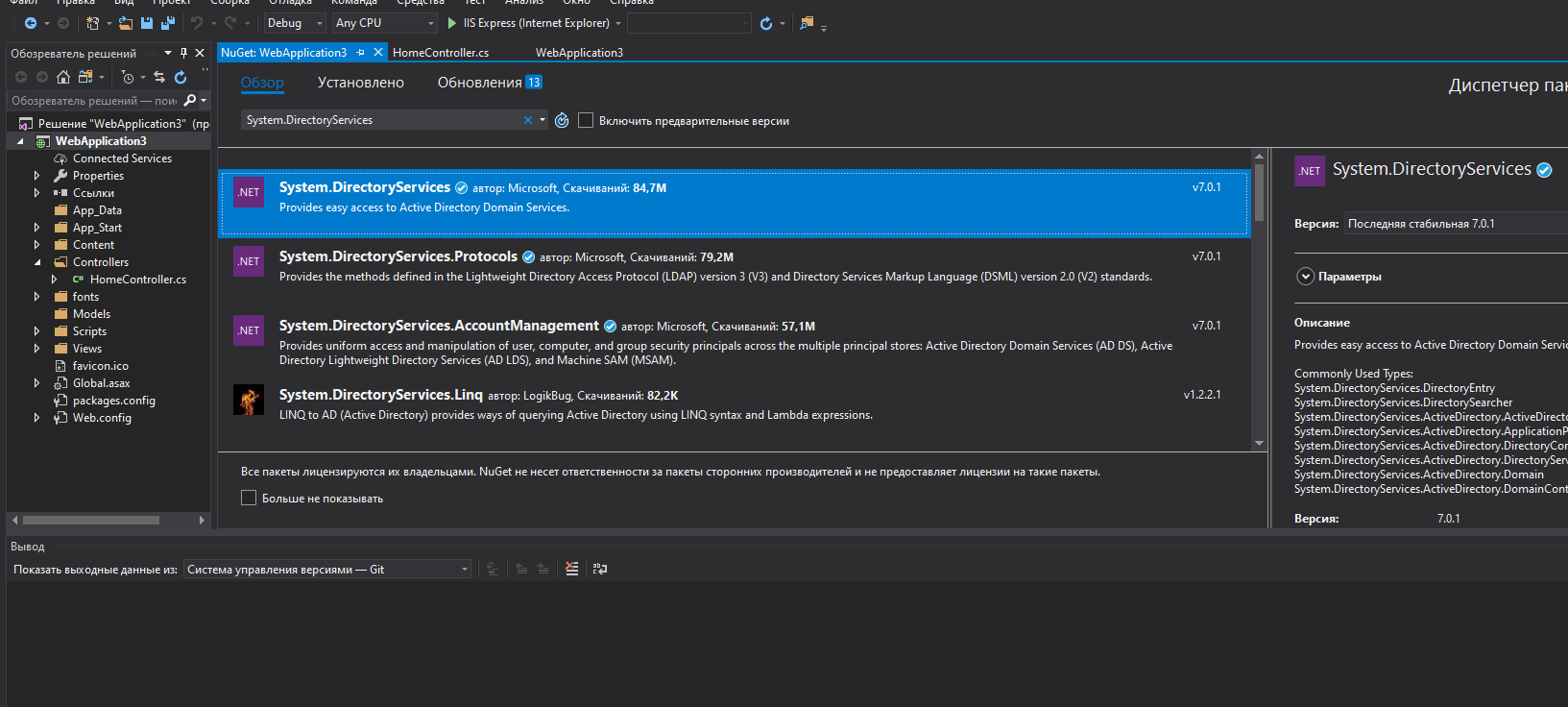


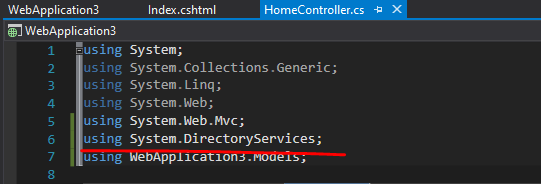
На клиенте



http://ad2.grintsevich.dom

* 1. Вернитесь в Visual Studio. Подключите в проект библиотеку «System.DirectoryServices».





