

ASP .NET

ASP. net “Active Server Pages”:

Microsoftek egindako Web aplikazioen framework-a CLR (Common Language Runtime) erabiltzen duena web orri dinamikoak sortzeko ondorengo programazio lengoai bat erabiliz: C#, VB.NET etab. Bi eredu ditu: Web Forms eta ASP.NET MVC.

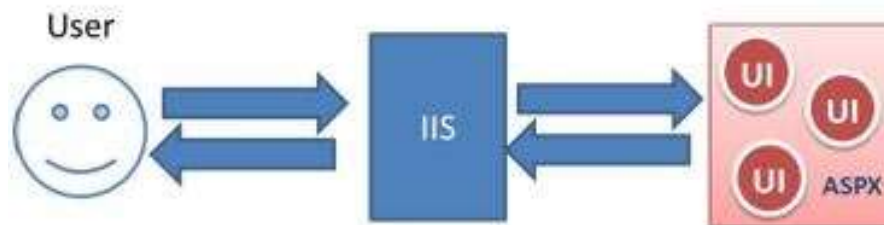


ASP .NET

- **Zer da Web Forms?**

Microsoftek lehenengo sortu zuen eredua ASP.NET Web Forms izan zen arazo batzuei aurre egiteko eta ebentoei zuzendutako programazioa web mundura eramateko:

- HTTP egoera gabeko protokoloa da. ASP .Net Web Forms maila altuko abstrakzioa lortzen du egoera gabeko web aplikazioentzat eta egoera duen eredua sortu web programatzaileentzat, VIEWSTATE erabiliz.
- WinForms bezala lan egiteko aukera, kontrol konplexuak erabilita
- Code-behind kontzeptua erabiltzen du. Kontrolen ebentoak harrapatu eta kodigoa jarri.



ASP .NET

- **Zer da MVC?**

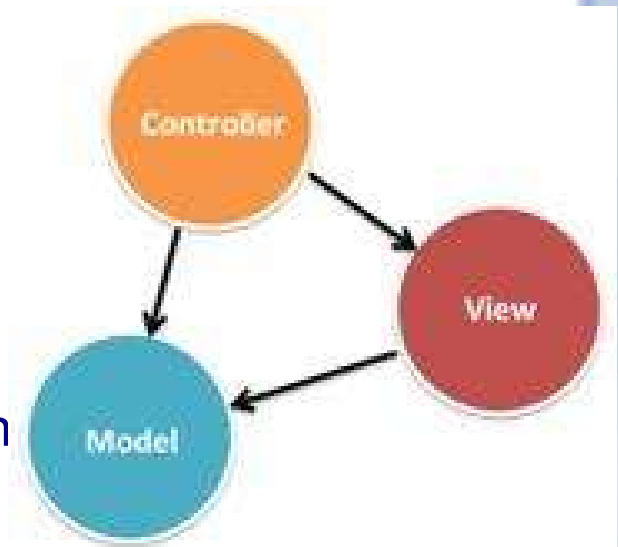
MVC aspalditik ezagutzen den arkitektura-eredua da. Ez da Microsoftena. Arkitektura-eredu bat arazo zehatz bat konpontzen duen zerbait da: gure sistemak nola banatu esaten diguna eta zeintzu modulo eta klase sortu behar ditugun.

MVC ereduak osagaiak banatzen ditu eta horrela, interfaze osagaiak ez du ezagutu behar aplikazioaren logika zein den edo erabiltzen diren datuak nolakoa diren. Era berean, beste osagaiek berdin funtzionatzeko dute. Sistema honetan hiru zati bereiztu behar dira: M (Model), V (View) eta C (Controller).

Model parte azkarra bezala hartzen da, dakielako zein diren aplikazioaren arauak, logika eta datuak eta ez du besteei menpekotasunik.

Controller eskaerak hartzen eta bidaltzen ditu. Erabiltzailearen interakzioa kontrolatzen du eta beste osagai biek harreman estua du.

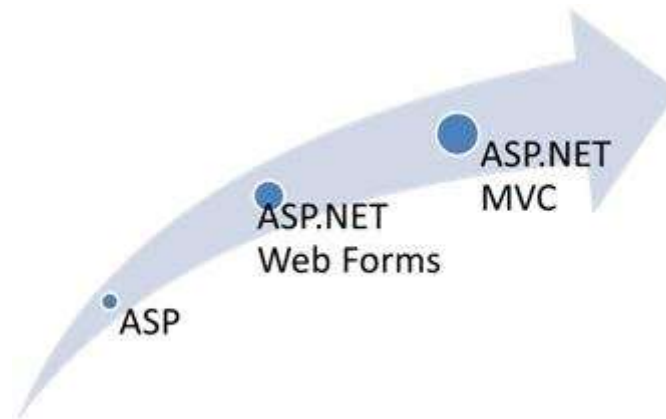
View, datuen irudikapena besterik ez da, eta ez du ezer kontrolatzen.



