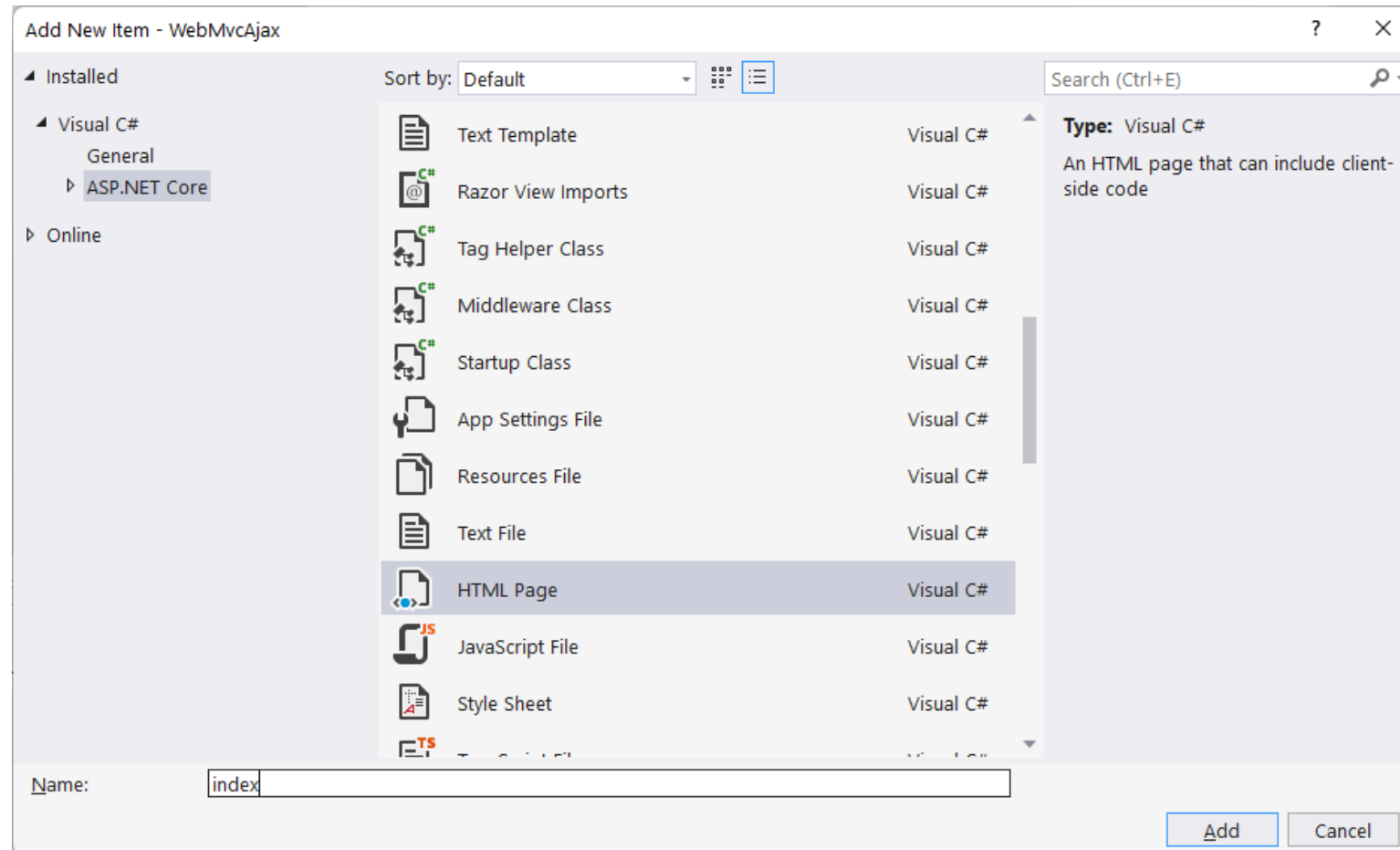


Asinhroni HTTP zahtevi

Uvod u AJAX i Fetch Api

- AJAX = Asynchronous JavaScript and XML
- AJAX omogućava parcijalno ažuriranje html strane razmenom malih količina podataka sa serverom
- Pri klasičnom zahtevu html strana mora da promeni kompletni sadržaj iako se mali deo strane menja
- Fetch API koristi AJAX (Asynchronous JavaScript and XML) poziv za preuzimanje podataka sa servera
- Fetch API je JavaScript interfejs koji omogućava asinhrono slanje HTTP zahteva prema serveru i prijem odgovora
- Fetch API je zamena za starije AJAX tehnologije i pruža bolju podršku za moderne veb tehnologije

Dodavanje strane index.html folder wwwroot



Fajl Program.cs

Postavlja se početna strana sajta index.html

```
app.UseDefaultFiles();
app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

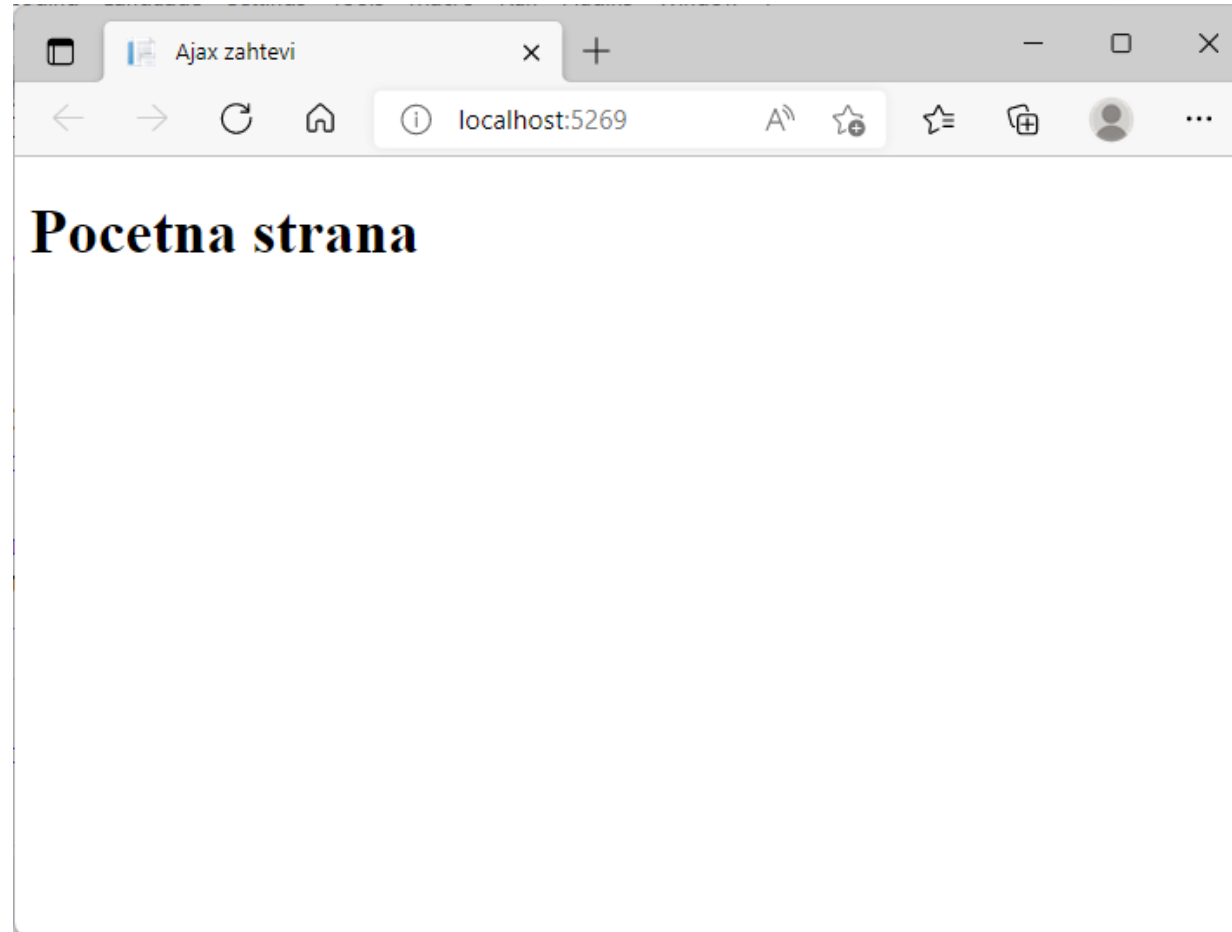
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();
```

Strana index.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Ajax zahtevi</title>
</head>
<body>
  <h1>Pocetna strana</h1>
</body>
</html>
```

Početna strana aplikacije



Model folder klasa Osoba

```
namespace WebApp07.Models
{
    public class Osoba
    {
        public int OsobaId { get; set; }
        public string Ime { get; set; }
        public string Prezime { get; set; }
        public string Telefon { get; set; }
    }
}
```

Klasa HomeController