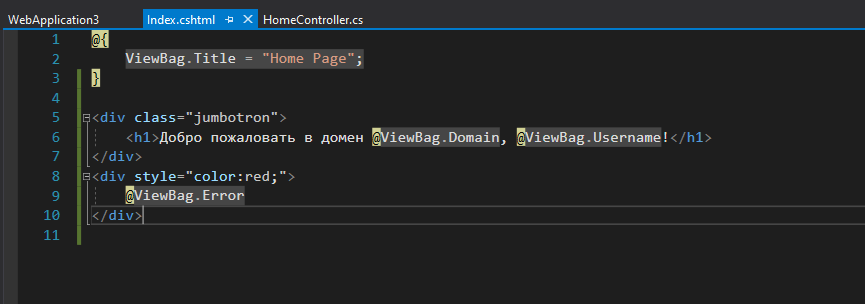
* 1. В существующем файле Controllers\HomeController.cs в методе Index, формирующем главную страницу сайта, подготовьте название домена и имя пользователя в удобочитаемом формате, без префиксов:

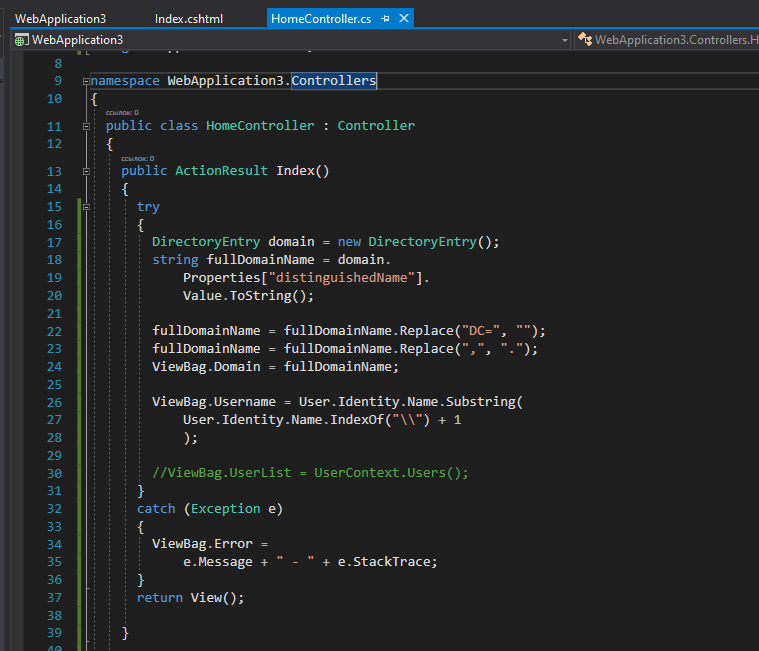
|  |
| --- |
| public ActionResult Index()  {  try  {  //получаем текущий домен  DirectoryEntry domain = new DirectoryEntry();  //получаем различающееся имя домена  string fullDomainName = domain.  Properties["distinguishedName"].  Value.ToString();  //убираем DC=,  //чтобы привести имя домена к привычной форме  fullDomainName = fullDomainName.Replace("DC=", "");  //заменяем запятые в иерархии доменов на привычные точки  fullDomainName = fullDomainName.Replace(",", ".");  //передаем имя домена в представление  ViewBag.Domain = fullDomainName;  //передаем имя пользователя в представление  ViewBag.Username = User.Identity.Name.Substring(  User.Identity.Name.IndexOf("\\") + 1  );  //получаем список пользователей и их свойств  //берем с помощью контекста  //список моделей для вывода на странице  ViewBag.UserList = UserContext.Users();  }  catch (Exception eeeee)  {  ViewBag.Error =  eeeee.Message + " - " + eeeee.StackTrace;  }  return View();  } |

