Komunikacija sa udaljenim serverom

Angular HttpClient API

- Klasa HttpClient omogućava asinhronu komunikaciju sa udaljenim serverom
- Nalazi se u paketu @angular/common/http
- **HttpClientModule** se importuje u koreni modul aplikacije
- Ovaj modul se ubacuje u imports niz glavnog modula aplikacije

```
import { HttpClientModule } from '@angular/common/http';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Klasa HttpClient

- Klasa HttpClient je tzv. injectable klasa i ubacuje se u klasu servisa posredstvom njegovog konstruktora
- Klasa HttpClient ima metode kojima se generišu http zahtevi ka serveru
- Podatke za aplikaciju najčešće obezbeđuje web api
- Metoda get() generiše GET zahtev (čitanje podataka)
- Metoda post() generiše POST zahtev (kreiranje novih podataka)
- Metoda put() generiše PUT zahtev (promena postojećih podataka)
- Metoda delete() generiše DELETE zahtev (brisanje postojećih podataka)

Kreiranje klase

ng g class osoba

```
export class Osoba {
     constructor(public id: number, public ime: string, public prezime: string, public starost: number ) {
     }
}
```

Fajl src\assets\data\osobe.json

```
[
    {"id": 1,"ime": "Marko", "prezime": "Markovic", "starost": 23},
    {"id": 2,"ime": "Petar", "prezime": "Milic", "starost": 24},
    {"id": 3,"ime": "Jovan", "prezime": "Jovic", "starost": 21},
    {"id": 4,"ime": "Milan", "prezime": "Mirkovic", "starost": 25},
    {"id": 5,"ime": "Dragan","prezime": "Petrovic", "starost": 28}
]
```

Simulacija udaljenog web api-ja

Kreiranje servisa

```
ng g service osoba
```

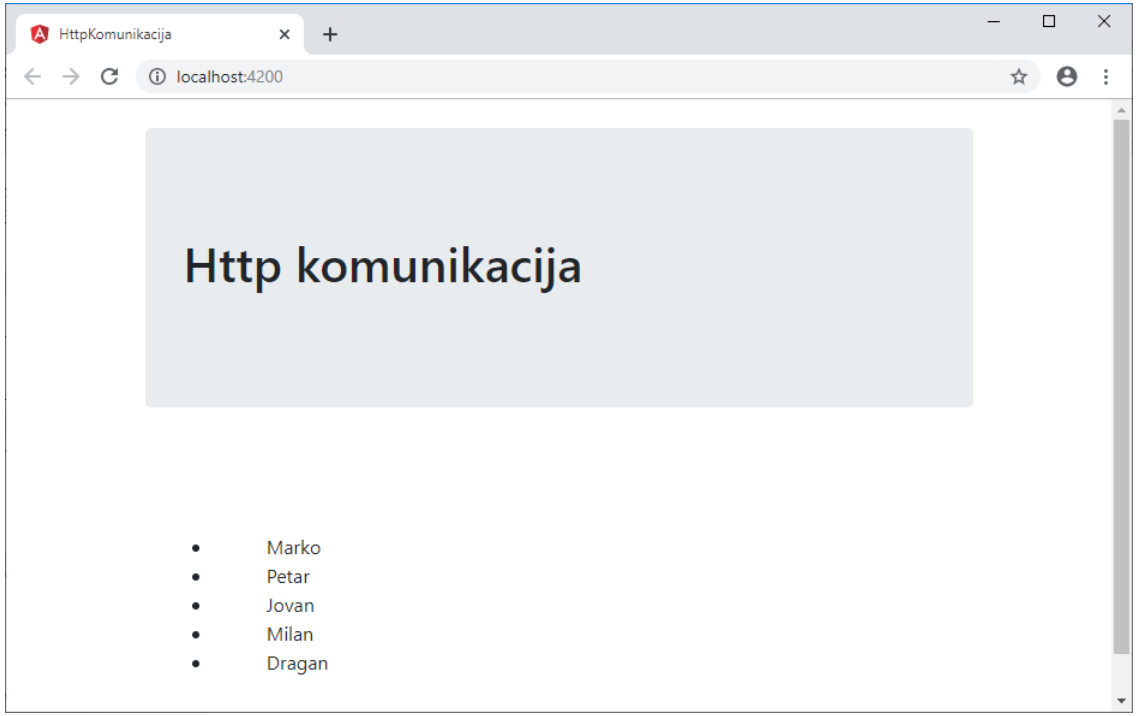
```
import { Injectable } from '@angular/core';
import { Osoba } from './osoba';
import { Observable } from 'rxjs';
import { HttpClient } from '@angular/common/http';

@Injectable({
  providedIn: 'root'
})
export class OsobaService {

  constructor(private http: HttpClient) { }
  adresa = '/assets/data/osobe.json';

  vratiOsobe(): Observable<Osoba[]> {
    return this.http.get<Osoba[]>(this.adresa);
  }
}
```

Glavna komponenta – pristup podacima posredstvom servisa

```
export class AppComponent implements OnInit {
  title = 'HttpKomunikacija';

  osobe: Osoba[];
  constructor(private oServis: OsobaService) {
  }
  ngOnInit(): void {
      this.oServis.vratiOsobe()
        .subscribe(os => this.osobe = os);
    }
}
```

Šablon glavne komponente

```
<div class="container">
    <div class="jumbotron">
      <h1>Http komunikacija</h1>
    </div>
    <div class="row">
      <div class="col-md-6">

        <ul>
          <li *ngFor="let osoba of osobe">
            {{osoba.ime}}
          </li>
        </ul>
      </div>
    </div>
  </div>
```



Obrada grešaka pri http komunikaciji -1

```
import { HttpClient, HttpErrorResponse } from '@angular/common/http';
import { Observable, throwError } from 'rxjs';
import { catchError } from 'rxjs/operators';
```

```
vratiOsobe(): Observable<Osoba[]>{
    return this.http.get<Osoba[]>(this.adresa)
    .pipe(catchError(this.errorHandler));