ASP .NET

- **Zer da ASP .NET MVC?**

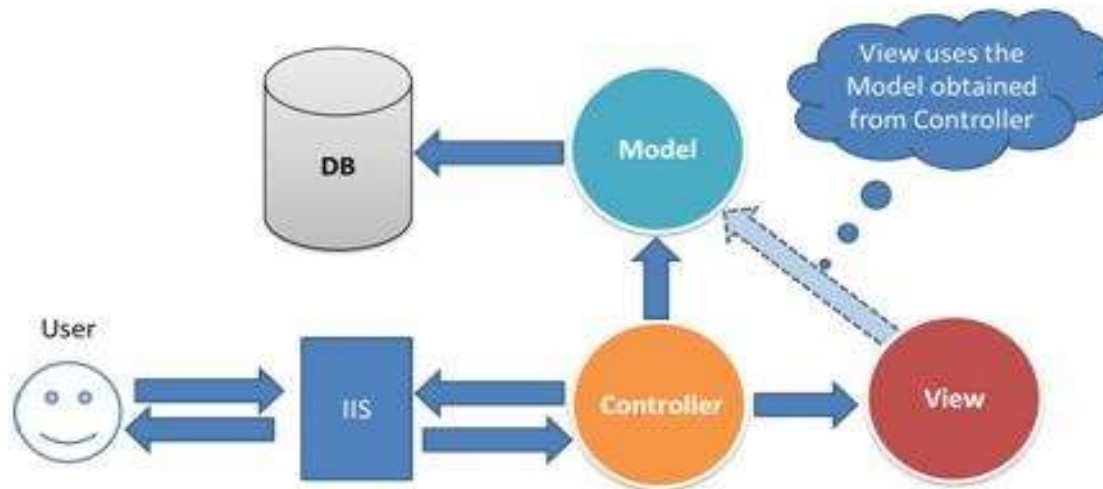
ASP.NET MVC Microsofteko Web aplikazioen frameworka da, MVC daukan abantaila guztiak aprobetxatzeko:



- Egoera gabeko web orriak programatzen dira,
- Zati bakoitza banaturik dago: model, view eta controller
- Ez du kontrol konplexurik erabiltzen, web kontrol arruntak baizik
- JQUERY, AJAX, ANGULAR.....teknologiak erabiltzeko aproposa
- Aplikazio berean talde lana egiteko aproposa