```
public class HomeController : Controller
{
    private readonly List<Osoba> listaOsoba = new List<Osoba>();
    public HomeController()
    {
        Osoba os1 = new Osoba { OsobaId = 1, Ime = "Janko", Prezime = "Petrovic", Telefon = "123-456" };
        Osoba os2 = new Osoba { OsobaId = 2, Ime = "Mika", Prezime = "Mikic", Telefon = "345-458" };
        Osoba os3 = new Osoba { OsobaId = 3, Ime = "Ivana", Prezime = "Ivanovic", Telefon = "567-785" };
        Osoba os4 = new Osoba { OsobaId = 4, Ime = "Jovana", Prezime = "Jovanovic", Telefon = "789-6756" };

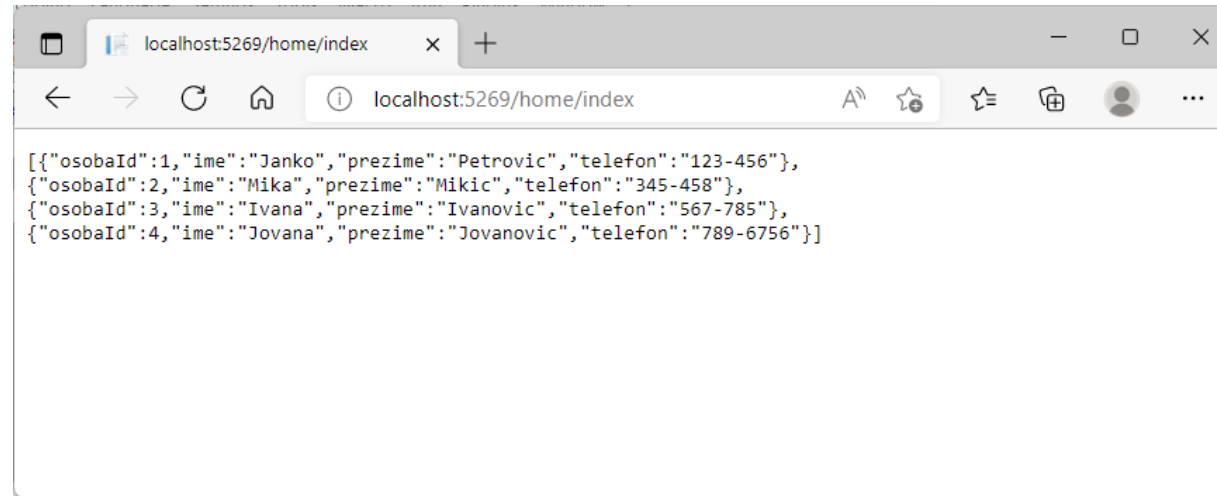
        listaOsoba.Add(os1);
        listaOsoba.Add(os2);
        listaOsoba.Add(os3);
        listaOsoba.Add(os4);
    }
}
```


Metoda Index klase HomeController

```
public JsonResult Index(int id = 0)
{
    if (id == 0)
    {
        return Json(listaOsoba);
    }
    else
    {
        Osoba os = listaOsoba.SingleOrDefault(o => o.OsobaId == id);

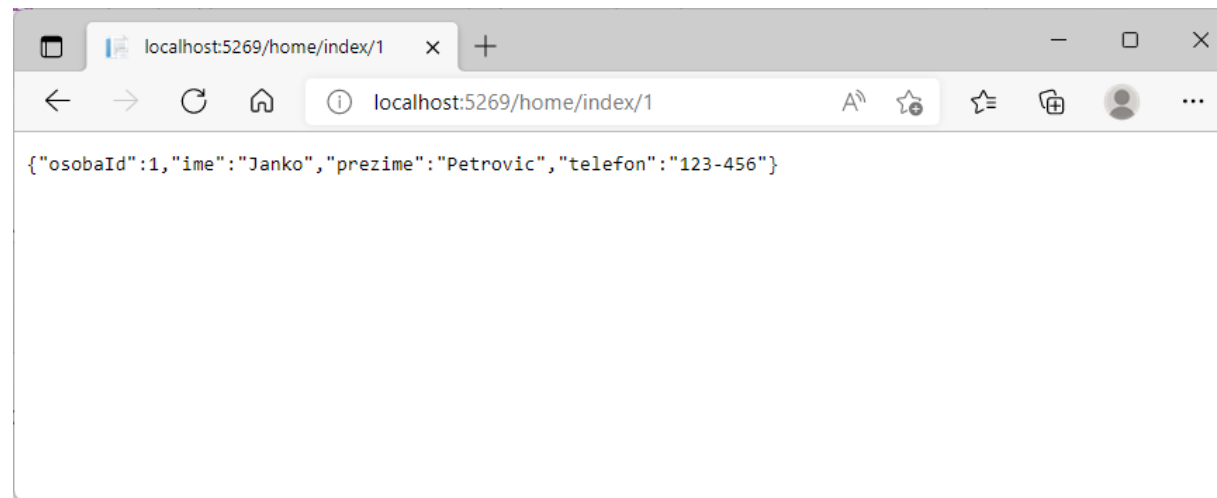
        if (os != null)
        {
            return Json(os);
        }
        else
        {
            return Json(new Osoba());
        }
    }
}
```

Podaci koje generiše server



A screenshot of a web browser window with the address bar showing 'localhost:5269/home/index'. The main content area displays a JSON array of four objects, each representing a person with attributes: osobaId, ime, prezime, and telefon.

```
[{"osobaId":1,"ime":"Janko","prezime":"Petrovic","telefon":"123-456"},  
{"osobaId":2,"ime":"Mika","prezime":"Mikic","telefon":"345-458"},  
{"osobaId":3,"ime":"Ivana","prezime":"Ivanovic","telefon":"567-785"},  
{"osobaId":4,"ime":"Jovana","prezime":"Jovanovic","telefon":"789-6756"}]
```



A screenshot of a web browser window with the address bar showing 'localhost:5269/home/index/1'. The main content area displays a single JSON object representing the first person from the array above.

```
{"osobaId":1,"ime":"Janko","prezime":"Petrovic","telefon":"123-456"}
```

Strana index.html

```
<body>
  <h1>Pocetna strana</h1>
  <span id="span1"></span>
  <hr>
  <form>
    Unesite id osobe:<input id="Text1" type="text"><br>
    <input id="Button1" type="button" value="Pronadji" onclick="PronadjiOsobu()">
  </form>
  <br />
  <span id="span2"></span>
</body>
</html>
```

Fetch api

- `fetch()` API je moderan način za slanje HTTP zahteva u JavaScriptu i zamjenjuje starije metode kao što su `XMLHttpRequest` i `jQuery AJAX`
- `fetch()` API je standardni način za slanje asinhronih HTTP zahtjeva u JavaScriptu
- `fetch()` API omogućuje slanje različitih vrsta HTTP zahteva, uključujući `GET`, `POST`, `PUT`, `DELETE` i druge
- `fetch()` API ima ugrađenu podršku za rad s JSON podacima, binarnim podacima, HTML-om i drugim vrstama sadržaja

Fetch api- generisanje zahteva

```
const odgovor = await fetch('/Home/Index');
```

- Ovde je `odgovor` objekat koji predstavlja HTTP odgovor koji se vraća nakon što se izvrši AJAX zahtev na nekoj adresi
- Svojstvo **ok** objekta **odgovor** vraća `true` ukoliko je odgovor uspešan
- Svojstvo **status** objekta **odgovor** predstavlja kod statusa HTTP odgovora koji server vraća kao odgovor na zahtev koji je poslat preko AJAX-a
- HTTP statusni kodovi predstavljaju tri cifre koje se vraćaju sa servera kao odgovor na zahtev
- Statusni kod 200 znači da je zahtev uspešno obrađen
- Statusni kod 404 znači da server nije mogao da pronađe tražene resurse.

Fetch api- generisanje zahteva

- Metoda **json()** objekta odgovor parsira JSON string koji je vraćen kao telo HTTP odgovora i vraća JavaScript objekte koji predstavljaju podatke koji su bili u tom JSON-u
- Error je ugrađeni objekat u JavaScript koji predstavlja greške koje se mogu javiti tokom izvršavanja programa.

Implementacija asinhronog HTTP zahteva