  }

  errorHandler(error: HttpErrorResponse){
    return throwError(error.message || "Server Error");
  }
```

pipe() funkcija kao argumente uzima funkcije koje treba da kombinuje
catchError () funkacija hvata greške

Obrada grešaka pri http komunikaciji -2

```
export class AppComponent implements OnInit {
  title = 'HttpKomunikacija';

  osobe: Osoba[];
  constructor(private oServis: OsobaService) {
  }
  ngOnInit(): void {
      this.oServis.vratiOsobe()
        .subscribe(
          os => this.osobe = os,
          grr => console.log( 'Greska: ' + grr)
          );
    }
}
```

Angular In-Memory Web API

- Ovaj modul simulira REST (REpresentational State Transfer) API back-end

```
npm i angular-in-memory-web-api
```

Klasa Osoba

```
ng g class osoba
```

```
export class Osoba {
    constructor(public id: number, public ime: string, public prezime: string,
     public starost: number ) {
    }
}
```

Kreiranje baze podataka u memoriji

ng g class baza

```
import { InMemoryDbService } from 'angular-in-memory-web-api';
import { Osoba } from './osoba';

export class Baza implements InMemoryDbService {
    createDb() {
      const osobe: Osoba[] = [
                new Osoba(1, 'Marko', 'Markovic', 25),
                new Osoba(2, 'Petar', 'Petrovic', 34),
                new Osoba(3, 'Jovan', 'Jovanovic', 27),
                new Osoba(4, 'Milan', 'Mirkovic', 22)
              ];

      return {osobe};
    }
}
```

'api/osobe' adresa web api -ja

Konfigurisanje glavnog modula aplikacije

```
import { Baza } from './baza';
import { InMemoryWebApiModule } from 'angular-in-memory-web-api';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule,
    InMemoryWebApiModule.forRoot(Baza)
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Kreiranje servisa za komunikaciju sa api -jem

```
import { Injectable } from '@angular/core';
import { HttpClient, HttpErrorResponse} from '@angular/common/http';
import { Observable, throwError } from 'rxjs';
import { Osoba } from './osoba';
import { catchError } from 'rxjs/operators';

@Injectable({
  providedIn: 'root'
})
export class OsobaService {

  constructor(private http: HttpClient) { }

  private apiUrl = '/api/osobe';

  private errorHandler(error: HttpErrorResponse) {
    return throwError(error.message || 'Server eror');
  }
```

GET metode servisa za komunikaciju sa web api aplikacijom

```
vratiOsobe(): Observable<Osoba[]> {
    return this.http.get<Osoba[]>(this.apiUrl)
      .pipe(catchError(this.errorHandler));
  }
```

```
vratiOsobu(id: number): Observable<Osoba> {
    const url = `${this.apiUrl}/${id}`;
    return this.http.get<Osoba>(url).pipe(
      catchError(this.errorHandler)
    );
  }
```

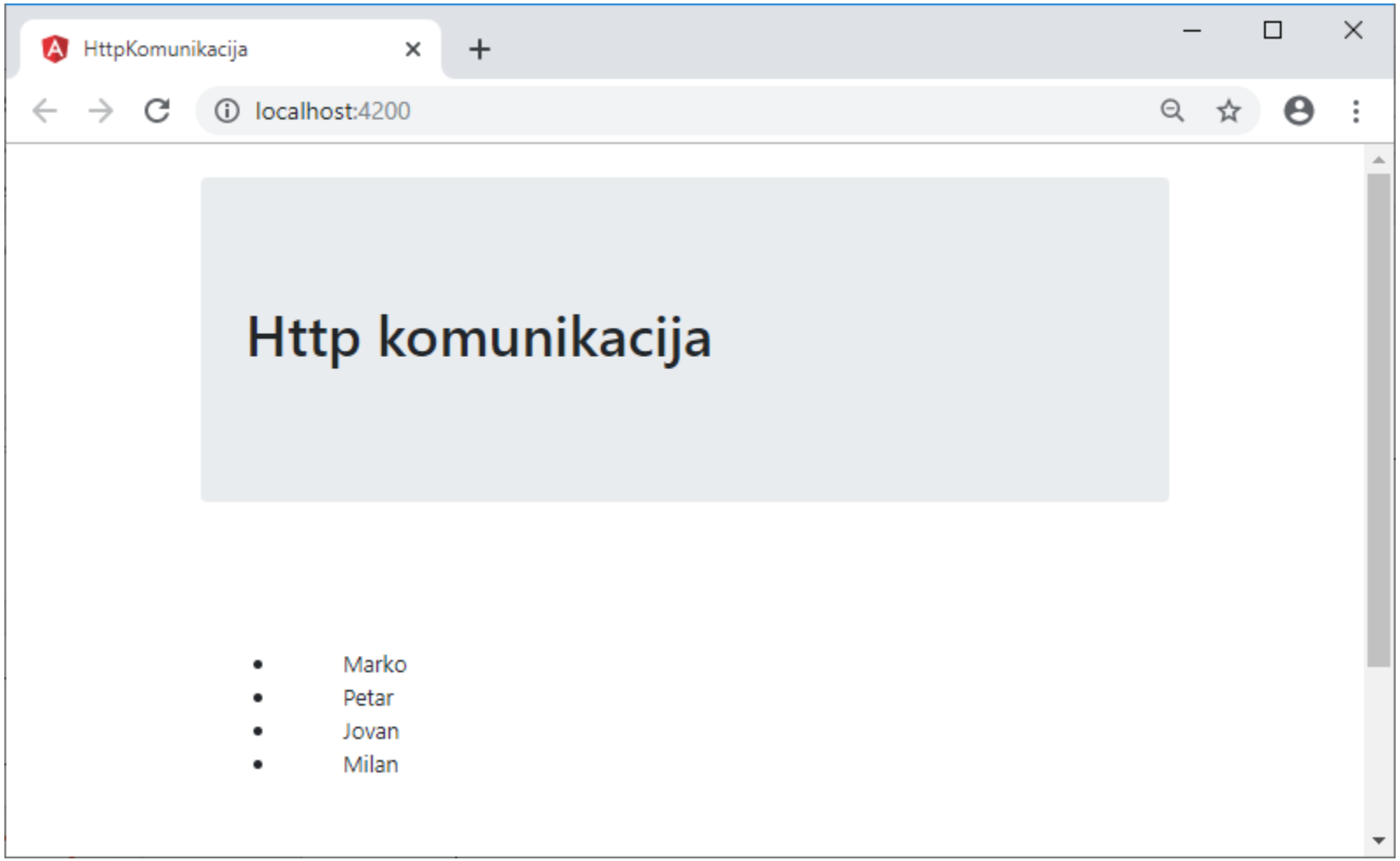
Glavna komponenta – poziv get metode

```
export class AppComponent implements OnInit {
  title = 'HttpKomunikacija';
  osobe: Osoba[];
  odabranaOsoba: Osoba;
  idOsobe =4;


  constructor(private oServis: OsobaService) {
  }

  prikaziOsobe(): void {
    this.oServis.vratiOsobe()
      .subscribe(
        os => this.osobe = os,
        grr => console.log('Greska: ' + grr)
      );
  }
  ngOnInit(): void {
    this.prikaziOsobe();
  }
```

