Html forme

Element <form>

- Element <form> omogućava unos podataka od strane korisnika
- Atribut action elementa <form> specificira url adresu strane (ili metodu klase) kojoj će podaci sa forme biti poslati
- Atribut method specificira kako će podaci biti poslati
  - GET metoda je podrazumevana opcija
  - POST je preferirana metoda
- Atribut enctype specificira na koji način se enkoduju podaci sa forme pre nego što se pošalju serveru

Kontrole forme - element <input>

- Element <input> je osnovni element za unos podataka
- Ima više različitih oblika u zavisnosti od atributa **type**
  - type="text"
  - type="password"
  - type="hidden"
  - type="checkbox"
  - type="radio"
  - type="reset"
  - type="submit"
  - type="image"
  - type="button"
  - type="file"
  - type="email"

Atributi <input> elementa

- Atribut **id** koristi se za pristup elementu iz CSS-a ili javascripta
  - Atribut id mora biti jedinstven na formi
- Atribut **value** postavlja podrazumevanu vrednost za numeričke i tekstualne kontrole za unos podataka
- Atribut **name** se koristi od strane servera da se referenciraju polja forme kada se ona submituje, takođe se koristi za pristup elementu iz klijentskog koda
- Atribut **placeholder** opisuje očekivanu vrednost unutar nekog text polja
- Atribut **required** ukazuje da je polje obavezno i da se forma ne može submitovati ukoliko je polje prazno

Text polje

- Element <input type="text"> kreira polje za unos teksta
- Svojstvo **value** služi za iščitavanje unetog teksta iz javascript koda

```
<input type="text" name="ime" id="textIme">
```

Element <**label**>

- Element <**label**> element služi za pridruživanje teksta ulaznoj kontroli, na osnovu id atributa

```
<label for="textIme">Ime:</label> <br>
<input type="text" name="ime" id="textIme">
```

Element **<fieldset>**

- Element **<fieldset>** je opcioni element definiše grupu kontrola na formi
- Unutar <fieldset> elementa se može naći <legend> element i tada on mora biti prvi element unutar <fieldset> elementa

Element button

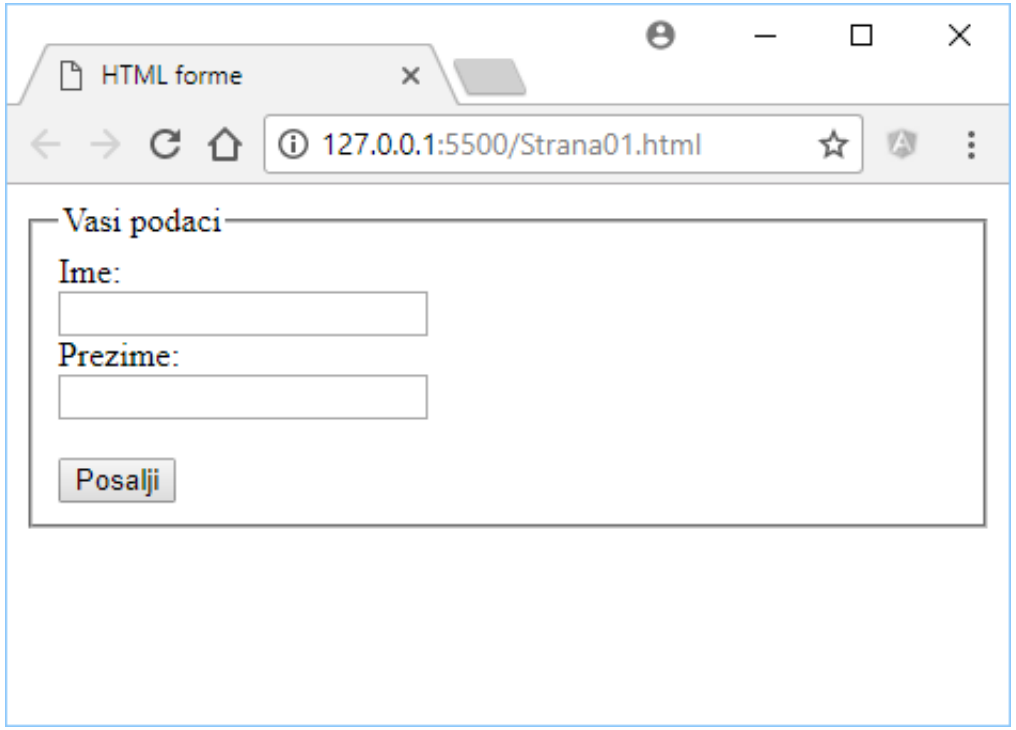
- Element **<button>** se koristi za definisanje dugmeta, podrazumevani tip dugmeta je submit dugme
- Atribut type ima vrednosti button, reset i submit

```
<button type="submit">Posalji</button>
```

Primer HTML forme sa upotrebom elementa fieldset

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>HTML forme</title>
</head>
<body>
    <form action="/strana02.html" method="get" >
        <fieldset>
            <legend>
                Vasi podaci
            </legend>
            <label for="textIme">Ime:</label> <br>
            <input type="text" name="ime" id="textIme"> <br>

            <label for="textPrezime">Prezime:</label> <br>
            <input type="text" name="prezime" id="textPrezime"> <br>
            <br>

            <button type="submit">Posalji</button>
        </fieldset>
    </form>
</body>
</html>
```

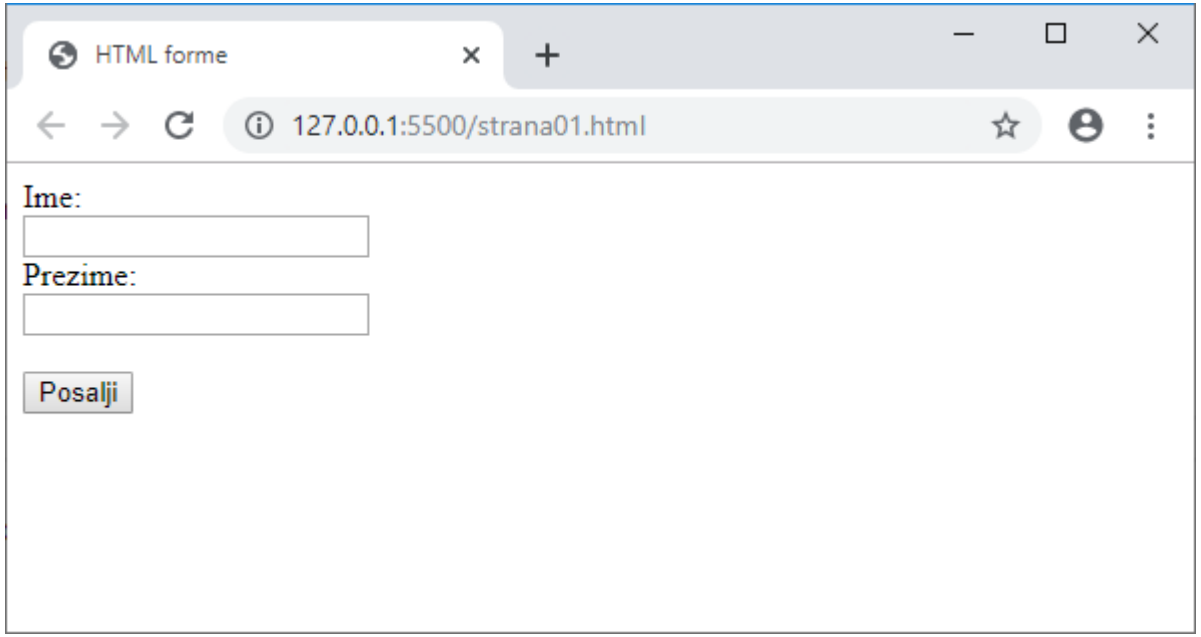
Primer HTML forme sa upotrebom elementa fieldset



Forma bez upotrebe fieldset elementa