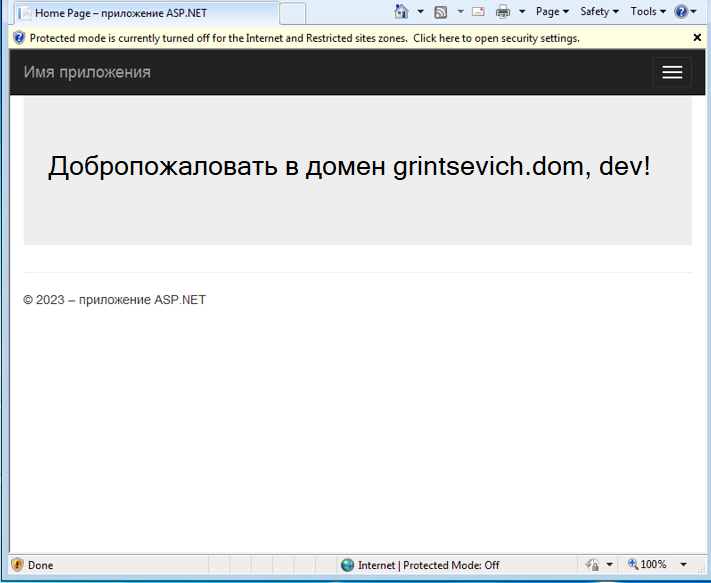
* 1. В существующем файле Views\Home\Index.cshtml подготовьте красивое приветствие для пользователей:

|  |
| --- |
| <div class="jumbotron">  <h1>Добро пожаловать в домен @ViewBag.Domain, @ViewBag.Username!</h1>  </div>  <div style="color:red;">  @ViewBag.Error  </div> |

* 1. Опубликуйте новую версию проекта в папку и передайте ее на сервер IIS вместо старой. Снова зайдите на веб-узел с операционной системы, добавленной в домен, для тестирования отображения приветствия в браузере.







* 1. Для вывода списка пользователей попробуем следовать паттерну Model-View-Controller. Для начала создадим модель – класс для отображения в программе объектов, хранимых в БД (в нашем случае – Active Directory) – в новом файле Models\UserModel.cs:

|  |
| --- |
| public class UserModel  {  //фамилия и имя  public string Username { get; set; }  //логин  public string Login { get; set; }  //телефон  public string Phone { get; set; }  //почта  public string Email { get; set; }  //все свойства  public string FullDesc { get; set; }  } |

* 1. Для заполнения списков моделей данными из БД создадим в новом файле Models\UserContext.cs класс-контекст с первым отладочным методом, выбирающем все объекты и все их свойства:

|  |
| --- |
| public class UserContext  {  public static List<UserModel> UsersDebug()  {  //получаем текущий домен  DirectoryEntry domain = new DirectoryEntry();  List<UserModel> UserList = new List<UserModel>();  using (DirectorySearcher ds = new DirectorySearcher(domain))  {  using (SearchResultCollection results = ds.FindAll())  {  if (results != null && results.Count > 0)  {  for (int i = 0; i < results.Count; i++)  {  //свойства текущего пользователя  ResultPropertyCollection res = results[i].Properties;  //заполняемая модель  UserModel u = new UserModel();  //перечень всех имеющихся свойств  u.FullDesc = "";  foreach (string e in res.PropertyNames)  {  u.FullDesc += e.ToString() + " = " +  res[e.ToString()][0].ToString() +  "; ----- ";  }  UserList.Add(u);  }  }  }  }  domain.Close();  return UserList;  }  } |