Šablon glavne komponente –prikaz podataka

```
<div class="container">
    <div class="jumbotron">
      <h1>Http komunikacija</h1>
    </div>
    <div class="row">
      <div class="col-md-6">
        <ul>
          <li *ngFor="let osoba of osobe">
            {{osoba.ime}}
          </li>
        </ul>
      </div>
    </div>
 </div>
```

Korisnički interfejs



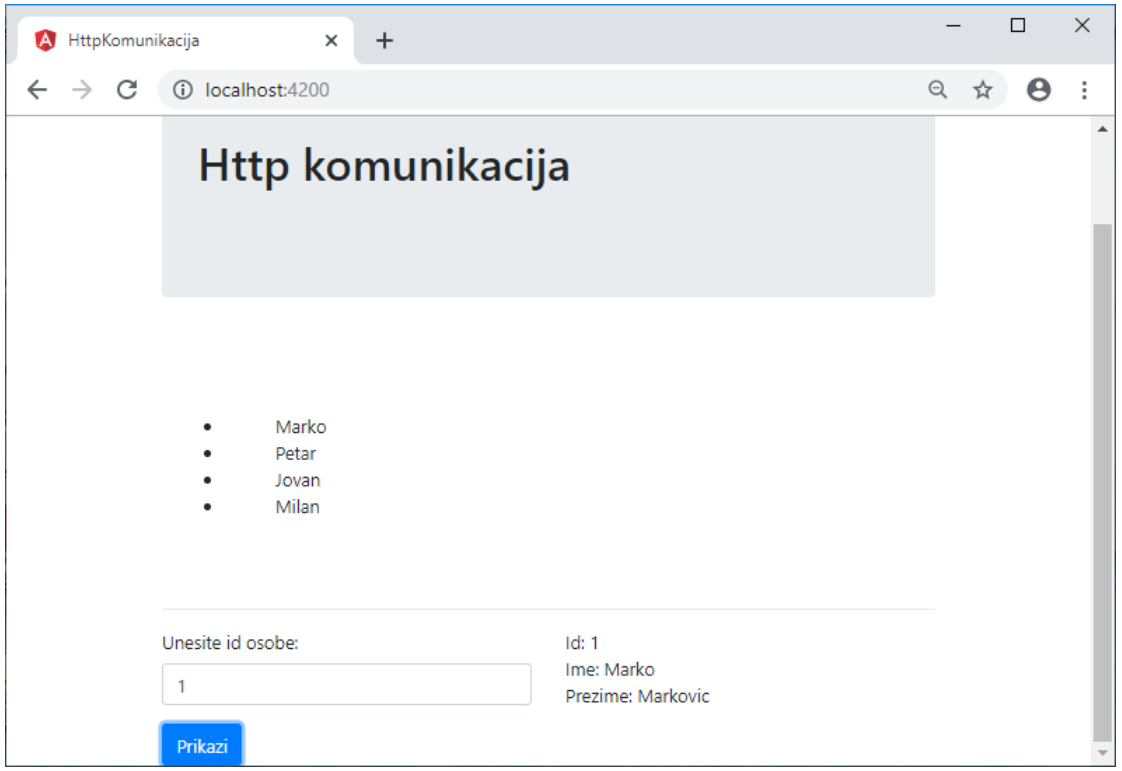
Glavna komponenta – funkcija pronadji()

```
pronadji(id: number): void {
    this.oServis.vratiOsobu(id).subscribe(
      os => this.odabranaOsoba = os,
      grr => {
        console.log('Greska: ' + grr);
        this.odabranaOsoba = null;
      }
    );
  }
```

Glavna komponenta – UI za pretragu

```
<div class=row>
       <div class="col-md-6">
           <div class="form-group">
              <label for="">Unesite id osobe:</label>
              <input type="text" #id class="form-control">
           </div>

            <button (click) = "pronadji(id.value)" class="btn btn-primary">Prikazi</button>
        </div>
     <div class="col-md-6" *ngIf="odabranaOsoba">
        Id:  {{odabranaOsoba.id}} <br>
        Ime: {{odabranaOsoba.ime}} <br>
        Prezime:  {{odabranaOsoba.prezime}}
     </div>

</div>
```