ASP .NET

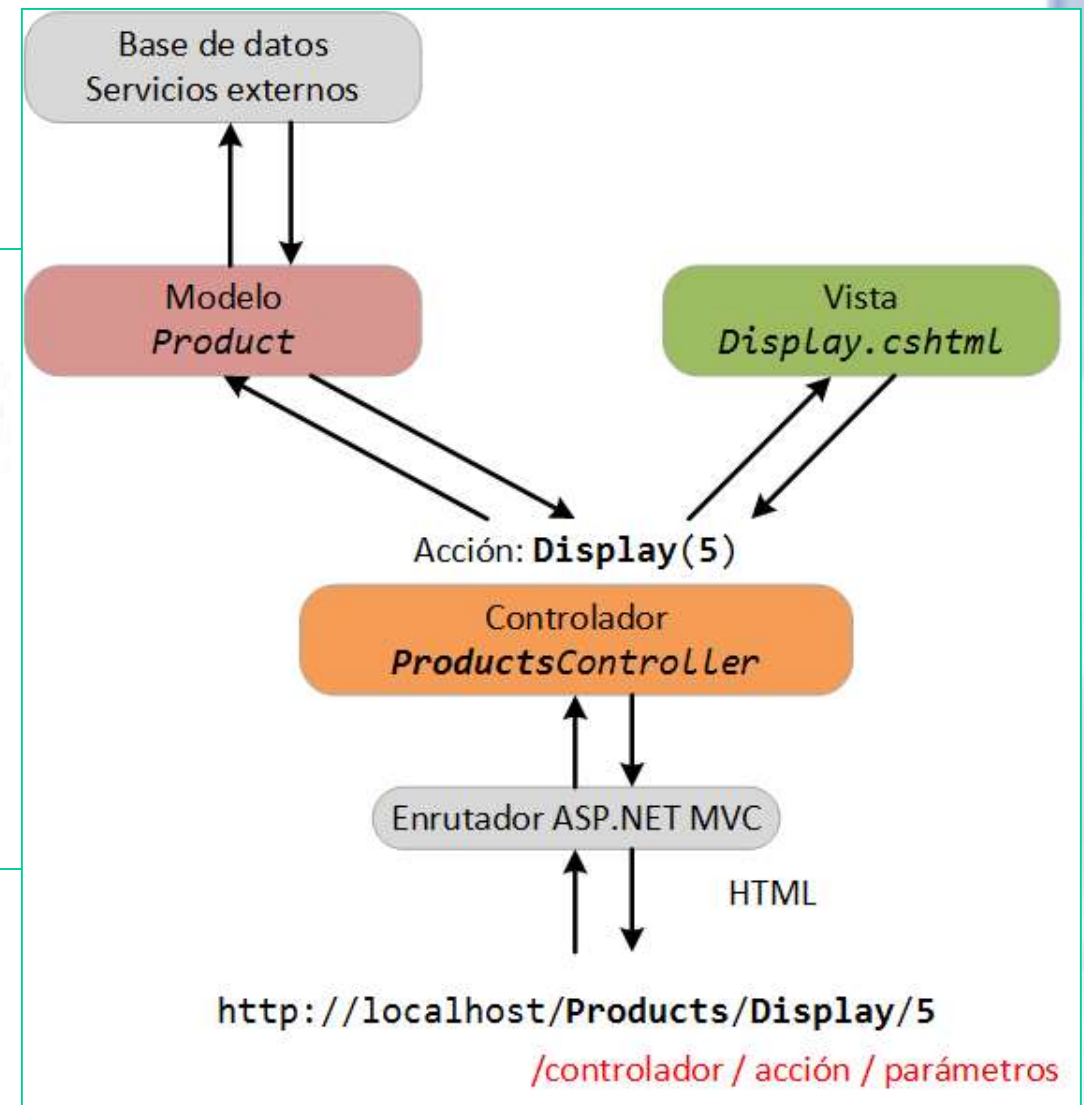
Nola egiten du lan ASP .NET MVC?



1. Erabiltzaileak eskaera egiten du (URL nabegadorean).
2. Kontroladoreak(C) eskaera hori kontrolatzen du (URL nola idazten den arabera)
3. Kontroladorea datubaseko datuak hartzen ditu (horretarako zerbitzuak erabiliz) eta datuen eredua erabiltzen du (M)
4. Kontroladoreak behar den bista(V) aukeratzen du (html, js, css..... kodigoa hemen) eta datuak pasatzen dizkio
5. Kontroladoreak erabiltzaileari bista bidaltzen dio, datuak barne.

ASP .NET

Nola egiten du lan ASP .NET MVC?



ASP .NET- MVC

1. adibidea:

- Web aplikazioa sortu «Aplicación web de ASP .NET Core (Modelo-Vista-Controlador)» txantiloia erabiliz, eta izena: "Adibidea"

Crear un proyecto

asp x Borrar

Todos los lenguajes ▼ Todas las plataformas ▼ Todos los tipos de proye

C# Linux macOS Windows Nube Servicio Web

Plantillas de proyecto recientes

- Setup Wizard
- Aplicación de Windows Forms (.NET Framework) C#
- Biblioteca de controles de Windows Forms (.NET Framework) C#

Aplicación web de ASP.NET Core (Modelo-Vista-Controlador)

Una plantilla de proyecto para crear una aplicación ASP.NET Core con controladores y vistas de ASP.NET Core MVC de ejemplo. Esta plantilla también puede usarse para servicios RESTful HTTP.

C# Linux macOS Windows Nube Servicio Web

Información adicional

Aplicación web de ASP.NET Core (M

Framework i

.NET 6.0 (Compatibilidad a largo plazo)