```
<form action="/strana02.html" method="get">
        <label for="textIme">Ime:</label> <br>
        <input type="text" name="ime" id="textIme"> <br>

        <label for="textPrezime">Prezime:</label> <br>
        <input type="text" name="prezime" id="textPrezime"> <br>
        <br>

        <button type="submit">Posalji</button>
</form>
```

Forma bez upotrebe fieldset elementa

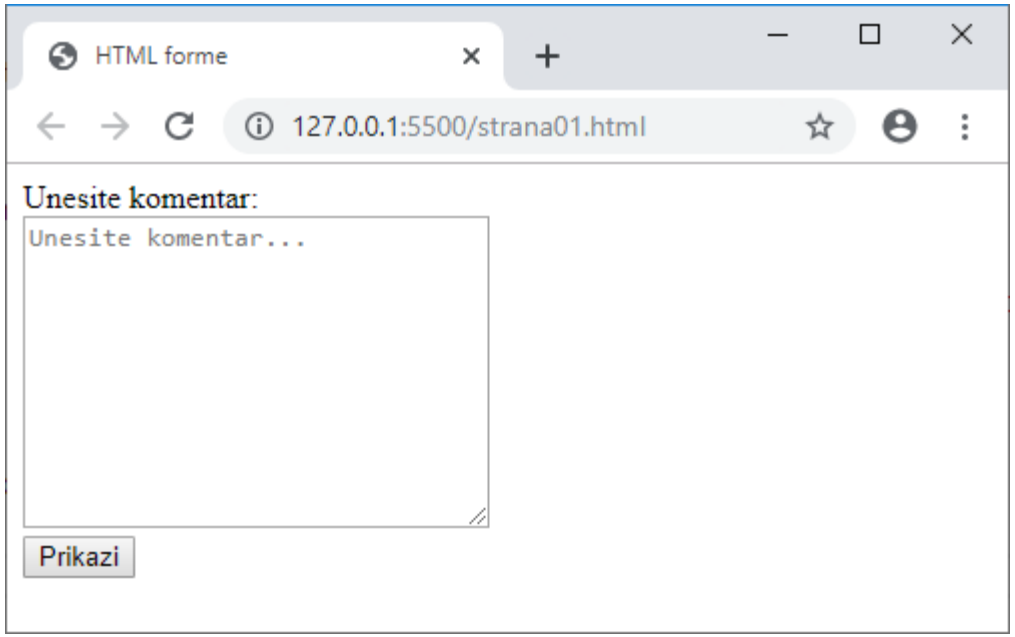


Element <textarea>

- Element **<textarea>** predstavlja multiline polje za unos teksta

```
<body>
    <form action="">
        <label for="TextArea1">Unesite komentar:</label> <br>
        <textarea name="TextArea1" id="TextArea1" cols="30" rows="10"
        placeholder="Unesite komentar..."></textarea>
        <br>
        <input type="button" value="Prikazi">
    </form>
</body>
```

Prikaz tekst oblasti

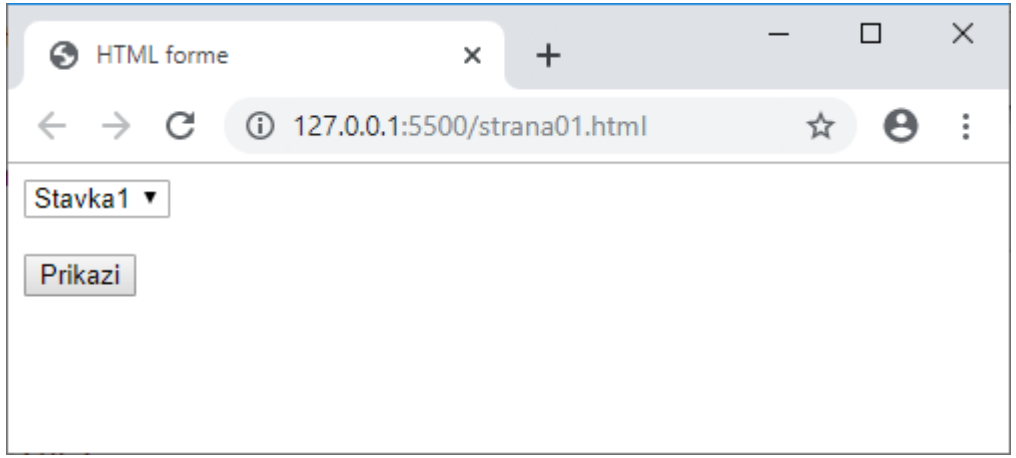


Element <select>

- Element **<select>** se koristi za definisanje drop-down liste ili listbox kontrole