UI za pretragu



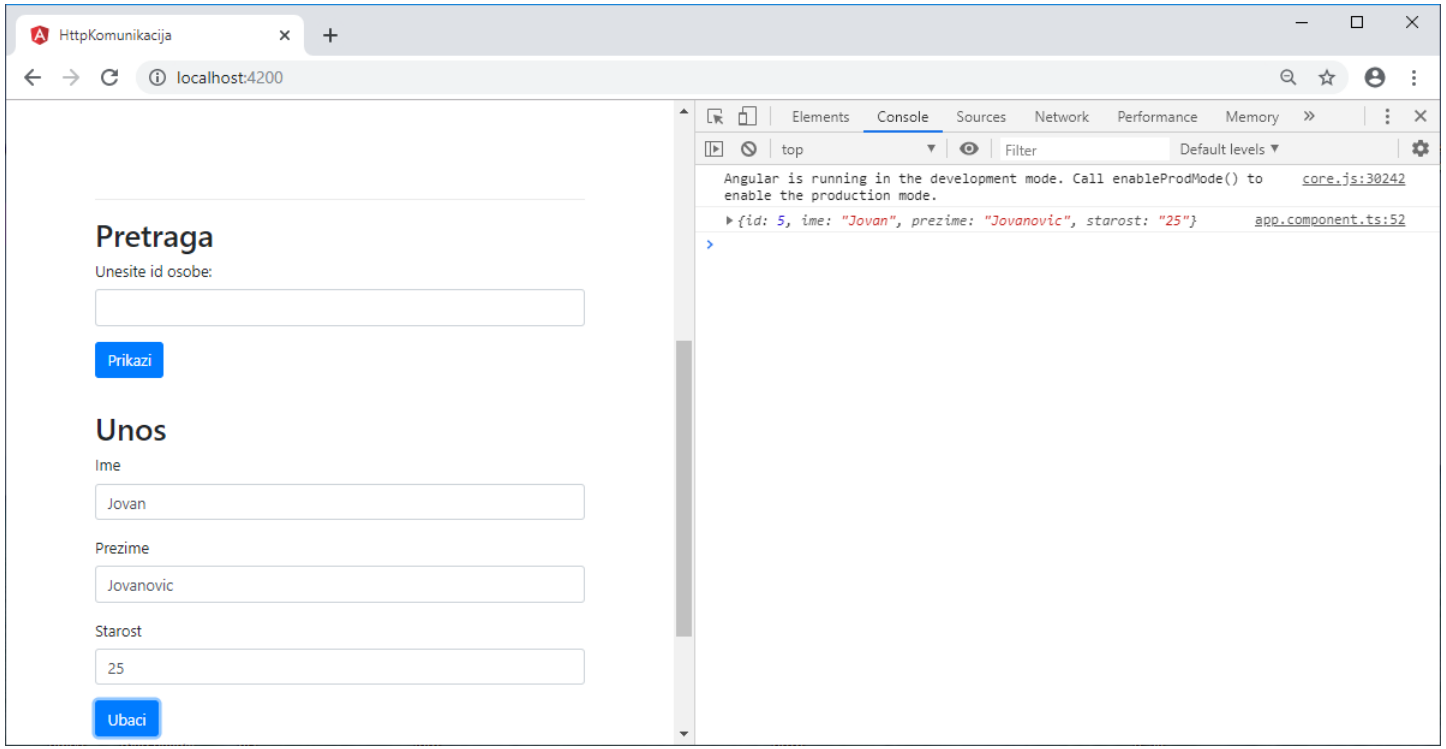
POST metoda servisa za komunikaciju sa web api aplikacijom

```
ubaciOsobu(os: Osoba): Observable<Osoba> {
    return this.http.post<Osoba>(this.apiUrl, os).pipe(
      catchError(this.errorHandler)
    );
  }
```

Interfejs za unos podataka

```
<div class="row">
      <div class="col-md-6">
      <h2>Unos</h2>
         <div class="form-group">
            <label for="">Ime</label>
            <input type="text" name="" id="" #ime class="form-control">
         </div>
         <div class="form-group">
            <label for="">Prezime</label>
            <input type="text" name="" id="" #prezime class="form-control">
         </div>

         <div class="form-group">
            <label for="">Starost</label>
            <input type="number" name="" id="" #starost class="form-control">
         </div>

         <button (click) = "ubaciOsobu(ime.value, prezime.value, starost.value)"
         class="btn btn-primary">Ubaci</button>
      </div>
</div>
```

Interfejs za unos podataka



Glavna komponenta – unos podataka

```
ubaciOsobu(ime: string, prezime: string, starost: number): void {

   this.idOsobe++;

   const os1 = new Osoba(this.idOsobe, ime, prezime, starost);

   this.oServis.ubaciOsobu(os1).subscribe(
      podaci => {
        console.log(podaci);
        this.prikaziOsobe();
      },
      gr => console.log(gr)
    );
  }
```

DELETE metoda servisa za komunikaciju sa web api aplikacijom

```
obrisiOsobu(id: number): Observable<{}> {
  const url = `${this.apiUrl}/${id}`;
  return this.http.delete<Osoba>(url).pipe(
      map(() => id),
      catchError(this.errorHandler),
    );
  }
```