Authentication de campo i

Ninguno

☒ Configurar para HTTPS i

☐ Habilitar Docker i

Sistema operativo de Docker i

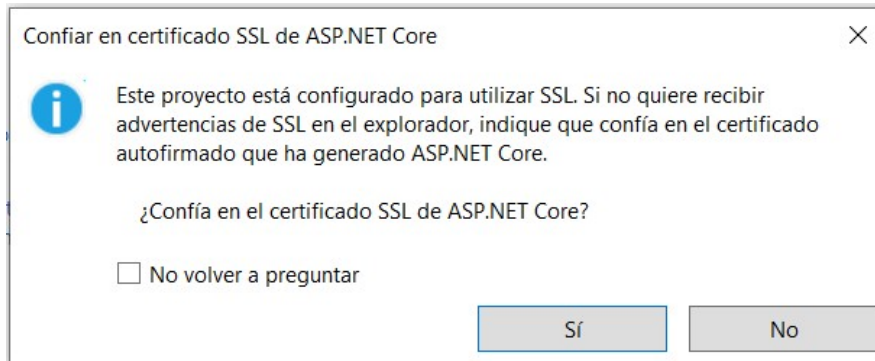
Linux

☐ Do not use top-level statements i

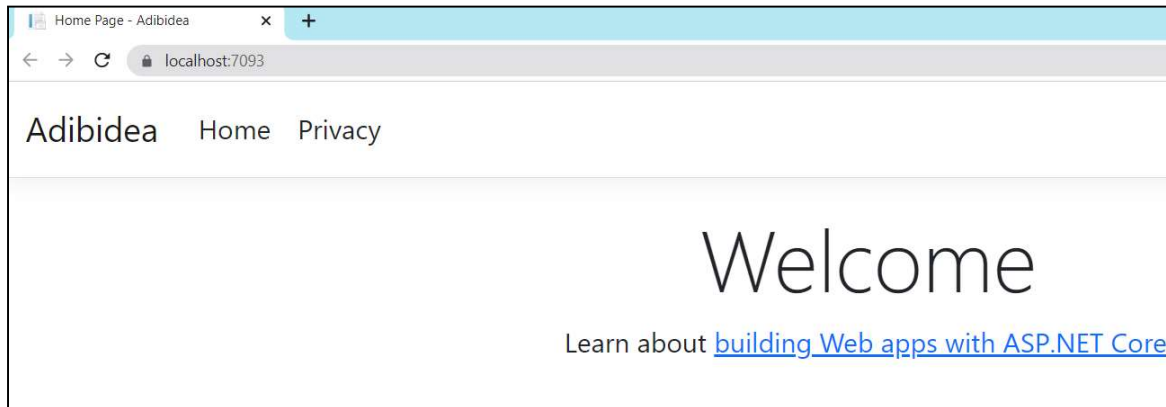
ASP .NET- MVC

1. adibidea:

- Web aplikazioa exekutatu. SSL ziurtagirian konfidantza daukagula galdetzen badu, baietz esango diogu.



- Web orria bistaratuko du:



ASP .NET- MVC

1. adibidea:

- Lehenengo **controller** bat gehituko dugu **Controllers** karpetan: **Agregar – Controlador - Controlador de MVC: en blanco (KaixoMunduaController)**
- Kodigo hau aldatu kontroladorean:

```
public string Index()  
{  
    return "Hau da nire hasierako ekintza...";  
}  
public string OngiEtorri()  
{  
    return "Hau da Ongi Etorri metodoa...";  
}
```

URL-a nola interpretatzen den:
/[Controller]/[ActionName]/[Parameters]

Defektuz



ASP .NET- MVC

1. adibidea:

- Parametroak pasatuko ditugu. Metodoa aldatu behar da:

```
public string OngiEtorri(string izena, int zenbat = 1)
{
    return HttpUtility.HtmlEncode("Kaixo " + izena + ", zenbat da: " + zenbat);
}
```



Kaixo izaskun, zenbat da: 4

ASP .NET- MVC

1. adibidea:

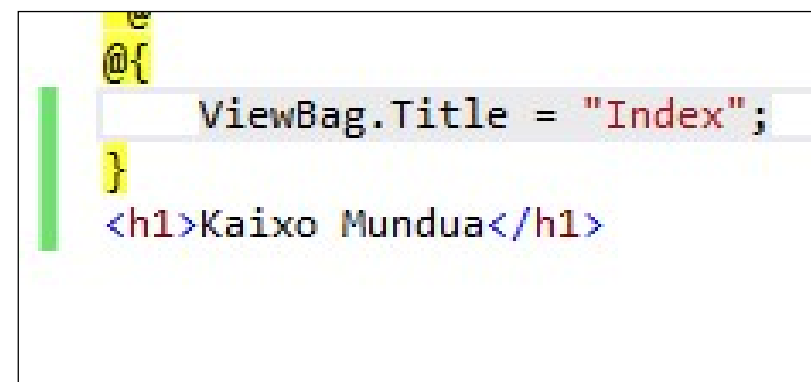
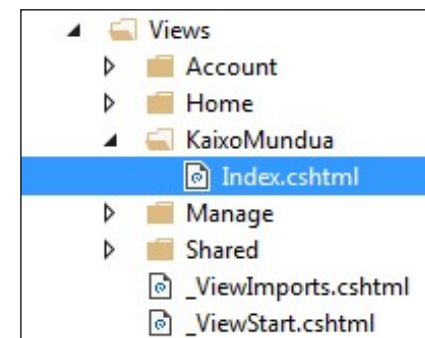
- Ikusi dugu kontroladoreak nola funtzionatzen duen, orain bistak nola erabili ahal ditugun ikusiko dugu. Kodigoa berriro aldatuko dugu eta ActionResult erabiliko dugu.
- Kontroladorean, Index metodoa aldatu return aldatzeko:

```
public IActionResult Index()  
{  
    return View();  
}
```