```
<form action="">
        <select name="Select1" id="Select1">
            <option value="1">Stavka1</option>
            <option value="2">Stavka2</option>
            <option value="3">Stavka3</option>
        </select>
        <br>
        <br>
        <input type="button" value="Prikazi">
 </form>
```

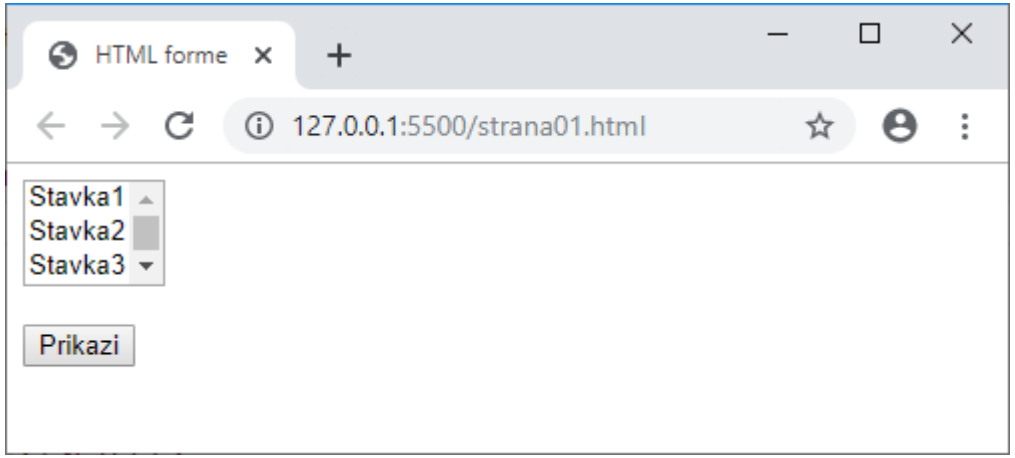
Element <select>



Element select sa atributom size

```
<form action="">
    <select name="Select1" id="Select1" size="3">
        <option value="1">Stavka1</option>
        <option value="2">Stavka2</option>
        <option value="3">Stavka3</option>
        <option value="4">Stavka4</option>
    </select>
    <br>
    <input type="button" value="Prikazi">
</form>
```

Element select sa atributom size



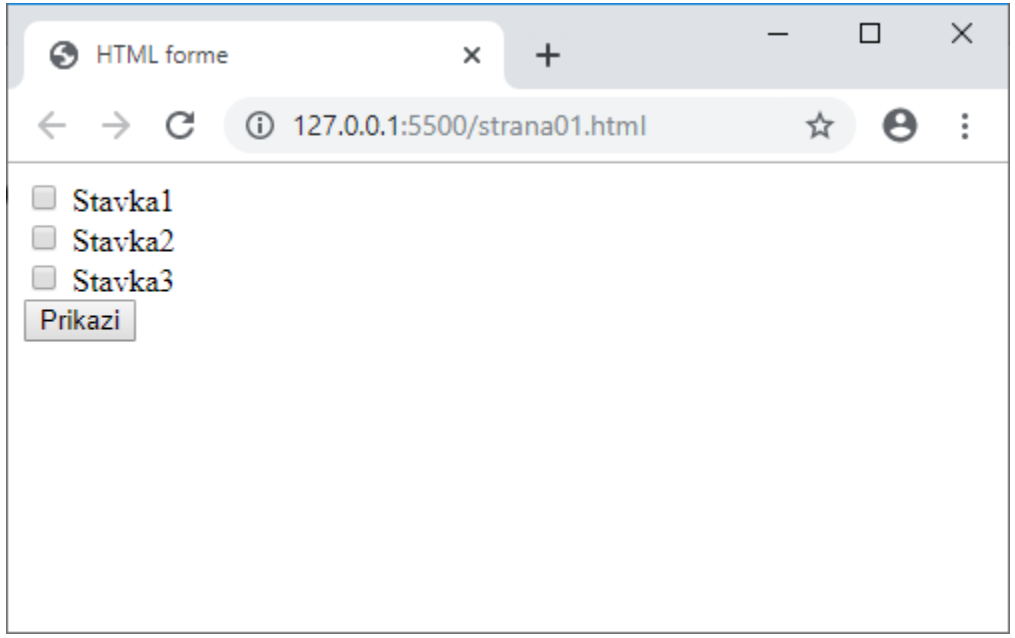
Kontrole checkbox i radio button

- Element <input type="checkbox"> kreira checkbox
- Element <input type="radio"> kreira radio button
- Atribut checked vraća logičku vrednost

Html checkbox

```
<body>
    <form action="">

        <input id="CheckBox1" type="checkbox" name="CheckBox1">
        <label for="CheckBox1">Stavka1</label>
        <br />

        <input id="CheckBox2" type="checkbox" name="CheckBox2">
        <label for="CheckBox2">Stavka2</label>
        <br />

        <input id="CheckBox3" type="checkbox" name="CheckBox3">
        <label for="CheckBox3">Stavka3</label>
        <br />

        <input id="Button1" type="button" value="Prikazi>
    </form>
</body>
```

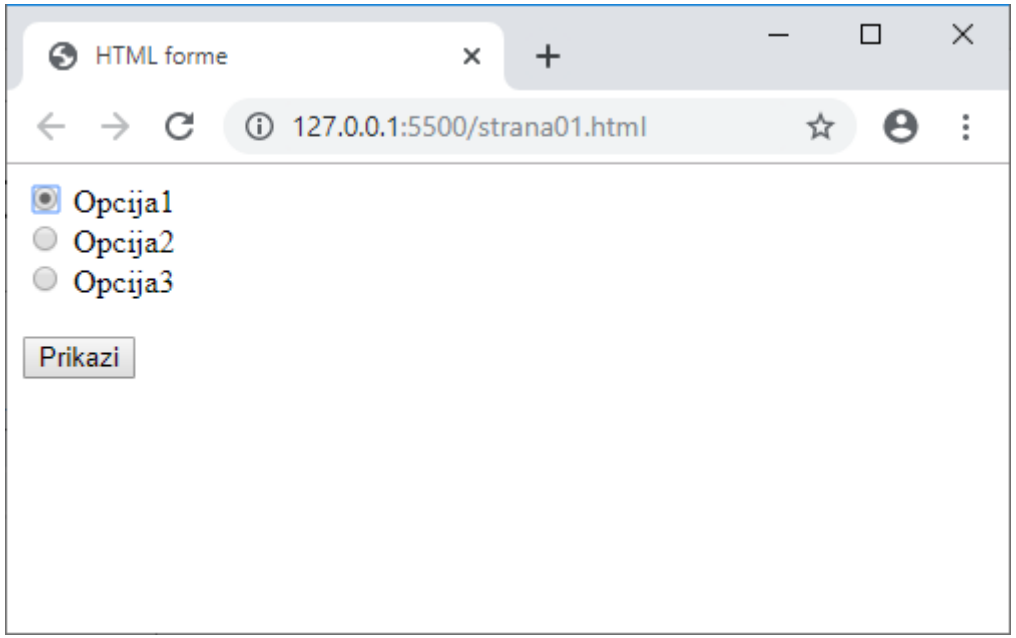
Prikaz CheckBoxova



Grupa radio buttona, svojstvo name