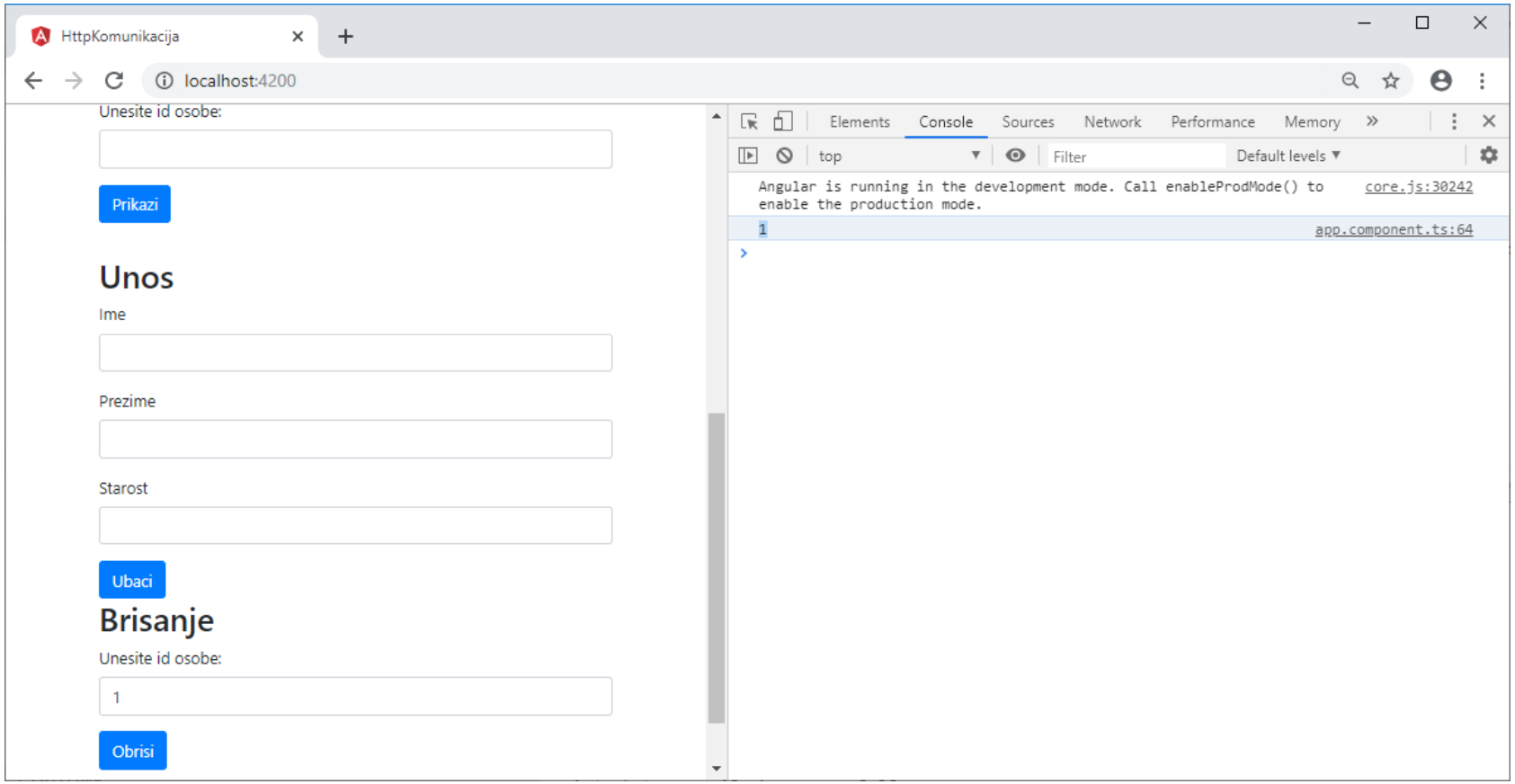
Glavna komponenta – brisanje podataka

```
obrisi(id: number)  {
    this.oServis.obrisiOsobu(id).subscribe(
      os => {
        console.log(os);
        this.prikaziOsobe();
      },
      grr => console.log('Greska: ' + grr)
      );
  }
```

Interfejs za brisanje

```
 <div class="row">
      <div class="col-md-6">
         <h2>Brisanje</h2>
            <div class="form-group">
               <label for="">Unesite id osobe:</label>
               <input type="text" #idb class="form-control">
            </div>

             <button (click) = "obrisi(idb.value)"
             class="btn btn-primary">Obrisi</button>
         </div>
</div>
```

Interfejs za brisanje



PUT metoda servisa za komunikaciju sa web api aplikacijom

```
promeniOsobu(os: Osoba): Observable<Osoba> {
  const url = `${this.apiUrl}`;
  return this.http.put<Osoba>(url, os)
  .pipe(
    map(() => os),
    catchError(this.errorHandler)
  );
}
```

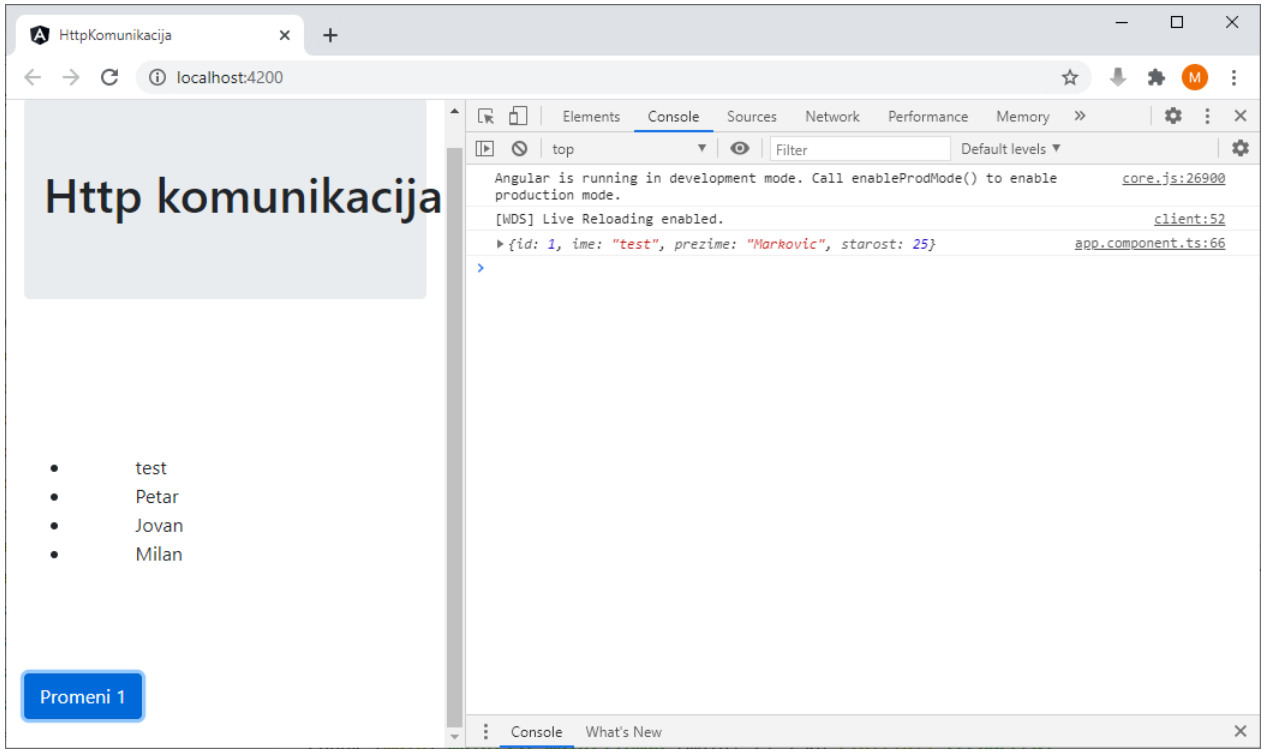
Glavna komponenta – promena podataka

```
promeni() {
    const os1 = this.osobe.find(x => x.id == 1);
    os1.ime = 'test';
    this.oServis.promeniOsobu(os1).subscribe(
      os => console.log(os),
      grr => console.log('Greska: ' + grr)
    );
  }
```