```
async function VratiOsobe() {  
  const odgovor = await fetch('/Home/Index');  
  
  if (!odgovor.ok) {  
    throw new Error(`HTTP error! Status: ${odgovor.status}`);  
  }  
  
  const osobe = await odgovor.json();  
  return osobe;  
}
```

Funkcija za prikaz podataka

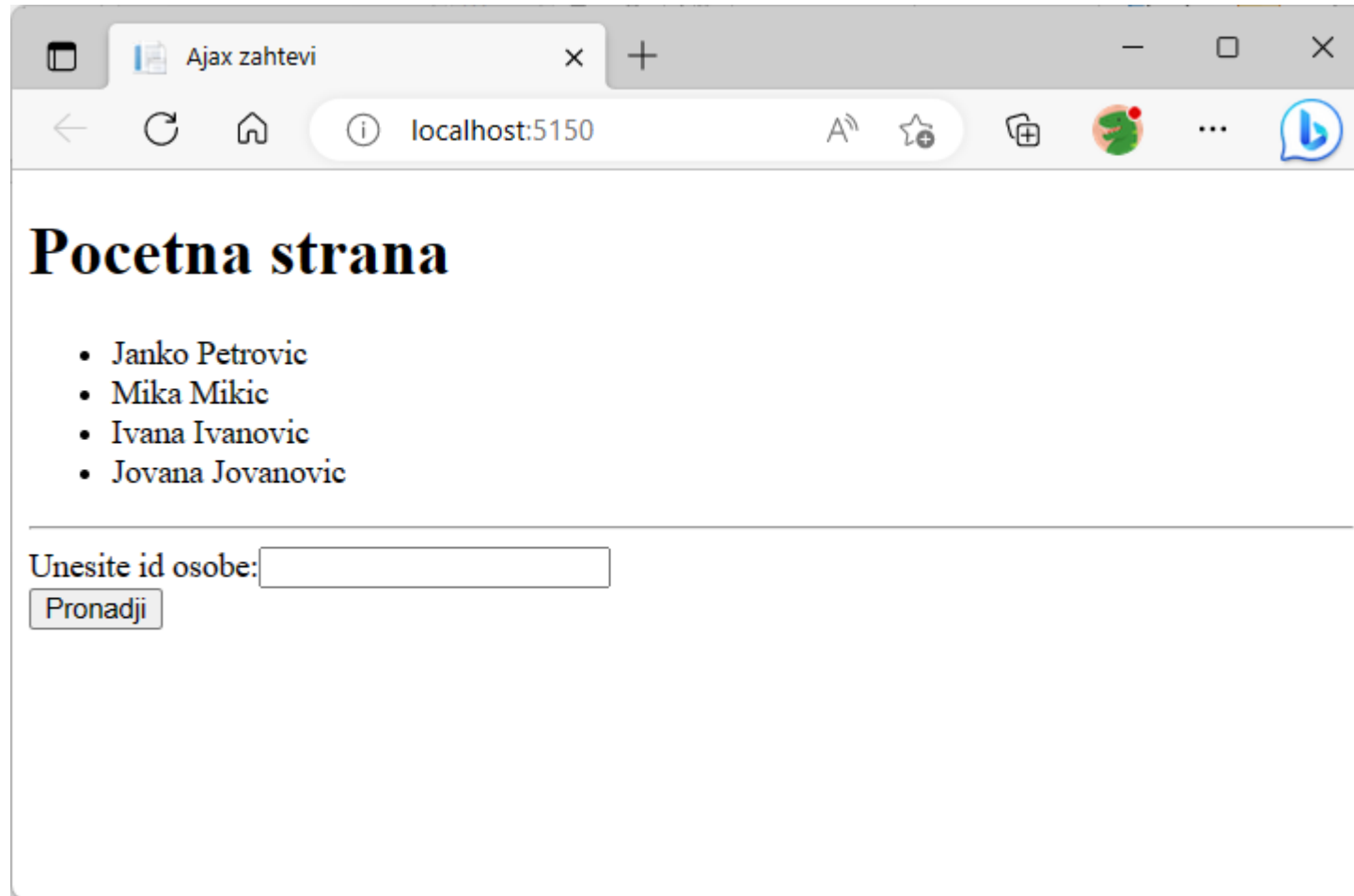
```
async function prikaziOsobe() {
  try {
    const osobe = await VratiOsobe();
    const span1 = document.getElementById("span1");
    let s = "<ul>";

    for (let i = 0; i < osobe.length; i++) {
      s += `<li>${osobe[i].ime} ${osobe[i].prezime}</li>`;
    }

    s += "</ul>";
    span1.innerHTML = s;
  } catch (error) {
    console.log(error);
  }
}
```

```
prikaziOsobe();
```


Podaci na strani index.html



Javascript funkcija VратиOsobu

```
async function VратиOsobu(id) {  
    const odgovor = await fetch(`/Home/Index/${id}`);  
  
    if (!odgovor.ok) {  
        throw new Error(`HTTP error! Status: ${odgovor.status}`);  
    }  
  
    const osoba = await odgovor.json();  
    return osoba;  
}
```

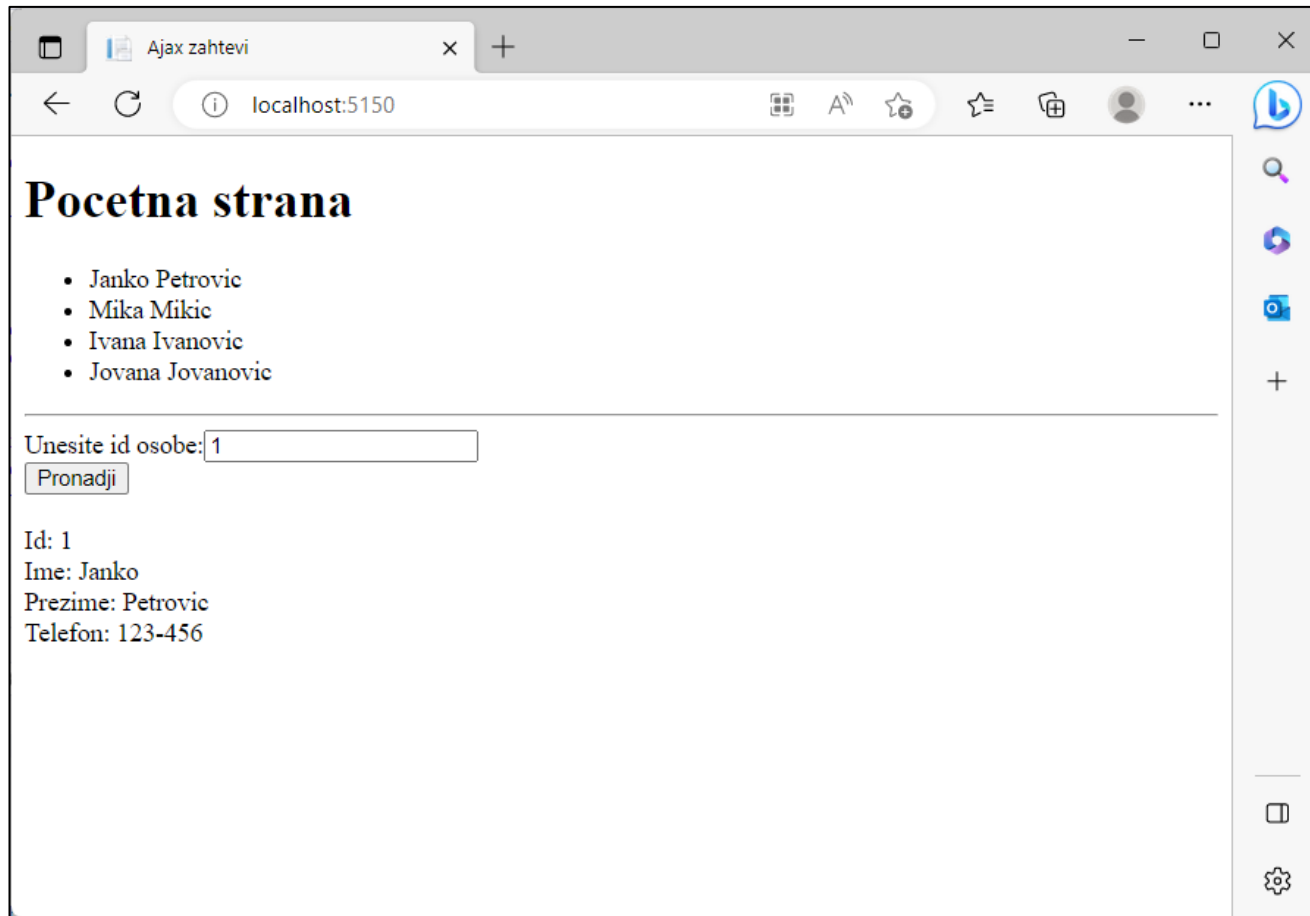
Javascript funkcija IscrtajOsobu