```
<body>
<form action="" method="get">
    <input type="radio" name="opcije" id="Radio1" value="v1">
    <label for="Radio1">Opcija1</label> <br>

    <input type="radio" name="opcije" id="Radio2" value="v2">
    <label for="Radio2">Opcija2</label> <br>

    <input type="radio" name="opcije" id="Radio3" value="v3">
    <label for="Radio3">Opcija3</label> <br><br>
    <input type="button" value="Prikazi">
</form>
</body>
```

Prikaz radio button kontrola



Angular forme

Forma

- Forma se koristi za prikupljanje podataka od korisnika
- Angular koristi dva tipa formi:
  - Template-driven forme (TDF)
  - Reaktivne forme (model driven form)
- Klasom **FormControl** predstavlja se ulazno polje angular forme
  - Instanca klasa FormControl kreira se za svako polje forme kod TDF
- Klasom **FormGroup** se predstavlja kolekcija kontrola forme

FormControl klasa

- Polje forme možemo kreirati i iz koda:
  let ime= new FormControl();
- Svojstvo **value** vraća trenutni sadržaj ovog polja (ime.value)
- Svojstvo **errors** vraća kolekciju grešaka
- Svojstvo **dirty** vraća true ukoliko se vrednost polja promenila
- Svojstvo **touched** vraća true ukoliko je polje dodirnuto
- Svojstvo **valid** vraća true ukoliko je polje prošlo validaciju

FormGroup klasa

```
let adresa= new FormGroup({
    ulica : new FormControl(""),
    grad : new FormControl(""),
    postanskiBroj : new FormControl("")
});
```

- Omogućava upravljanje grupom kontrola forme
- adresa.grad je način da se pristupi child kontroli grad
- svojstvo value vraća JSON objekat
- svojstvo errors vraća listu grešaka svih kontrola
- svojstvo valid vraća true ukoliko su sve kontrole grupe prošle validaciju

Template Driven Forms (TDF)

- Importuje se modul **FormsModule** iz biblioteke @angular/forms u glavni modul aplikacije
- Kada se uključi FormsModule angular svim forms elementima automatski dodaje **ngForm** direktivu
- Direktiva **ngForm** radi sledeće:
  - automatski kreira instancu **FormGroup** koja odgovara kompletnoj formi
  - kreira instancu klase **FormControl** za svaku kontrolu sa **ngModel** direktivom
- Instanci ngForm i svakoj instanci FormControl u šablonu možemo dodeliti lokalnu promenljivu
- Koristimo događaj **ngSubmit** da pošaljemo klasi komponente podatke sa forme

Glavni modul aplikacije

```
import { FormsModule } from "@angular/forms";

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

Šablon komponente Forma1

```
  <form>
    <div class="form-group">
        <label for="ime">Ime</label>
        <input type="text" name="ime" id="ime" class="form-control" />
    </div>

    <div class="form-group">
        <label for="prezime">Prezime</label>
        <input type="text" name="prezime" id="prezime" class="form-control" />
    </div>

    <div class="form-group">
        <button type="submit" class="btn btn-primary">Posalji</button>
    </div>
</form>
```

Klasa komponente

```
export class AppComponent {
  title = 'forme';

  onSubmit(forma: NgForm) {
        console.log('Ime',forma.controls['ime'].value);
        console.log('Prezime',forma.controls['prezime'].value);
        console.log('JSON',forma.value);
        console.log('Forma validna:' + forma.valid);
        console.log('Forma submitovana:' + forma.submitted);
      }

}
```

iščitavanje forme

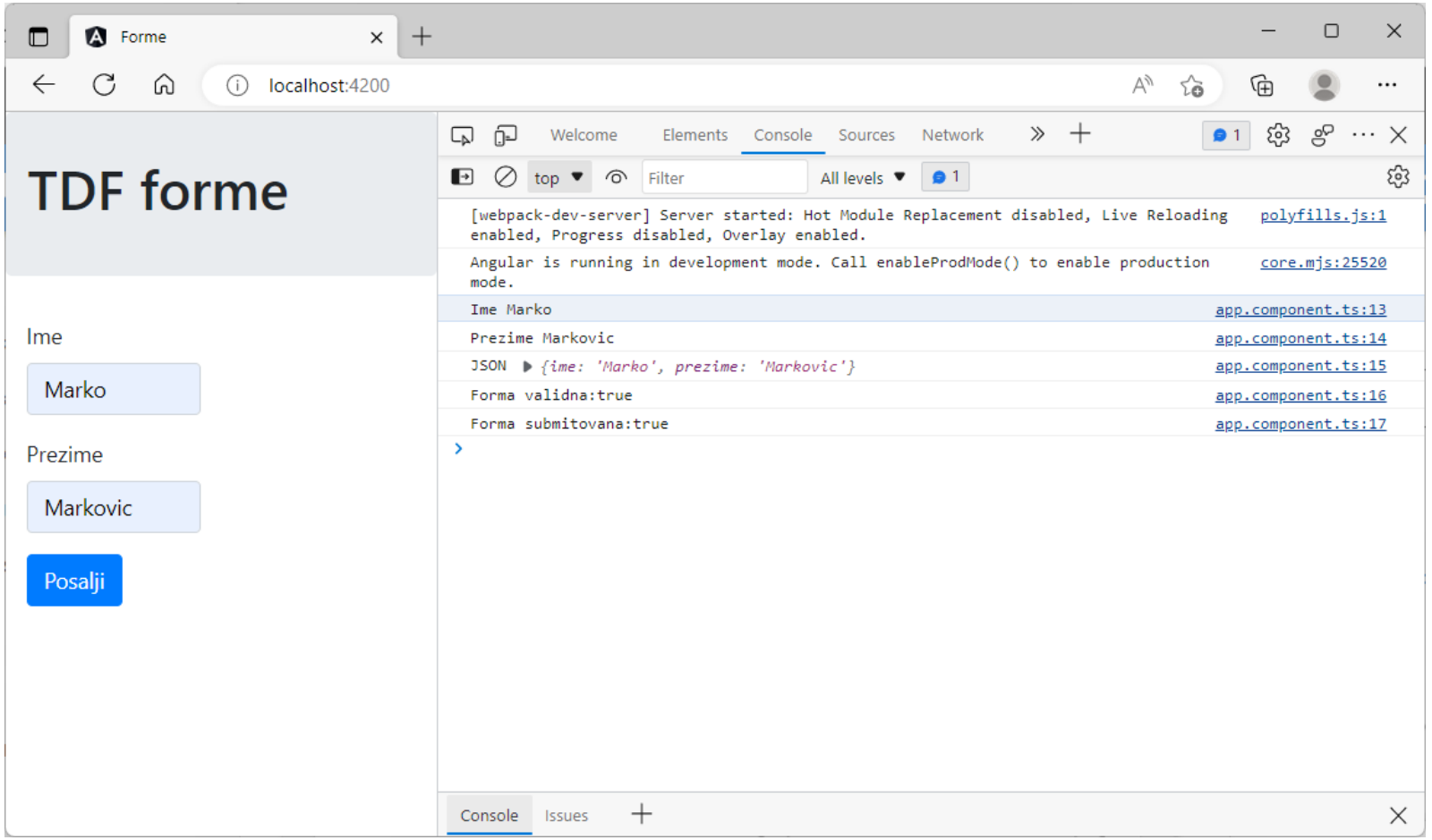
Direktiva ngForm i direktiva ngModel

```
<form #form1="ngForm" (ngSubmit)="onSubmit(form1)">
    <div class="form-group">
        <label for="ime">Ime</label>
        <input type="text" name="ime" id="ime" class="form-control" ngModel />
    </div>

    <div class="form-group">
        <label for="prezime">Prezime</label>
        <input type="text" name="prezime" id="prezime" class="form-control" ngModel />
    </div>

    <div class="form-group">
        <button type="submit" class="btn btn-primary">Posalji</button>
    </div>
</form>
```