Promena podataka

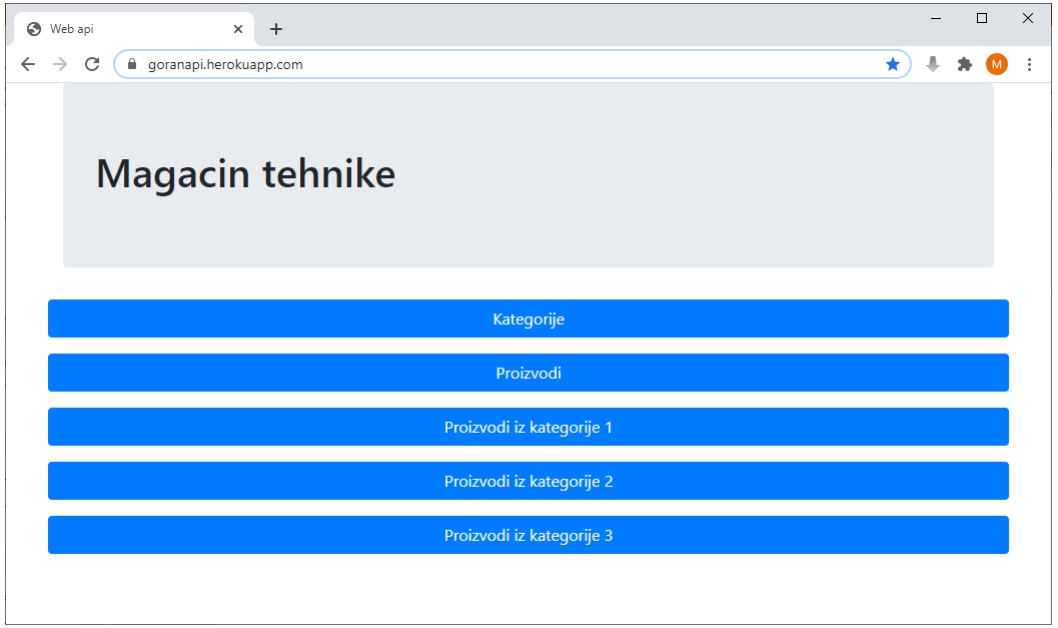
```
<div class="row">
        <div class="col-md-6">
            <button (click)="promeni()" class="btn btn-primary">Promeni 1</button>
        </div>
</div>
```