- **Views** karpeta barruan karpeta bat sortuko da, kontroladorearen izen berdinekin: **KaixoMundua** eta karpeta horren barruan bista bat sortuko dugu metodoaren izen berdinekin **Index**. (**Agregar-Vista-Vista de Razor: vacia**) Eta kodigoa gehitu:

```
@{  
    ViewBag.Title = "Index";  
}  
<h1>Kaixo Mundua</h1>
```

Kontroladorean jarri dugu «return View()», zein bista zabaldu nahi dugun zehaztu barik, beraz, metodoaren izen berdineko bista aurkitzen saiatuko da.



ASP .NET- MVC

1. adibidea:

- Txantiloak eta bistak aldatuko ditugu. Txantilo orokor bat dago:
/Views/Shared karpetan: _Layout.cshtml

- _Layout.cshtml aldaketak:

- Titulua: <title>@ViewData["Title"] – **Kaixo Mundua**</title>

Estiloak aldatzeko:

<link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />

<link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />

<link rel="stylesheet" href="~/Adibidea.styles.css" asp-append-version="true" />

wwwroot karpeta barruan **lib** karpetan **bootstrap** dago eta **css** karpetan **site.css** dago

Gure pertsonalizazioa egiteko site.css barruan aldatu behar dugu(navbar-inverse klasea aldatu daiteke).

ASP .NET- MVC

1. adibidea:

- Txantiloak eta bistak aldatuko ditugu. Txantiloak orokor bat dago:

/Views/Shared karpeta: **_Layout.cshtml**

- Topatu **@RenderBody()** lerroa. RenderBody kontenedore bat da, bista guztiak barruan exekutatu dira.

- _Layout.cshtml aldaketak:

- Titulua: <title>@ViewBag.Title - Ikastaroak App</title>

- Index.cshtml aldaketak:

- ViewBag.Title = "Ikastaroen zerrenda";

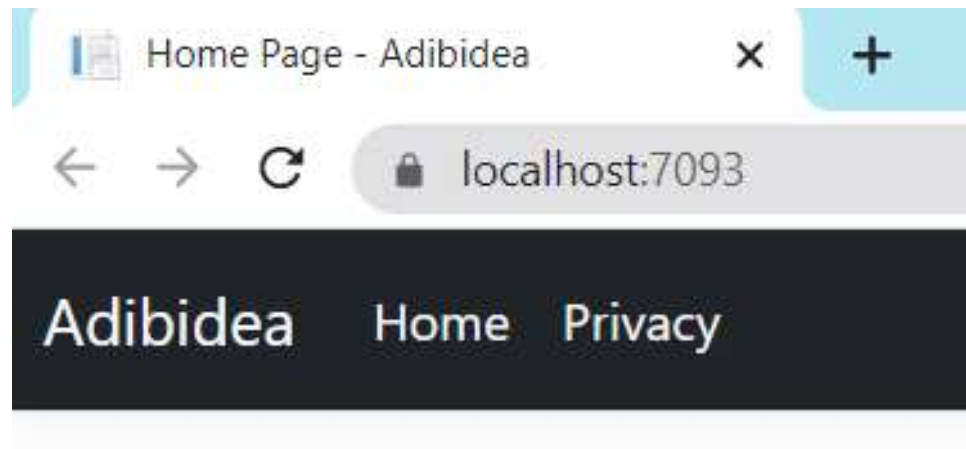
- @Styles.Render("~/Content/css")

Estiloak aldatzeko: Content karpeta barruan **bootstrap.css** eta **site.css** daude.