Lokalna templejt promenljiva **form1** daje referencu na glavnu FormGroup instancu forme

Direktiva **ngmodel** govori angularu da kreira instancu klase FormControl

Konzola


Forme
localhost:4200
TDF forme
Ime
Marko
Prezime
Markovic
Posalji
Welcome
Elements
Console
Sources
Network
top
Filter
All levels
[webpack-dev-server] Server started: Hot Module Replacement disabled, Live Reloading enabled, Progress disabled, Overlay enabled.
polyfills.js:1
Angular is running in development mode. Call enableProdMode() to enable production mode.
core.mjs:25520
Ime Marko
app.component.ts:13
Prezime Markovic
app.component.ts:14
JSON {ime: 'Marko', prezime: 'Markovic'}
app.component.ts:15
Forma validna:true
app.component.ts:16
Forma submitovana:true
app.component.ts:17
Console
Issues

Model klasa forme

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

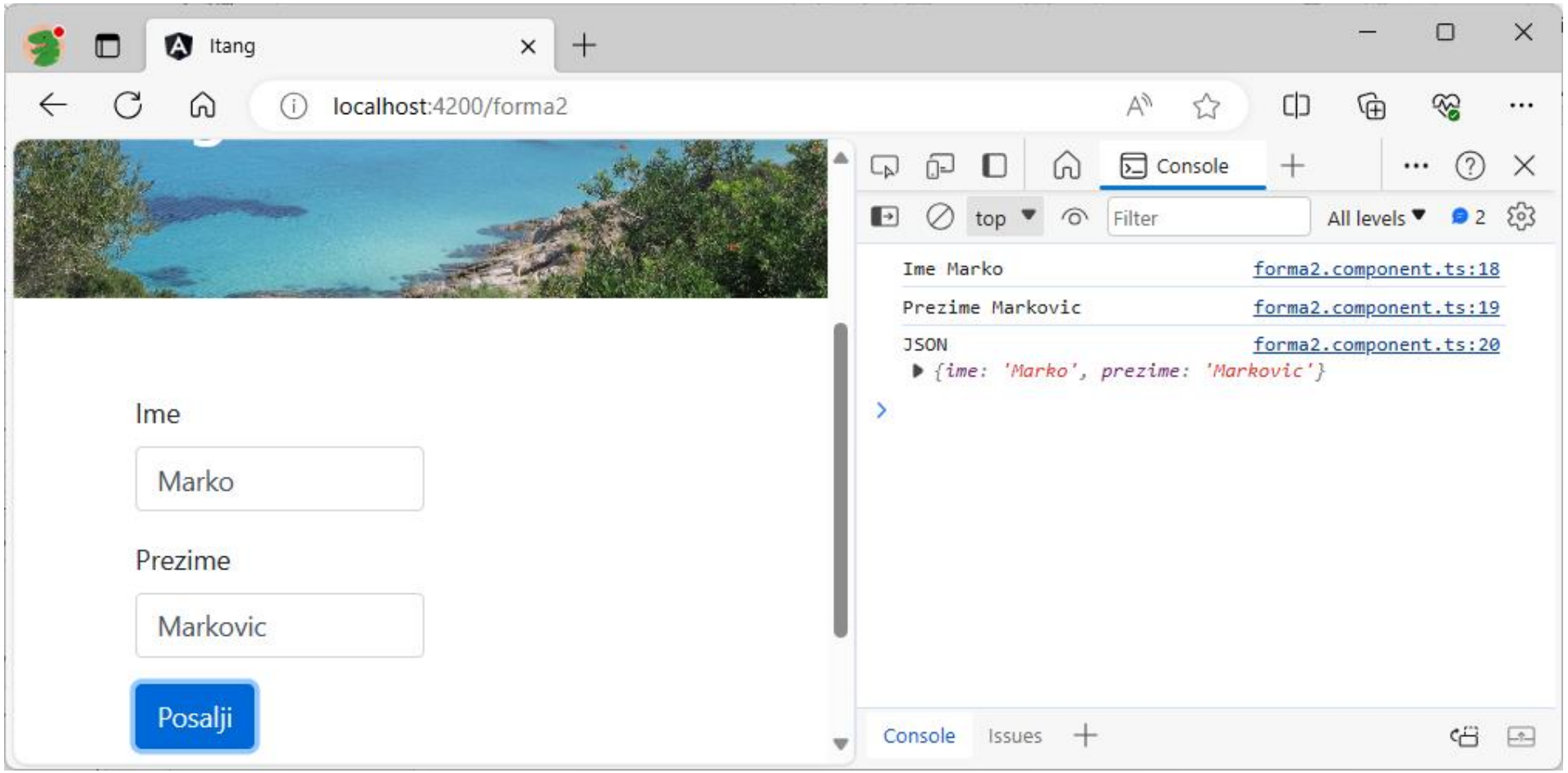
Glavna komponenta

```
export class Forma2Component  {

  title = 'forme';

   model = new Osoba(1, 'Marko', 'Markovic');

  onSubmit(forma: NgForm ) {

    console.log('Ime',this.model.ime);
    console.log('Prezime',this.model.prezime);
    console.log('JSON',forma.value);
  }

}
```

Binding

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

        <form #form1="ngForm" (ngSubmit)="onSubmit(form1)">

            <div class="form-group">
                <label for="ime">Ime</label>
                <input type="text" name="ime" class="form-control" [(ngModel)]="model.ime">
            </div>