```
function iscrtajOsobu(osoba) {  
    const span2 = document.getElementById("span2");  
    if (osoba == null) {  
        span2.innerHTML = "Nije pronađena osoba."  
    } else {  
        span2.innerHTML = `  
        Id: ${osoba.osobaId}<br>  
        Ime: ${osoba.ime}<br>  
        Prezime: ${osoba.prezime}<br>  
        Telefon: ${osoba.telefon} `;  
    }  
}
```

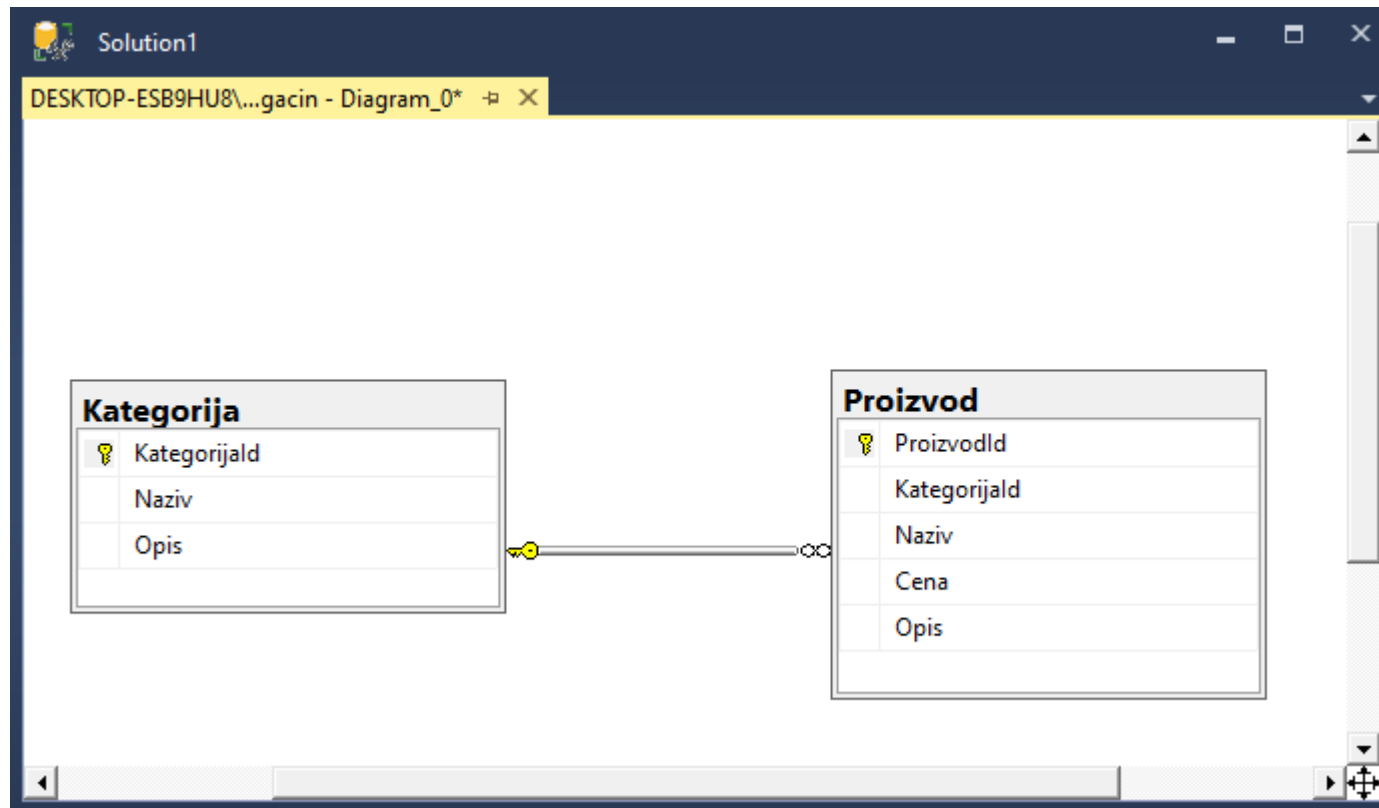
Javascript funkcija PronadjiOsobu()

```
async function PronadjiOsobu() {  
    var Text1 = document.getElementById("Text1");  
  
    var id = Text1.value;  
    if (isNaN(id)) {  
        alert("Unesite broj");  
        span2.innerHTML = "";  
        Text1.value = "";  
        return;  
    }  
  
    try {  
        const osoba = await VratiOsobu(id);  
        iscrtajOsobu(osoba);  
    } catch (error) {  
        console.log(error);  
        span2.innerHTML = "Nije pronadjena osoba";  
    }  
}
```

Rezultat pretrage



Baza podataka



Primenom alata EF Core Power Tools kreirati model podataka na osnovu baze Magacin

Models folder klasa Proizvod

```
public partial class Proizvod
{
    [Key]
    public int ProizvodId { get; set; }

    public int KategorijaId { get; set; }

    [Required]
    [StringLength(120)]
    public string Naziv { get; set; }

    [Column(TypeName = "decimal(10, 2)")]
    public decimal Cena { get; set; }

    [StringLength(120)]
    public string Opis { get; set; }

    [JsonIgnore]
    [ForeignKey("KategorijaId")]
    [InverseProperty("Proizvodi")]
    public virtual Kategorija Kategorija { get; set; }
}
```

Ne serijalizuje se navigacioni properti

Model klasa Kategorija

```
[Table("Kategorija")]
public partial class Kategorija
{
    [Key]
    public int KategorijaId { get; set; }

    [Required]
    [StringLength(70)]
    public string Naziv { get; set; }

    [StringLength(120)]
    public string Opis { get; set; }

    [InverseProperty("Kategorija")]
    public virtual ICollection<Proizvod> Proizvodi { get; } = new List<Proizvod>();
}
```


DbContext klasa

```
public partial class MagacinContext : DbContext
{
    public MagacinContext(DbContextOptions<MagacinContext> options)
        : base(options)
    {
    }

    public virtual DbSet<Kategorija> Kategorije { get; set; }

    public virtual DbSet<Proizvod> Proizvodi { get; set; }
}
```

appsettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=GORAN-HP;Initial Catalog=Magacin;
    Integrated Security=True;Encrypt=False"
  }
}
```

Registracija DbContext klase kao servisa

```
var builder = WebApplication.CreateBuilder(args);  
var connectionString =  
builder.Configuration.GetConnectionString("DefaultConnection");  
  
builder.Services.AddDbContext<MagacinContext>(options =>  
    options.UseSqlServer(connectionString));
```

HomeController

```
public class HomeController : Controller
{
    private readonly MagacinContext _db;

    public HomeController(MagacinContext db)
    {
        _db = db;
    }
}
```

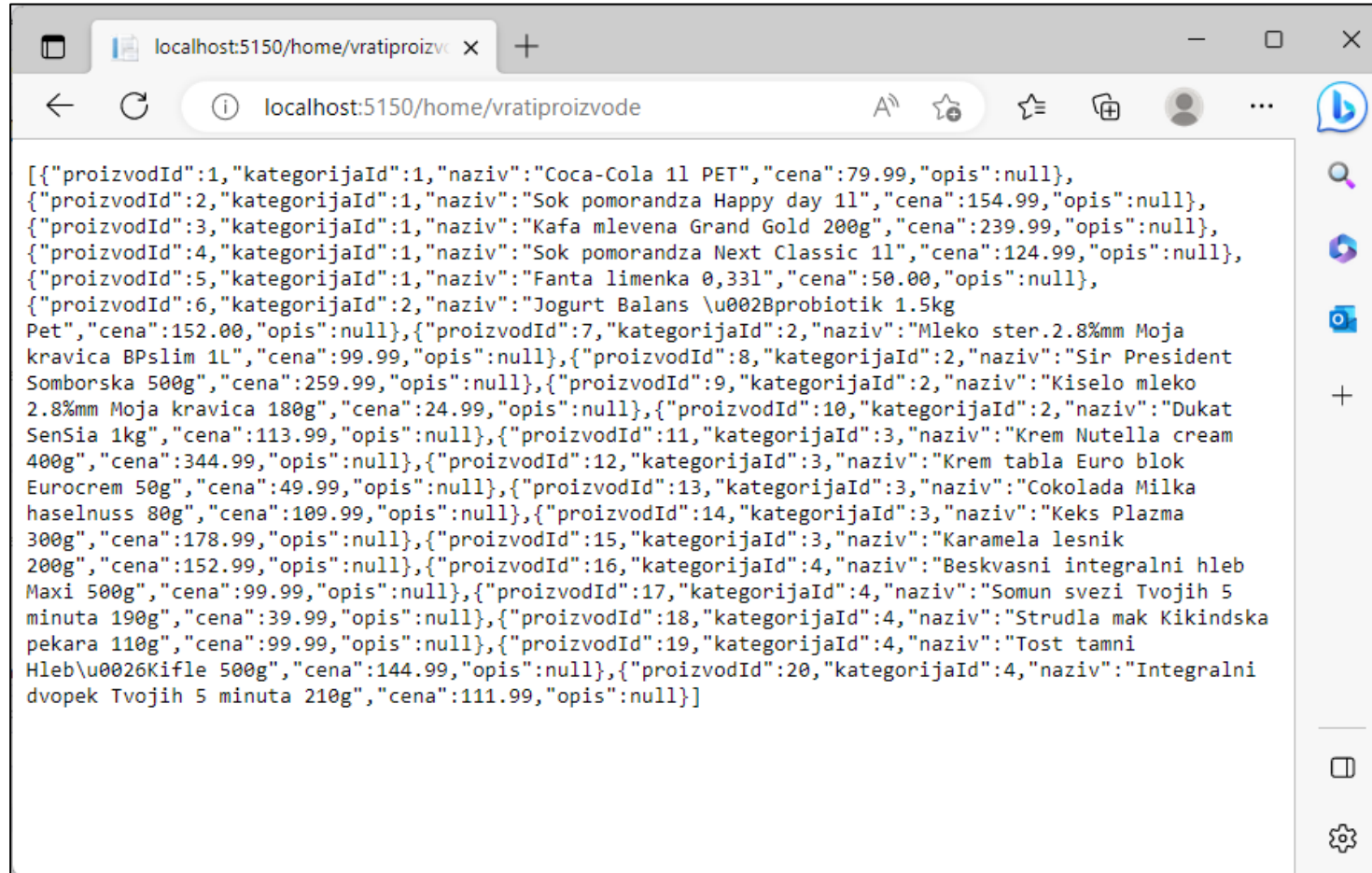
Metoda VратиProizvode() klasa klasa HomeController

```
public JsonResult VратиProizvode(int id = 0)
{
    IEnumerable<Proizvod> proizvodi = _db.Proizvodi;

    if (id != 0)
    {
        Kategorija k1 = _db.Kategorije.Find(id); // F
        if (k1 != null)
        {
            proizvodi = proizvodi.Where(p => p.KategorijaId == id);
        }
        else
        {
            return Json(new List<Proizvod> { new Proizvod { ProizvodId = 0 } });
        }
    }

    return Json(proizvodi);
}
```