* 1. Для создания новой отладочной страницы с выводом всех свойств всех объектов создадим новый метод FullDesc в существующем файле Controllers\HomeController.cs:

|  |
| --- |
| public ActionResult FullDesc()  {  //получаем список пользователей и их свойств  //берем с помощью контекста  //список моделей для вывода на странице  ViewBag.UserList = UserContext.UsersDebug();  return View();  } |

* 1. Для отображения новой страницы создадим для нее шаблон-представление Views\Home\FullDesc.cshtml:

|  |
| --- |
| @{  ViewBag.Title = "Debug";  }  <div>  <table class="table table-bordered">  <tr>  <th>  Полная информация о пользователях  </th>  </tr>  @foreach (var u in ViewBag.UserList)  {  <tr>  <td>  @u.FullDesc  <hr />  </td>  </tr>  }  </table>  <div style="color:red;">  @ViewBag.Error  </div>  </div> |

* 1. Опубликуйте новую версию приложения на веб-сервере IIS, откройте отладочную страницу, найдите учетные записи пользователь, определите название и значение свойства, отвечающего за категорию объектов «пользователь» (нужно для создания фильтра при выборке объектов из БД в следующем пункте).
  2. Создайте новый метод в классе-контексте для получения списка заполненных моделей для объектов категории «пользователь»:

|  |
| --- |
| public static List<UserModel> Users()  {  //получаем текущий домен  DirectoryEntry domain = new DirectoryEntry();  List<UserModel> UserList = new List<UserModel>();  using (DirectorySearcher ds = new DirectorySearcher(domain))  {  **ds.Filter = "(<название свойства>=<значение свойства после CN=>)";**  using (SearchResultCollection results = ds.FindAll())  {  if (results != null && results.Count > 0)  {  for (int i = 0; i < results.Count; i++)  {  //свойства текущего пользователя  //значение каждого свойства - тоже коллекция,  //но обычно в ней только нулевой элемент,  //многие свойства - необязательные, поэтому  //перед обращением к ним нужно проверять их наличие  ResultPropertyCollection res = results[i].Properties;  //заполняемая модель  UserModel u = new UserModel();  u.Username = "";  //фамилия  if(res["sn"] != null && res["sn"].Count > 0)  u.Username += res["sn"][0].ToString() + " ";  //имя  if (res["givenname"] != null && res["givenname"].Count > 0)  u.Username += res["givenname"][0].ToString();  //логин  if (res["name"] != null && res["name"].Count > 0)  u.Login = res["name"][0].ToString();  //телефон  if (res["telephonenumber"] != null &&  res["telephonenumber"].Count > 0)  u.Phone = res["telephonenumber"][0].ToString();  //почта  if (res["mail"] != null && res["mail"].Count > 0)  u.Email = res["mail"][0].ToString();  //перечень всех имеющихся свойств  u.FullDesc = "";  foreach (string e in res.PropertyNames)  {  u.FullDesc += e.ToString() + " = " +  res[e.ToString()][0].ToString() +  "; ----- ";  }  UserList.Add(u);  }  }  }  }  domain.Close();  return UserList;  } |

* 1. Добавьте в класс HomeController в метод Index в конец блока try получение заполненного списка моделей пользователей и его передачу в представление:

|  |
| --- |
| ViewBag.UserList = UserContext.Users(); |

* 1. Добавьте вывод списка пользователей в виде таблицы в представление главной страницы:

|  |
| --- |
| <div>  <table class="table table-bordered">  <tr>  <th width="25%">  ФИО  </th>  <th width="25%">  Логин  </th>  <th width="25%">  Телефон  </th>  <th width="25%">  E-mail  </th>  </tr>  @foreach (var u in ViewBag.UserList)  {  <tr>  <td>  @u.Username  </td>  <td>  @u.Login  </td>  <td>  @u.Phone  </td>  <td>  @u.Email  </td>  </tr>  }  </table>  </div> |



* 1. Опубликуйте новую версию приложения на веб-сервере IIS, убедитесь в успешном выводе списка пользователей. Все ли пользователи в списке Вам знакомы?