            <div class="form-group">
                <label for="prezime">Prezime</label>
                <input type="text" name="prezime" id="prezime" class="form-control“
               [(ngModel)]="model.prezime">
            </div>
            <button class="btn btn-primary"  type="submit">Posalji</button>

        </form>
    </div>
</div>
```

Korisnički interfejs

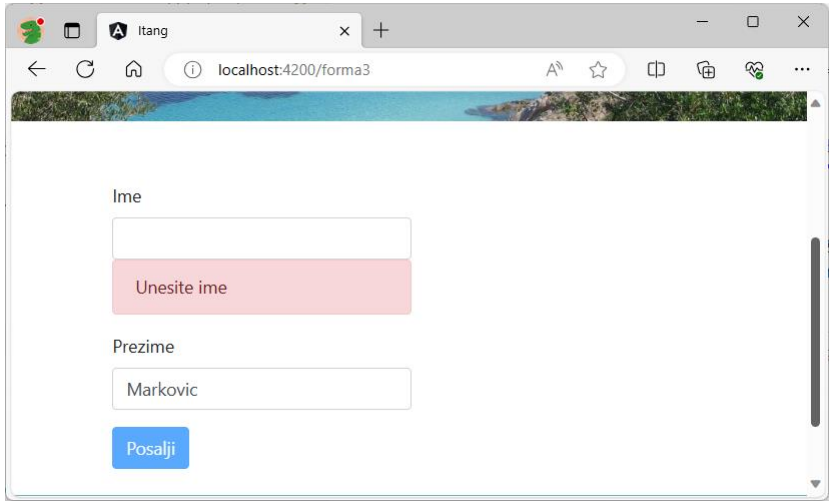


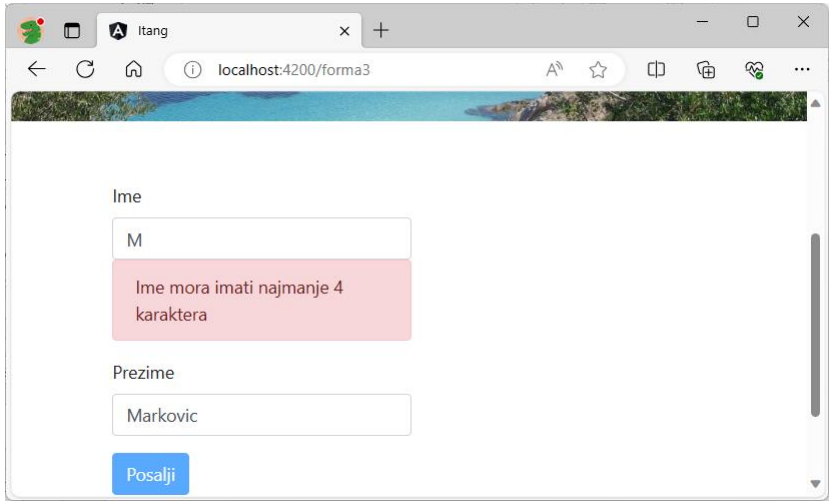
Validacija u TDF

```
<div class="form-group">
    <label for="ime">Ime</label>
    <input type="text" name="ime" class="form-control" [(ngModel)]="model.ime"
#ime="ngModel" required minlength="4">