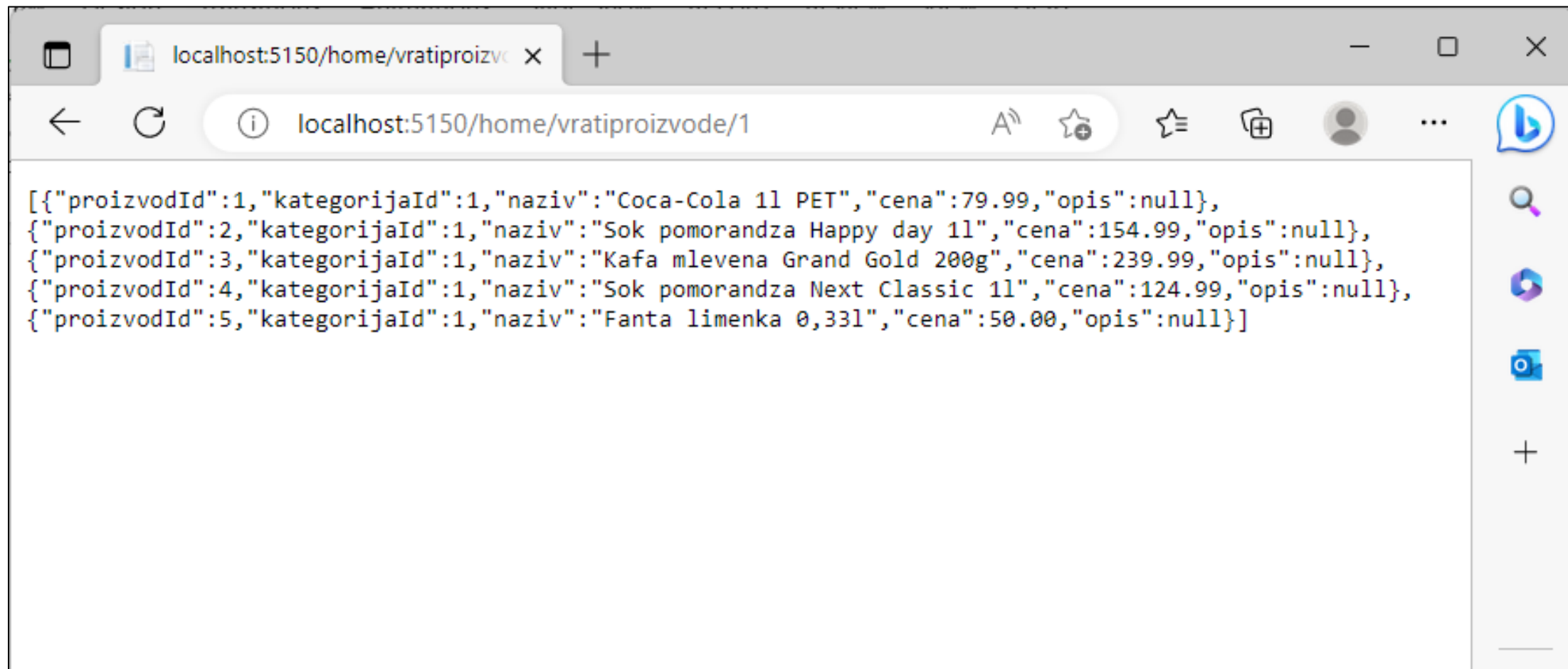
Poziv serverske metode VratiProizvode



The screenshot shows a web browser window with the address bar displaying 'localhost:5150/home/vratiproizvode'. The main content area contains a JSON array of 20 objects, each representing a product with fields for 'proizvodId', 'kategorijaId', 'naziv', 'cena', and 'opis'. The products listed include Coca-Cola 1l PET, Sok pomorandza Happy day 1l, Kafa mlevena Grand Gold 200g, Sok pomorandza Next Classic 1l, Fanta limenka 0,33l, Jogurt Balans \u002Bprobiotik 1.5kg Pet, Mleko ster.2.8%mm Moja kravica BPslim 1L, Sir President Somborska 500g, Kiselo mleko 2.8%mm Moja kravica 180g, Dukat SenSia 1kg, Krem Nutella cream 400g, Krem tabla Euro blok Eurocrem 50g, Cokolada Milka haselnuss 80g, Keksz Plazma 300g, Karamela lesnik 200g, Beskvasni integralni hleb Maxi 500g, Somun svezi Tvojih 5 minuta 190g, Strudla mak Kikindska pekara 110g, Tost tamni Hleb\u0026Kifle 500g, and Integralni dvopek Tvojih 5 minuta 210g.

```
[{"proizvodId":1,"kategorijaId":1,"naziv":"Coca-Cola 1l PET","cena":79.99,"opis":null}, {"proizvodId":2,"kategorijaId":1,"naziv":"Sok pomorandza Happy day 1l","cena":154.99,"opis":null}, {"proizvodId":3,"kategorijaId":1,"naziv":"Kafa mlevena Grand Gold 200g","cena":239.99,"opis":null}, {"proizvodId":4,"kategorijaId":1,"naziv":"Sok pomorandza Next Classic 1l","cena":124.99,"opis":null}, {"proizvodId":5,"kategorijaId":1,"naziv":"Fanta limenka 0,33l","cena":50.00,"opis":null}, {"proizvodId":6,"kategorijaId":2,"naziv":"Jogurt Balans \u002Bprobiotik 1.5kg Pet","cena":152.00,"opis":null}, {"proizvodId":7,"kategorijaId":2,"naziv":"Mleko ster.2.8%mm Moja kravica BPslim 1L","cena":99.99,"opis":null}, {"proizvodId":8,"kategorijaId":2,"naziv":"Sir President Somborska 500g","cena":259.99,"opis":null}, {"proizvodId":9,"kategorijaId":2,"naziv":"Kiselo mleko 2.8%mm Moja kravica 180g","cena":24.99,"opis":null}, {"proizvodId":10,"kategorijaId":2,"naziv":"Dukat SenSia 1kg","cena":113.99,"opis":null}, {"proizvodId":11,"kategorijaId":3,"naziv":"Krem Nutella cream 400g","cena":344.99,"opis":null}, {"proizvodId":12,"kategorijaId":3,"naziv":"Krem tabla Euro blok Eurocrem 50g","cena":49.99,"opis":null}, {"proizvodId":13,"kategorijaId":3,"naziv":"Cokolada Milka haselnuss 80g","cena":109.99,"opis":null}, {"proizvodId":14,"kategorijaId":3,"naziv":"Keksz Plazma 300g","cena":178.99,"opis":null}, {"proizvodId":15,"kategorijaId":3,"naziv":"Karamela lesnik 200g","cena":152.99,"opis":null}, {"proizvodId":16,"kategorijaId":4,"naziv":"Beskvasni integralni hleb Maxi 500g","cena":99.99,"opis":null}, {"proizvodId":17,"kategorijaId":4,"naziv":"Somun svezi Tvojih 5 minuta 190g","cena":39.99,"opis":null}, {"proizvodId":18,"kategorijaId":4,"naziv":"Strudla mak Kikindska pekara 110g","cena":99.99,"opis":null}, {"proizvodId":19,"kategorijaId":4,"naziv":"Tost tamni Hleb\u0026Kifle 500g","cena":144.99,"opis":null}, {"proizvodId":20,"kategorijaId":4,"naziv":"Integralni dvopek Tvojih 5 minuta 210g","cena":111.99,"opis":null}]
```