Promena podataka



Primer web api -aplikacije

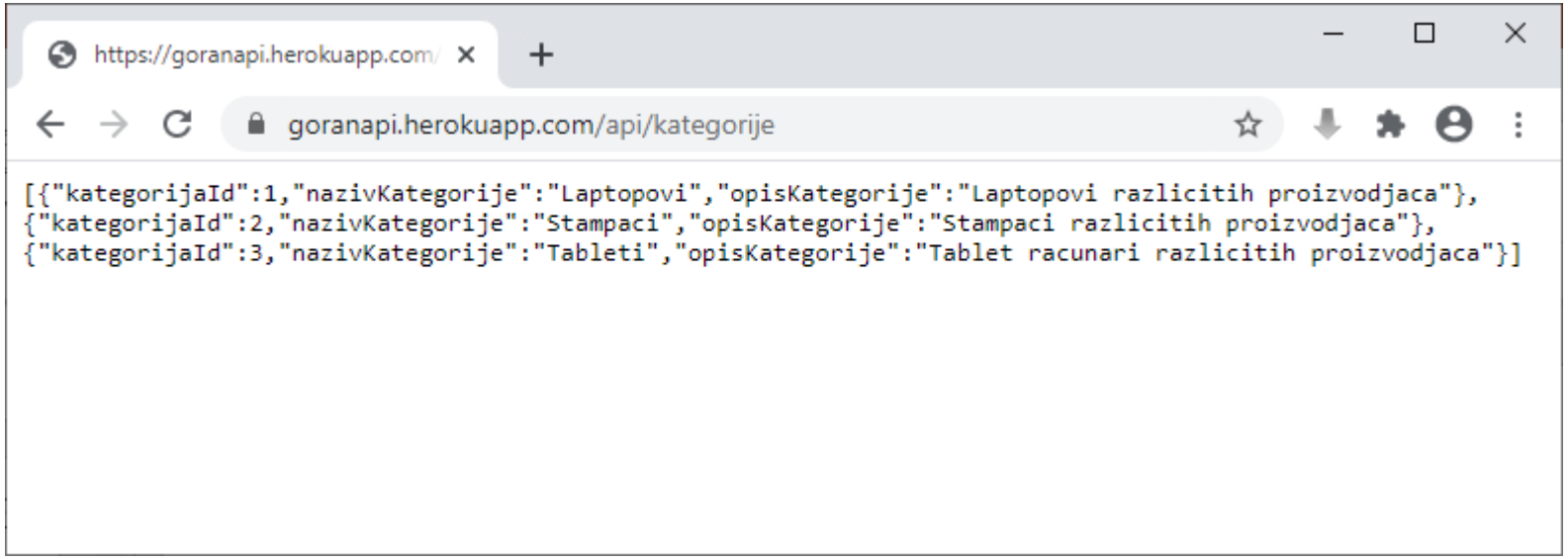
https://goranapi.herokuapp.com/



web api dozvoljava CORS zahteve

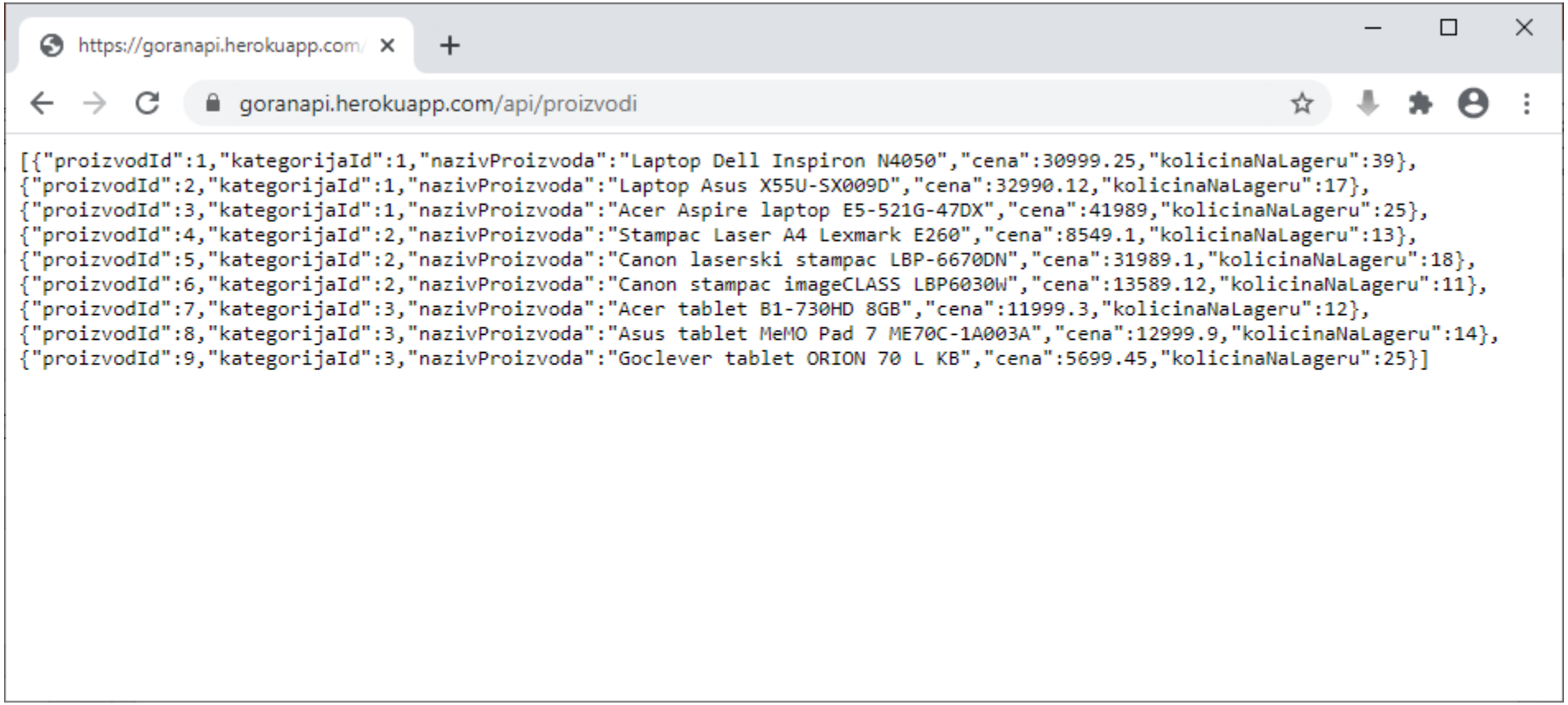
Prikaz kategorija

https://goranapi.herokuapp.com/api/kategorije



Prikaz proizvoda

https://goranapi.herokuapp.com/api/proizvodi




```
[{"proizvodId":1,"kategorijaId":1,"nazivProizvoda":"Laptop Dell Inspiron N4050","cena":30999.25,"kolicinaNaLageru":39},
{"proizvodId":2,"kategorijaId":1,"nazivProizvoda":"Laptop Asus X55U-SX009D","cena":32990.12,"kolicinaNaLageru":17},
{"proizvodId":3,"kategorijaId":1,"nazivProizvoda":"Acer Aspire laptop E5-521G-47DX","cena":41989,"kolicinaNaLageru":25},
{"proizvodId":4,"kategorijaId":2,"nazivProizvoda":"Stampac Laser A4 Lexmark E260","cena":8549.1,"kolicinaNaLageru":13},
{"proizvodId":5,"kategorijaId":2,"nazivProizvoda":"Canon laserski stampac LBP-6670DN","cena":31989.1,"kolicinaNaLageru":18},
{"proizvodId":6,"kategorijaId":2,"nazivProizvoda":"Canon stampac imageCLASS LBP6030W","cena":13589.12,"kolicinaNaLageru":11},
{"proizvodId":7,"kategorijaId":3,"nazivProizvoda":"Acer tablet B1-730HD 8GB","cena":11999.3,"kolicinaNaLageru":12},
{"proizvodId":8,"kategorijaId":3,"nazivProizvoda":"Asus tablet MeMO Pad 7 ME70C-1A003A","cena":12999.9,"kolicinaNaLageru":14},
{"proizvodId":9,"kategorijaId":3,"nazivProizvoda":"Goclever tablet ORION 70 L KB","cena":5699.45,"kolicinaNaLageru":25}]
```

Klasa Proizvod

```
export class Proizvod {
    constructor(public proizvodId: number, public kategorijaId: number,
 public nazivProizvoda: string, public cena: number,
 public kolicinaNaLageru: number ) {
    }
}
```