ASP .NET- MVC

1. adibidea:

Ariketa: Egin aldaketak goiko menuan atzealde beltza eta testua zuria agertzeko



ASP .NET- MVC

1. adibidea:

Parametroak pasatzen kontroladoretik bistara. Kontroladorearen kodigoa aldatuko dugu eta bista sortuko dugu (Views-KaixoMundua karpeta barruan):

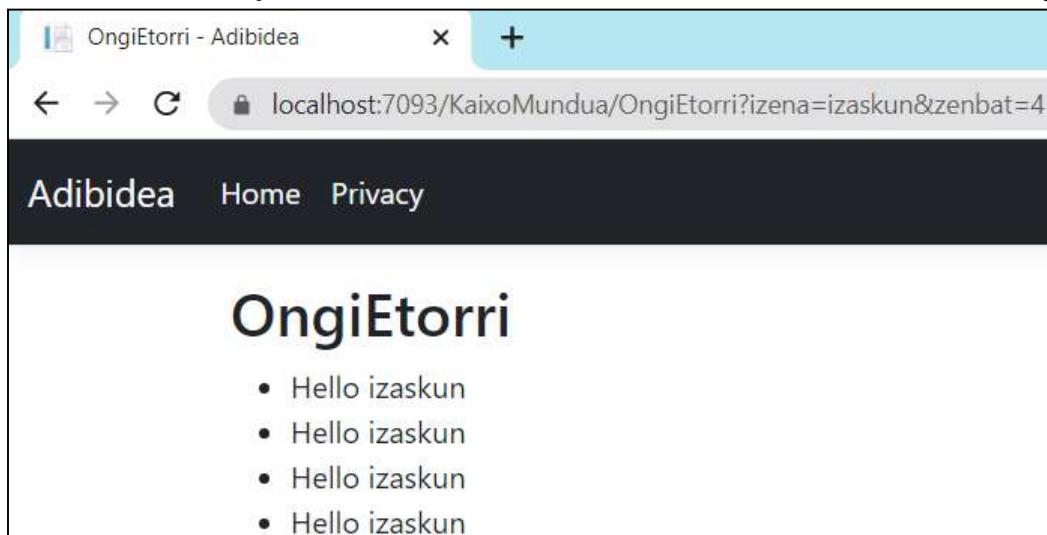
KaixoMunduaController.cs

```
public ActionResult OngiEtorri(string izena, int zenbat = 1)
{
    ViewBag.Mezua = "Hello " + izena;
    ViewBag.Zenbat = zenbat;
    return View();
}
```

OngiEtorri.cshtml

```
@{
    ViewBag.Title = "OngiEtorri";
}
<h2>OngiEtorri</h2>
<ul>
    @for (int i=0; i < ViewBag.Zenbat; i++) {
        <li>@ViewBag.Mezua</li>
    }
</ul>
```

Eta orria probatzeko: <https://localhost:7093/KaixoMundua/OngiEtorri?izena=izaskun&zenbat=4>



ASP .NET- MVC

1. adibidea:

Parametroak pasatzen kontroladoretik bistara hirugarren osagaia erabiliz, “**Models**”:

- Ereduaren osagaia gehituko dugu gure aplikazioan.
- Horretarako **Models** karpetan barruan klase bat sortuko dugu: Kontaktua

```
public class Kontaktua
{
    public Kontaktua(string nan, string izena, string abizena)
    {
        Nan = nan;
        Izena = izena;
        Abizena = abizena;
    }
    public string Nan { get; set; }
    public string Izena { get; set; }
    public string Abizena { get; set; }
    public virtual string Emaila { get; set; }
    public string IzenOsoa
    { get { return Izena + " " + Abizena; } }

    public virtual string Gorde()
    {
        return "Kontaktua ondo gorde da";
    }
}
```

ASP .NET- MVC

1. adibidea:

Kontrolador berria sortuko dugu: **AgendaController** (controlador de MVC con acciones de **lectura y escritura**)

Kontroladorearen kodigoa aldatuko dugu (**Details** metodoa):

AgendaController.cs

```
using WebApplication1.Models;
....
public ActionResult Details()
{
    Kontaktua kontaktua = new Kontaktua("13456765", "izaskun", "Kortabitarte");
    kontaktua.Emaila = "ikortabitarte@uni.eus";
    return View(kontaktua);
}
```

ASP .NET- MVC

1. adibidea:

Bista sortzeko **Details.cshtml** lehenengo karpeta bat sortu behar dugu: **Views/Agenda**

Metodoaren bertan **return View(kontaktua)** dagoen
gainean jarri eta eskuineko botoiarekin: **Agregar Vista – Vista de Razor**

Aukeratu:

Plantilla:Details

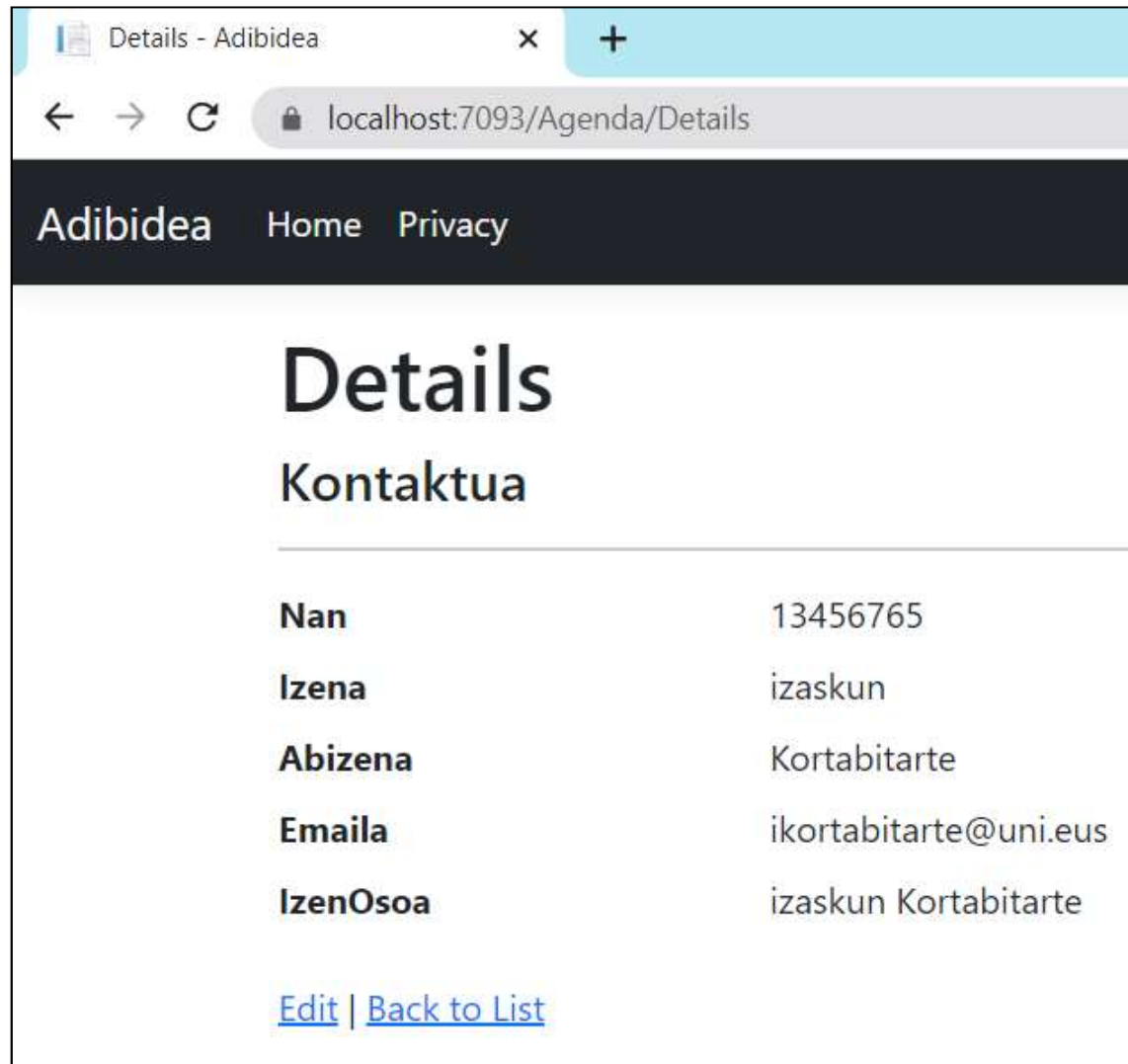
Clase de modelo: Kontaktua

Agregar Vista de Razor	
Nombre de vista	Details
Plantilla	Details
Clase de modelo	Kontaktua (Adibidea.Models)
Opciones	
<input type="checkbox"/>	Crear como vista parcial
<input checked="" type="checkbox"/>	Hacer referencia a bibliotecas de scripts
<input checked="" type="checkbox"/>	Usar página de diseño
<div></div>	
(Dejar en blanco si se define en un archivo _viewstart de Razor)	

ASP .NET- MVC

1. adibidea:

Sortzen du kodigo guztia eta honela bistaratuko da:



ASP .NET- MVC

1. adibidea:

Kontroladore berdinean, beste metodo batean, kontaktu guztiak ikusteko prestatuko dugu eta bista sortuko dugu:

AgendaController.cs

```
public ActionResult Index()
{
    List<Kontaktua> IstKontaktuak = new List<Kontaktua>();

    Kontaktua kontaktua1 = new Kontaktua("13456765", "izaskun", "Kortabitarte");
    kontaktua1.Emaila = "ikortabitarte@uni.eus";
    IstKontaktuak.Add(kontaktua1);

    Kontaktua kontaktua2 = new Kontaktua("13452345", "pepe", "Fernandez");
    kontaktua2.Emaila = "pfernandez@uni.eus";
    IstKontaktuak.Add(kontaktua2);

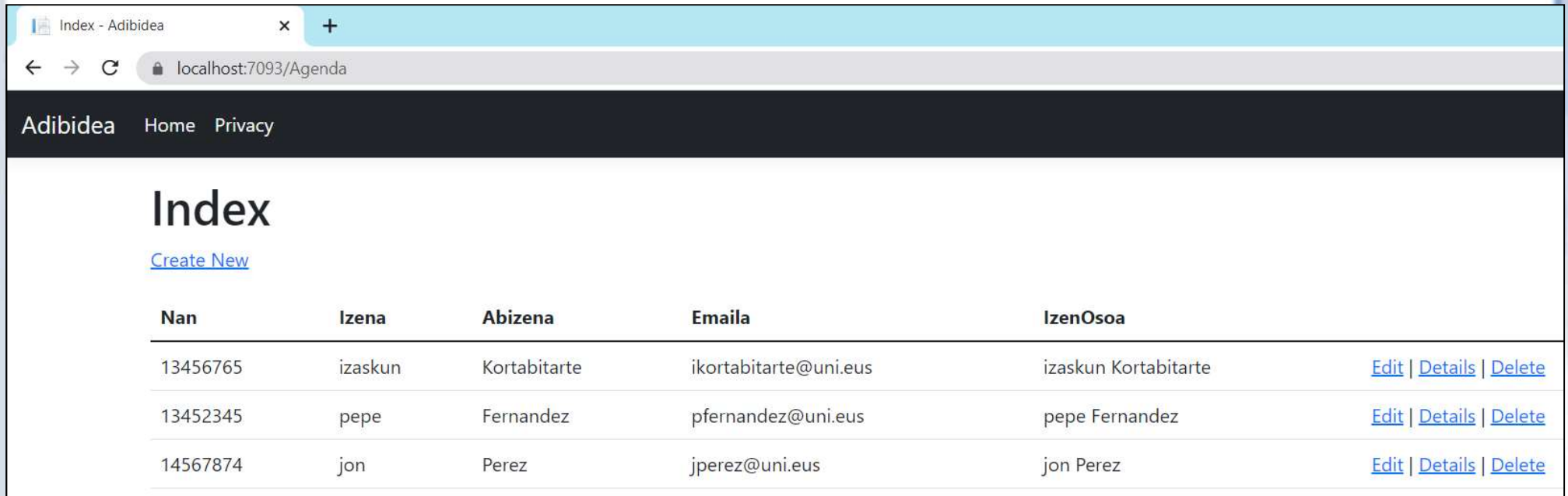
    Kontaktua kontaktua3 = new Kontaktua("14567874", "jon", "Perez");
    kontaktua3.Emaila = "jperez@uni.eus";
    IstKontaktuak.Add(kontaktua3);

    return View(IstKontaktuak);
}
```

Bista sortzeko **Agenda** karpeta barruan, metodoaren izen berdinekin eta txantiloia “**List**”: **Index.cshtml**

ASP .NET- MVC

1. adibidea:



Nan	Izena	Abizena	Emaila	IzenOsoa	
13456765	izaskun	Kortabitarte	ikortabitarte@uni.eus	izaskun Kortabitarte	Edit Details Delete
13452345	pepe	Fernandez	pfernandez@uni.eus	pepe Fernandez	Edit Details Delete
14567874	jon	Perez	jperez@uni.eus	jon Perez	Edit Details Delete

ASP .NET- MVC

Honaino iritsi bazara, **Zorionak!!!**

Amaitu duzu ASP .NET erabiliz egindako lehen aplikazioa

Ez ahaztu **GitHub**-era igotzen!!!.

ASP .NET- MVC

Estekak:

<https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-mvc-app/start-mvc?view=aspnetcore-6.0&tabs=visual-studio>