Filtriranje proizvoda po kategorijama

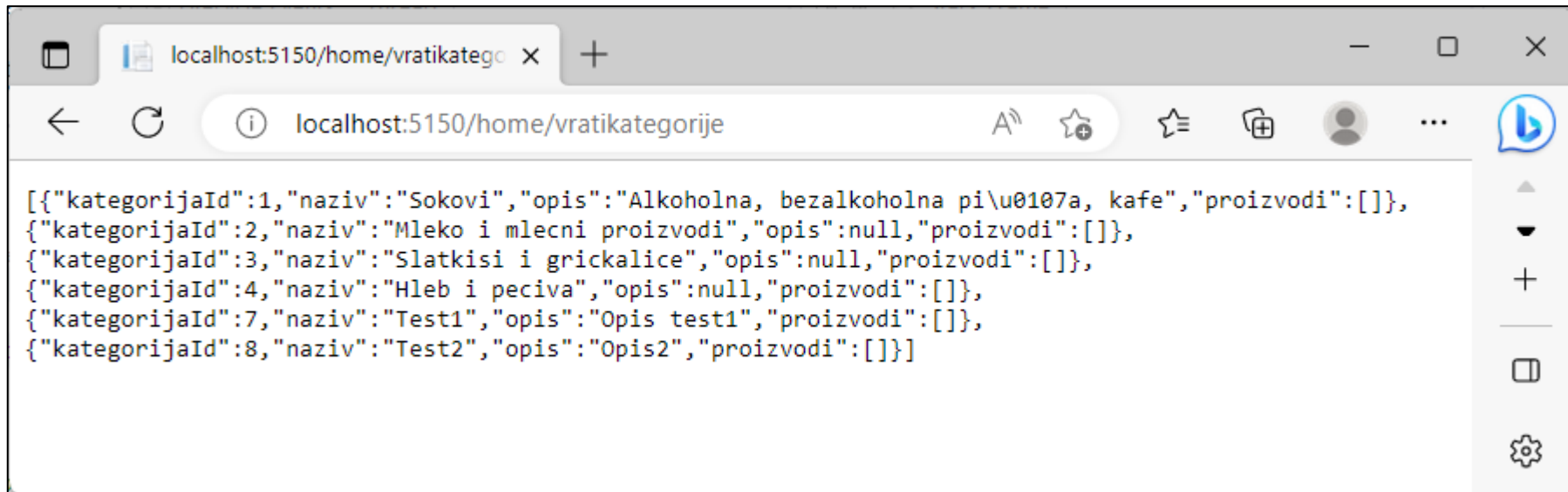


A screenshot of a web browser window. The address bar shows the URL `localhost:5150/home/vratiproizvode/1`. The main content area displays a JSON array of five product objects, all filtered under category ID 1. The browser interface includes a tab, navigation buttons, and a sidebar with various icons.

```
[{"proizvodId":1,"kategorijaId":1,"naziv":"Coca-Cola 1l PET","cena":79.99,"opis":null}, {"proizvodId":2,"kategorijaId":1,"naziv":"Sok pomorandza Happy day 1l","cena":154.99,"opis":null}, {"proizvodId":3,"kategorijaId":1,"naziv":"Kafa mlevena Grand Gold 200g","cena":239.99,"opis":null}, {"proizvodId":4,"kategorijaId":1,"naziv":"Sok pomorandza Next Classic 1l","cena":124.99,"opis":null}, {"proizvodId":5,"kategorijaId":1,"naziv":"Fanta limenka 0,33l","cena":50.00,"opis":null}]
```

Metoda VратиKategorije() HomeController

```
public JsonResult VратиKategorije()
{
    return Json(_db.Kategorije.ToList());
}
```



The screenshot shows a web browser window with the address bar displaying `localhost:5150/home/vratikategorije`. The main content area displays a JSON array of category objects. The browser interface includes a back button, a refresh button, and a search bar. The JSON response is as follows:

```
[{"kategorijaId":1,"naziv":"Sokovi","opis":"Alkoholna, bezalkoholna pi\u0107a, kafe","proizvodi":[]},
{"kategorijaId":2,"naziv":"Mleko i mlečni proizvodi","opis":null,"proizvodi":[]},
{"kategorijaId":3,"naziv":"Slatkisi i grickalice","opis":null,"proizvodi":[]},
{"kategorijaId":4,"naziv":"Hleb i peciva","opis":null,"proizvodi":[]},
{"kategorijaId":7,"naziv":"Test1","opis":"Opis test1","proizvodi":[]},
{"kategorijaId":8,"naziv":"Test2","opis":"Opis2","proizvodi":[]}]
```


Korisnički interfejs strane Index.cshtml

```
<div class="row">
  <div class="col-md-6">
    <form id="form1" action="" method="get">
      <div class="form-group">
        <label for="Select1">Odaberi kategoriju</label> <br>
        <select name="KategorijaId" id="Select1" class="form-control" onchange="Prikazi()">
          <option value="0">Svi proizvodi</option>
        </select>
      </div>
    </form>
  </div>
</div>
```

Korisnički interfejs strane Index.cshtml

```
<div class="row">
  <div class="col-md-6">

    <table class="table">
      <thead>
        <tr>
          <th>naziv</th>
          <th>cena</th>
        </tr>
      </thead>
      <tbody id="table1">
      </tbody>
    </table>

  </div>
</div>
```

JavaScript kod u pogledu Index.cshtml

```
@section Scripts{
  <script>

    async function Vratikategorije() {
      const response = await fetch('/home/Vratikategorije');
      if (!response.ok) {
        throw new Error('Network response was not ok.');      }
      const kategorije = await response.json();
      return kategorije;
    }

  </script>
}
```

Prikaz kategorija u Select elementu

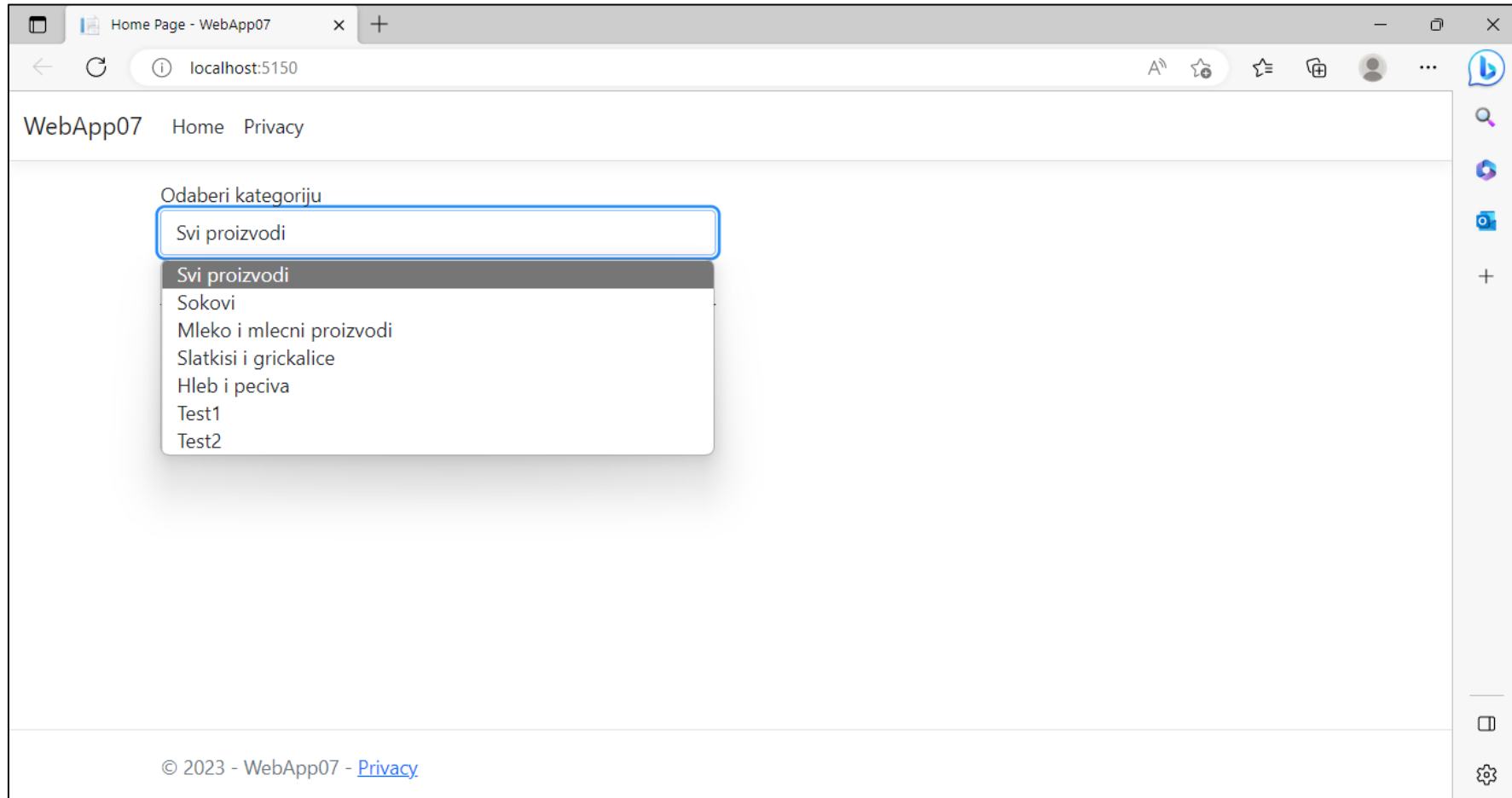
```
function PrikaziKategorije(kategorije) {  
  const select1 = document.getElementById("Select1");  
  kategorije.forEach((kategorija) => {  
    let option = document.createElement("option");  
    option.text = kategorija.naziv;  
    option.value = kategorija.kategorijaId;  
    select1.appendChild(option);  
  });  
}
```

Prikaz kategorija

```
window.onload = function () {  
  Vratikategorije().then(PrikaziKategorije).catch(console.error);  
};
```

Uz pomoć then() metode, možemo da preuzmemo podatke sa servera, i nakon toga manipuliramo sa njima ili prikažemo na web stranici.