    <div *ngIf="ime.invalid && (ime.dirty || ime.touched)" class="alert alert-danger">
    <div *ngIf="ime.hasError('required')">
        Unesite ime
    </div>
    <div *ngIf="ime.hasError('minlength')">
        Ime mora imati najmanje 4 karaktera
    </div>
    </div>
</div>
```

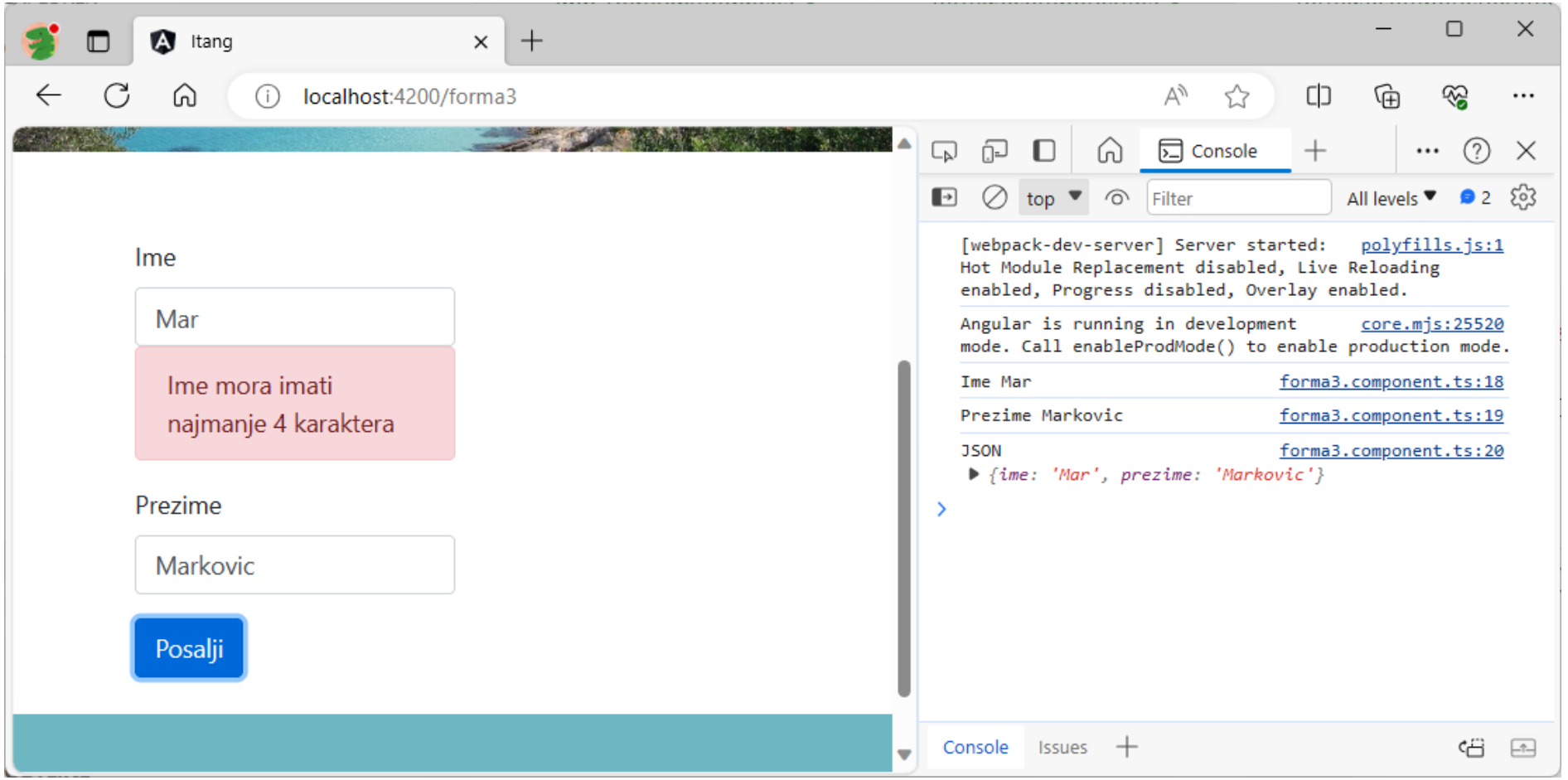
dirty – vrednost je modifikovana
touched – polje je dodirnuto

Prikaz validacionih grešaka





Submitovanje nevalidnih podataka

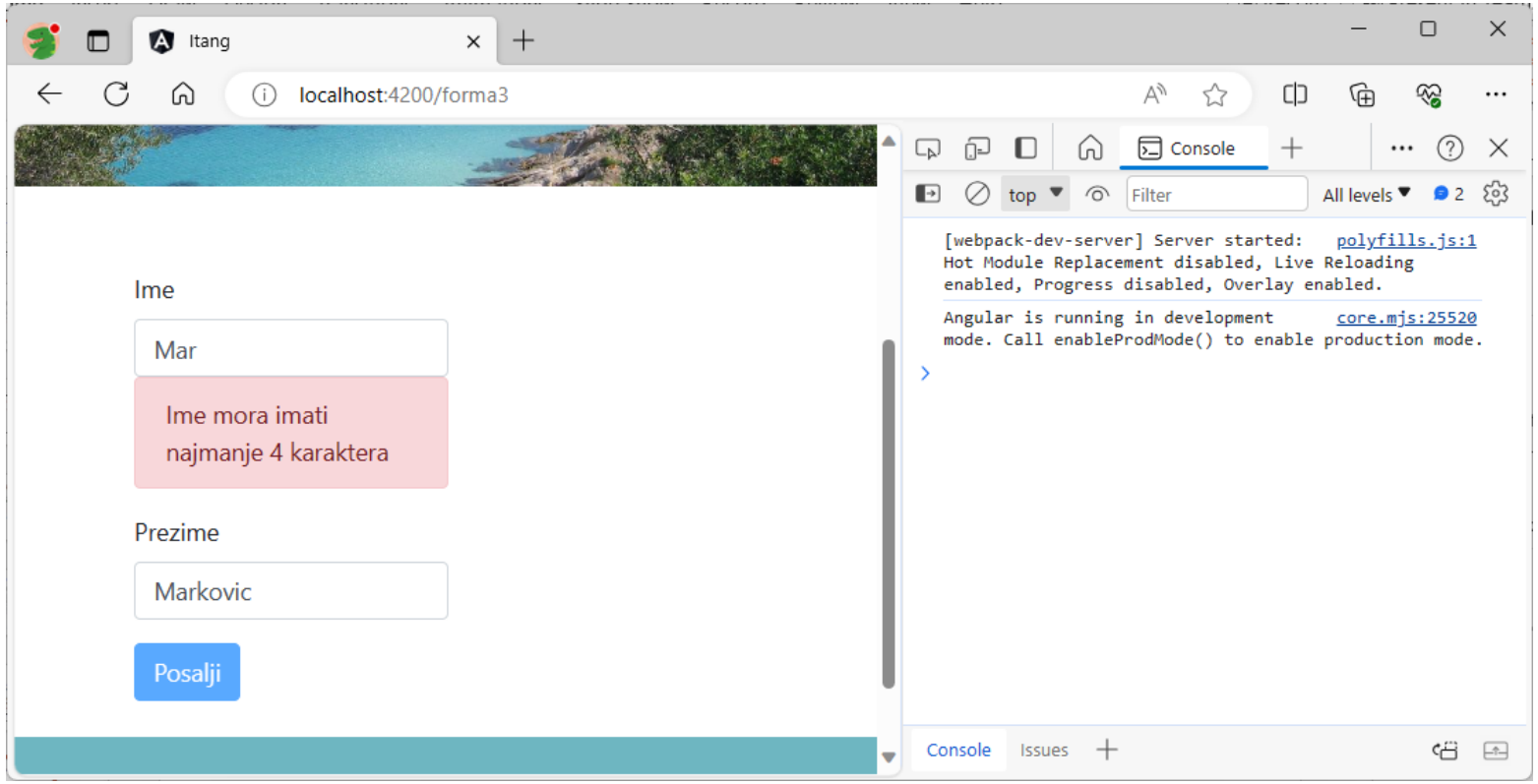


Svojstvo valid forme

- Svojstvo **valid** angular forme vraća true ako su sve kontrole validne
- Obrnuto od svojstva valid je svojstvo **invalid** koje vraća true ako bar jedna od kontrola nije validna
- Svojstva valid i invalid imaju sve kontrole angular forme
- Forma se sabmituje korišćenjem direktive **ngSubmit**
- Potrebno je dizejblovati submit dugme sve dok forma ne postane validna

```
<button [disabled]="form1.invalid" class="btn btn-
primary"  type="submit">Posalji</button>
```

Dugme Posalji dizejblovano



Primer upotrebe TDF formi

```
export class Student {
    constructor(public ime: string, public prezime: string, public pol: string,
     public smer: string  ) {
    }
}
```

Primer upotrebe TDF formi

```
export class AppComponent {

  title = 'primer TDF';
  smerovi = ['Informatika', 'Marketing', 'Menadzment'];
  student: Student = new Student('', '', 'muski', 'Informatika');

  setDefault() {
    this.student.ime = 'Marko';
    this.student.prezime = 'Markovic';
    this.student.pol = 'muski';
    this.student.smer = 'Informatika';
  }

  resetuj() {
    this.student = new Student('', '', 'muski', 'Informatika');
  }
```

Primer upotrebe TDF formi

```
  onSubmit() {
    console.log('Ime : ' + this.student.ime);
    console.log('Prezime : ' + this.student.prezime);
    console.log('Pol: ' + this.student.pol);
    console.log('Smer: ' + this.student.smer);
    this.resetuj();
  }
```

Podešavanje tekst polja