Klasa MagacinService

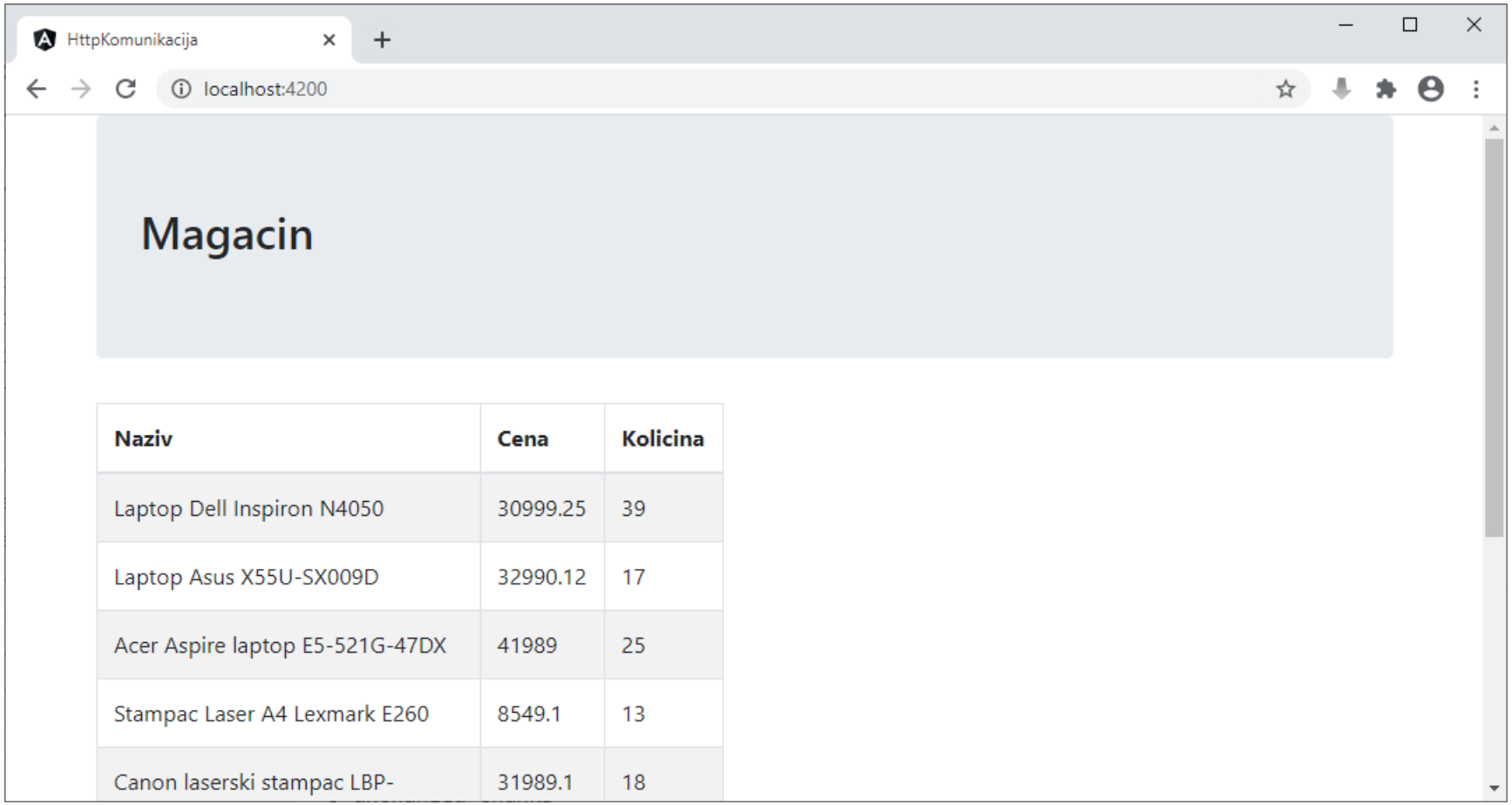
```
export class MagacinService {
  apiUrl = 'https://goranapi.herokuapp.com/';

  constructor(private http: HttpClient) { }

  private errorHandler(error: HttpErrorResponse) {
        return throwError(error.message || 'Server eror');
      }

  vratiProizvode(): Observable<Proizvod[]> {
    return this.http.get<Proizvod[]>(this.apiUrl + 'api/proizvodi' )
     .pipe(catchError(this.errorHandler));
  }
}
```

Glavna komponenta

```
export class AppComponent {

  naziv= 'Magacin';
  proizvodi:Proizvod[];

    constructor(private mServis:MagacinService) {
    }
    prikaziProizvode(): void {
      this.mServis.vratiProizvode()
        .subscribe(
          p=> this.proizvodi=p,
          grr => console.log('Greska: ' + grr)
        );
    }
    ngOnInit(): void {
      this.prikaziProizvode();
    }
}
```

```
<div class="container">
    <div class="jumbotron">
        <h2>{{naziv}}</h2>
    </div>
    <div class="row">
        <div class="col-md-6">
            <table class="table table-bordered table-striped">
                <thead>
                    <tr>
                        <th>Naziv</th>
                        <th>Cena</th>
                        <th>Kolicina</th>
                    </tr>
                </thead>
                <tbody>
                    <tr *ngFor="let proizvod of proizvodi">
                        <td>{{proizvod.nazivProizvoda}}</td>
                        <td>{{proizvod.cena}}</td>
                        <td>{{proizvod.kolicinaNaLageru}}</td>
                    </tr>
                </tbody>
            </table>
        </div>
    </div>
</div>
```

Prikaz podataka



Naziv	Cena	Kolicina
Laptop Dell Inspiron N4050	30999.25	39
Laptop Asus X55U-SX009D	32990.12	17
Acer Aspire laptop E5-521G-47DX	41989	25
Stampac Laser A4 Lexmark E260	8549.1	13
Canon laserski stampac LBP-	31989.1	18

Pitanje 1

Klasa koja omogućava komunikaciju sa udaljenim api-jem naziva se:

a. HttpClient
b. HttpModule
c. HttpConnect

Odgovor: a

Pitanje 2

Promena podataka na serveru vrši se korišćenjem sledeće metode HttpClient objekta:

a. get
b. put
c. update

Odgovor: b

Pitanje 3

Da bi se koristila klasa HttpClient za komunikaciju sa udaljenim serverom u glavni modul aplikacije potrebno je importovati sledeći modul:

a. RemoteModule
b. ClientModule
c. HttpClientModule

Odgovor: c