Prikaz kategorija



JavaScript funkcija koja vraća proizvode

```
async function VratiProizvode(id) {  
  const response = await fetch(`/home/vratiproizvode/${id}`);  
  if (!response.ok) {  
    throw new Error('Network response was not ok');  
  }  
  const proizvodi = await response.json();  
  console.log(proizvodi); //  
  return proizvodi;  
}
```

Prikaz proizvoda u body delu tabele

```
function PrikaziProizvode(proizvodi) {  
    const table1 = document.getElementById("table1");  
    table1.innerHTML = "";  
  
    for (let i = 0; i < proizvodi.length; i++) {  
        const tr = document.createElement("tr");  
        const td1 = document.createElement("td");  
        const td1Tekst = document.createTextNode(proizvodi[i].naziv);  
        td1.appendChild(td1Tekst);  
  
        const td2 = document.createElement("td");  
        const td2Tekst = document.createTextNode(proizvodi[i].cena);  
        td2.appendChild(td2Tekst);  
  
        tr.appendChild(td1);  
        tr.appendChild(td2);  
        table1.appendChild(tr);  
    }  
}
```


Obrada onchange događaja Select elementa

```
async function Prikazi() {  
  const Select1 = document.getElementById("Select1");  
  const id = Select1.value;  
  
  try {  
    const proizvodi = await VratiProizvode(id);  
    PrikaziProizvode(proizvodi);  
  } catch (error) {  
    console.error(error);  
  }  
}
```

Rezultat filtriranja

WebApp07 Home Privacy

Odaberi kategoriju

Sokovi

naziv	cena
Coca-Cola 1l PET	79.99
Sok pomorandza Happy day 1l	154.99
Kafa mlevena Grand Gold 200g	239.99
Sok pomorandza Next Classic 1l	124.99
Fanta limenka 0,33l	50

© 2023 - WebApp07 - [Privacy](#)

Metoda koja vraća parcijalni pogled

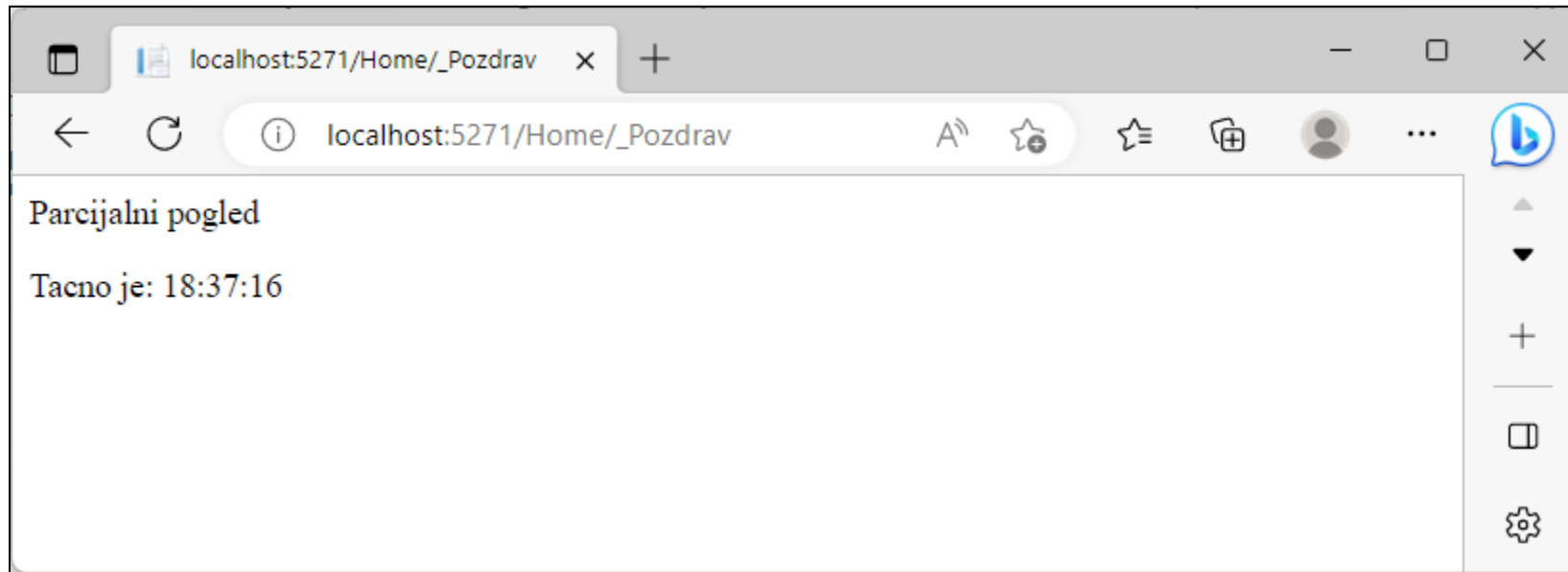
```
public PartialViewResult _Pozdrav()  
{  
    return PartialView();  
}
```

Sadržaj parcijalnog pogleda _Pozdrav

```
<div class="col-md-6">
  <div class="border border-primary p-2">
    Parcijalni pogled
  <p>
    Tacno je: @DateTime.Now.ToLongTimeString()
  </p>
</div>
</div>
```

Standardni GET poziv parcijalnog pogleda

```
<a asp-action="_Pozdrav">Poziv parcijalnog pogleda</a>
```



Index.cshtml klasa HomeController

```
@{
    ViewData["Title"] = "Home Page";
}

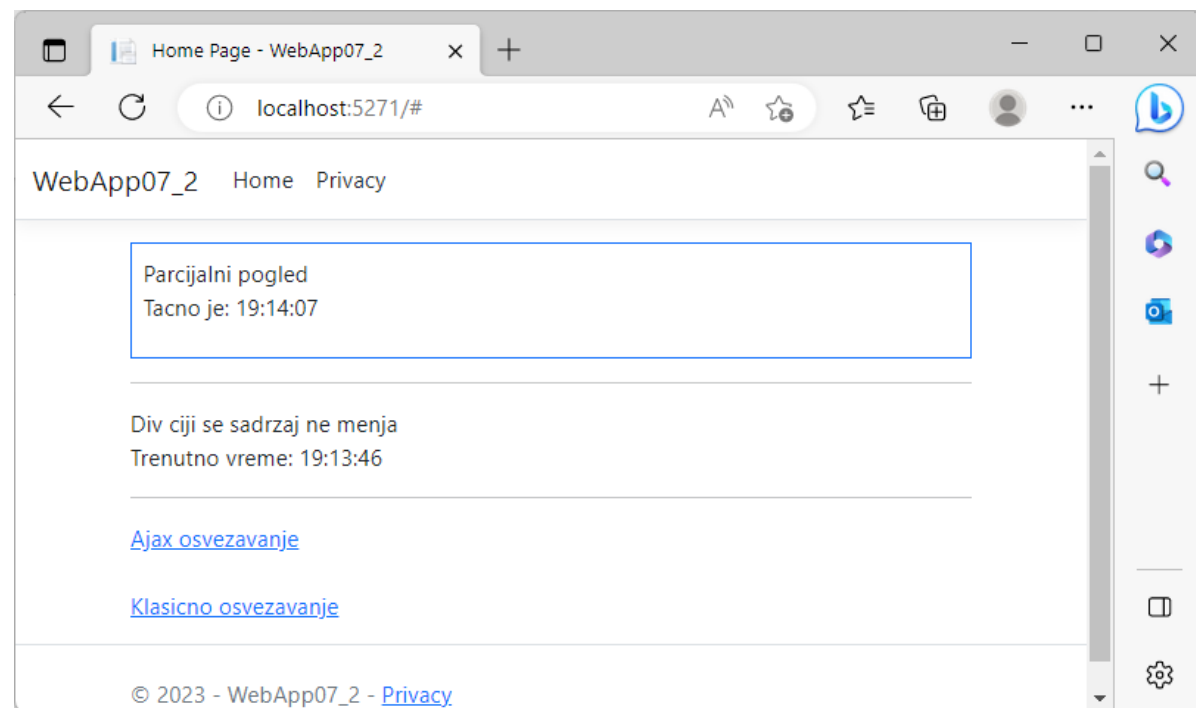
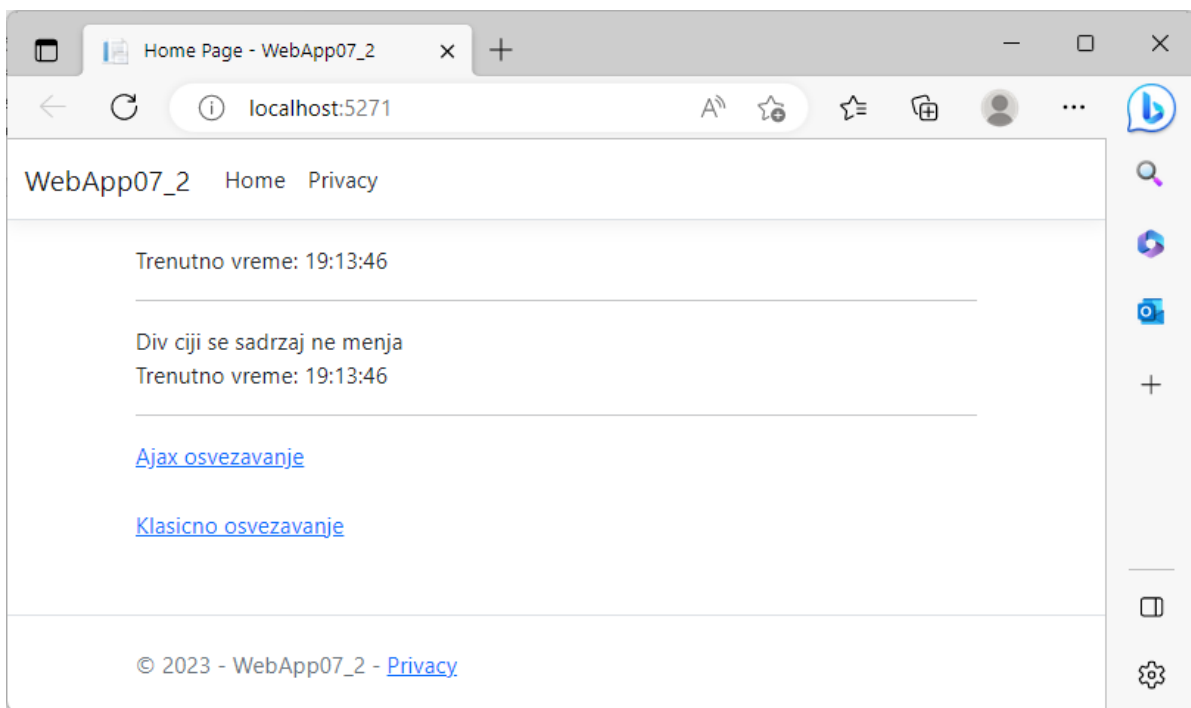
<div class="row mb-2" id="pogled-container">
    <div class="col-md-6">
        Trenutno vreme: @DateTime.Now.ToLongTimeString()
    </div>
</div>

<hr>
<div class="row">
    <div class="col-md-6">
        <span>Div ciji se sadrzaj ne menja</span> <br />
        Trenutno vreme: @DateTime.Now.ToLongTimeString()
    </div>
</div>
<hr>
<div class="row">
    <div class="col-md-6">
        <a href="#" onclick="PrikaziParcijalniPogled1()">Ajax osvezavanje</a>
        <br />
        <br />
        <a class="mt-3" asp-action="Index">Klasicno osvezavanje</a>
    </div>
</div>
```