```
<form #studentForm="ngForm" (ngSubmit)="onSubmit()">

    <div class="form-group">
        <label for="">Ime:</label>
        <input name="ime" [(ngModel)]="student.ime" required #ime="ngModel" class="form-control">
        <div *ngIf="ime.invalid &&  (ime.dirty || ime.touched)">
            Unesite ime
        </div>
    </div>

    <div class="form-group">
        <label for="">Prezime:</label>
        <input name="prezime" [(ngModel)]="student.prezime" required #prezime="ngModel"
            class="form-control">
        <div *ngIf="prezime.invalid && (prezime.dirty || prezime.touched)">
            Unesite prezime
        </div>
    </div>
```

Podešavanje radio inputa

```
<label for="">Pol:</label>
<div class="form-check">

    <input type="radio" value="muski" name="pol" [(ngModel)]="student.pol" required #pol="ngModel"
        class="form-check-input">
    <label for="" class="form-check-label">Muski</label>
</div>

<div class="form-check">
    <input type="radio" value="zenski" name="pol" [(ngModel)]="student.pol" required #pol="ngModel"
        class="form-check-input">
    <label for="" class="form-check-label">Zenski</label>
</div>

<div *ngIf="pol.invalid && (pol.dirty || pol.touched)">
    Odaberite pol
</div>
```

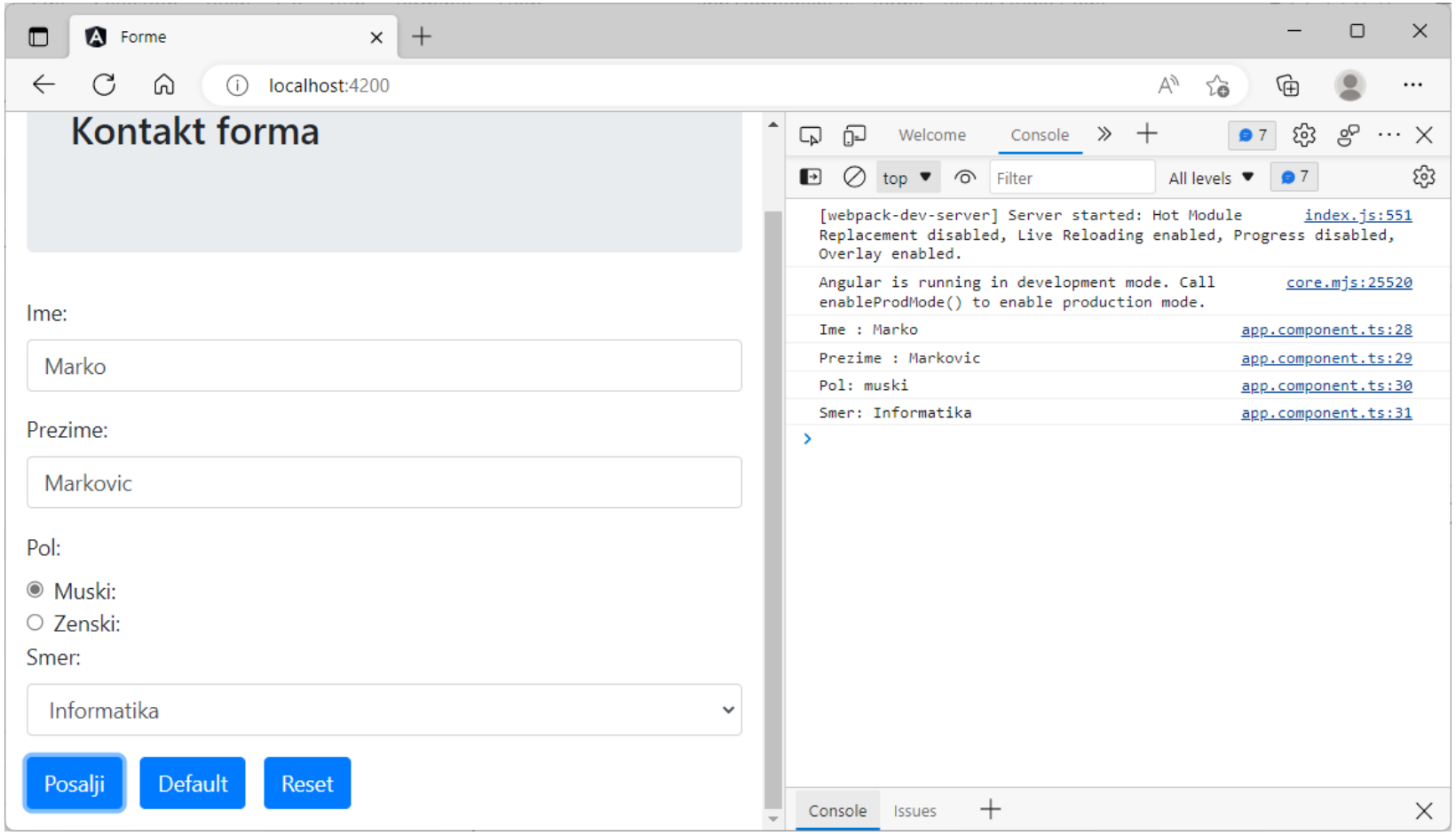
Select kontrola

```
<label for="">Smer:</label>

<select name="smer" [(ngModel)]="student.smer" required #smer="ngModel"
class="form-control">
    <option *ngFor="let smer of smerovi">
        {{smer}}
    </option>
</select>

<div *ngIf="smer.invalid && (smer.dirty || smer.touched)">
    Odaberite smer
</div>
```

Primer upotrebe TDF formi

```
<div class="form-group mt-3">

    <button [disabled]="studentForm.invalid" class="btn btn-primary">Posalji</button>
 
    <button type="button" (click)="setDefault()" class="btn btn-primary">Default</button>
 
    <button type="button" (click)="resetuj()" class="btn btn-primary">Reset</button>
</div>
```

Korisnički interfejs



Pitanje 1

Ulazno polje angular forme predstavlja se klasom:

a. FormControl
b. Control
c. FormField

Odgovor: a

Pitanje 2

Ako se u glavni modul aplikacije importuje FormsModule tada se za svaki tag <form> dodaje direktiva:

a. NgModel
b. NgForm
c. NgField

Odgovor: b

Pitanje 3

Direktiva NgForm:

a. kreira Form instancu koja odgovara formi NgForm
b. kreira FormControl instancu koja odgovara formi
c. kreira FormGroup instancu koja odgovara formi

Odgovor: c

Pitanje 4

Svojstvo dirty instance klase FormControl vraća true ako je :

a. vrednost odgovarajućeg ulaznog polja promenjena
b. vrednost odgovarajućeg ulaznog polja nepromenjena
c. odgovarajuće ulazno polje prazno

Odgovor: a