Ajax poziv parcijalnog pogleda

```
@section Scripts{
  <script>
    async function PrikaziParcijalniPogled1() {
      try {
        const odgovor = await fetch('/Home/_Pozdrav');
        if (!odgovor.ok) {
          throw new Error('Greska u mreznom odgovoru.');
```

Klasično i ajax osvežavanje strane



Metoda _Pozdrav1, klasa HomeController

```
public PartialViewResult _Pozdrav1(string ime)
{
    ViewBag.Ime = ime;
    return PartialView();
}
```

Sadržaj parcijalnog pogleda _Pozdrav1

```
<div>  
  Vreme: @DateTime.Now.ToLongTimeString() <br />  
  Pozdrav @ViewBag.Ime !!!  
</div>
```

Index.cshtml klasa HomeController -1

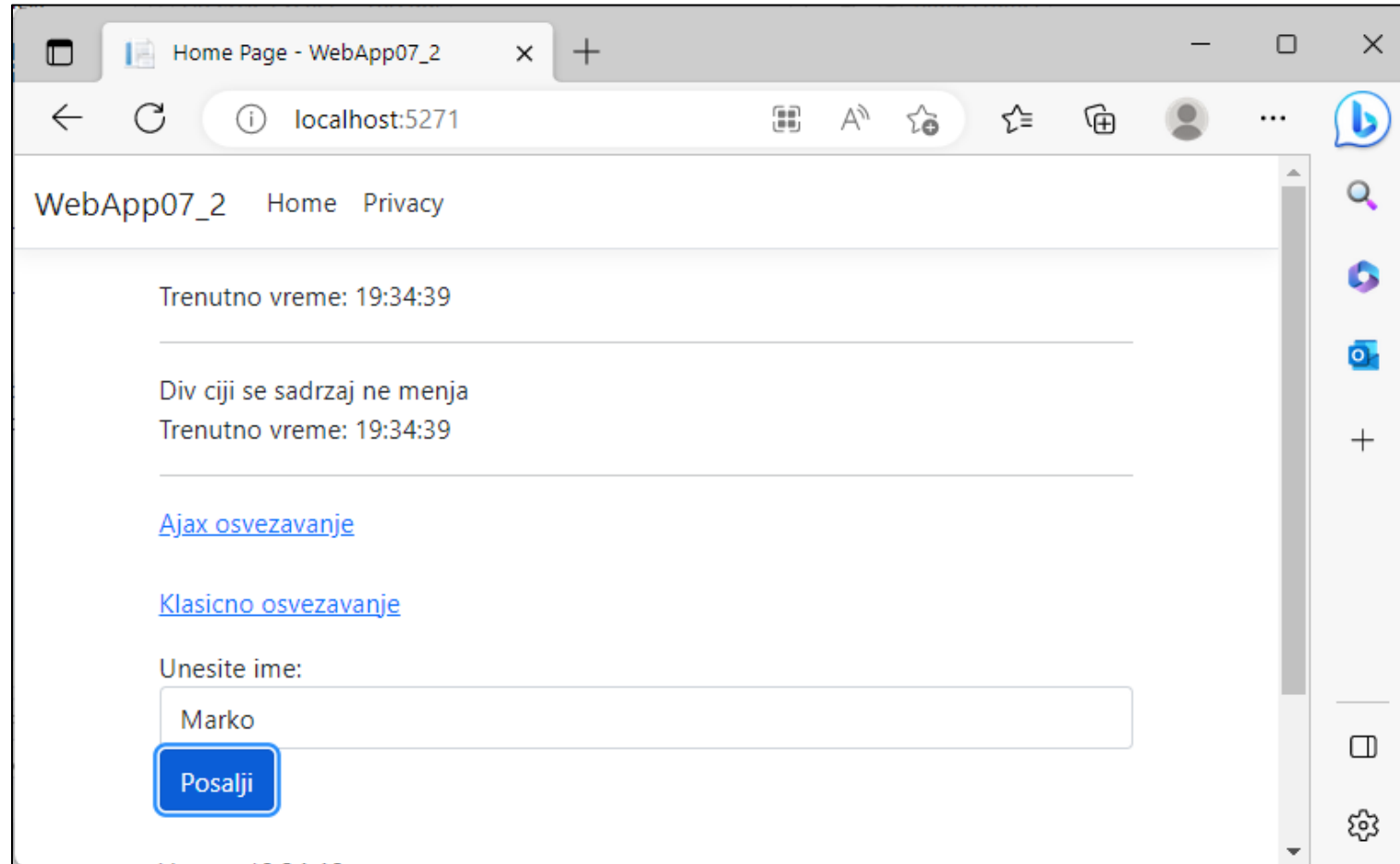
```
<div class="row mt-3">
  <div class="col-md-6">
    <form onsubmit="event.preventDefault(); PrikaziParcijalniPogled2()">
      <div class="form-group">
        <label for="Text1">Unesite ime:</label>
        <input id="Text1" type="text" name="ime" class="form-control" />

      </div>
      <input id="Submit1" type="submit" value="Posalji" class="btn btn-primary" />
    </form>
    <br />
    <div id="div3"></div>
  </div>
</div>
```

Metoda koja prikazuje parcijalni pogled

```
async function PrikaziParcijalniPogled2() {
  try {
    const ime = document.getElementById("Text1").value;
    const odgovor = await fetch(`/Home/_Pozdrav1?ime=${ime}`);
    if (!odgovor.ok) {
      throw new Error('Greska u mreznom odgovoruk.');
```

Generisanje ajax zahteva



Pitanje 1

Koji od navedenih pojmova predstavlja JavaScript API za generisanje ajax zahteva zahteva?

a) fetch api

b) ajax api

c) document api

Odgovor: a

Pitanje 2

Koji od navedenih opisa najbolje opisuje metodu `fetch()` u JavaScriptu?

- a) Metoda `fetch()` se koristi za pristupanje lokalnom skladištu pregledača
- b) Metoda `fetch()` se koristi za slanje HTTP zahteva i dobijanje HTTP odgovora u JavaScriptu
- c) Metoda `fetch()` se koristi za manipulaciju DOM elementima u JavaScriptu

Odgovor: b

Pitanje 3

Koje svojstvo objekta koji predstavlja odgovor ajax fetch zahteva postavljeno na vrednost true označava da je zahtev uspešno opslužen?

- a) ok
- b) status
- c) response

Odgovor: a

Pitanje 4

Koju vrednost treba da ima svojstvo status kod HTTP odgovora na fetch zahtev da bi se naglasilo da je uspešno obavljen zahtev?

- a) Kod 404
- b) Kod 200
- c) Kod 500

Odgovor: b

Pitanje 5

Koju metodu koristimo za čitanje odgovora na fetch zahtev koji vraća parcijalni pogled u HTML formatu?

- a) json()
- b) text()
- c) parse()

Odgovor: b