JavaScript
TypeScript
Angular Framework

Angular

- JavaScript (TypeScript) framework koji omogućava razvoj reaktivnih Single Page Aplikacija (SPA)
  - SPA je web sajt kod kojeg se sadržaj nalazi na jednoj stranici. U zavisnosti od toga šta korisnik zatraži da vidi (klikom, akcijom...) prikazuje se taj dio.
- Razvijen od strane kompanije Google
- Baziran na komponentama
- Podržava različite platforme: native, desktop, progressive web apps

Angular

- Podešavanje:
  - Instaliranje node.js
    https://nodejs.org/en/download/
  - Instaliranje Angular CLI
    - **npm install -g @angular/cli**

Angular CLI – osnove

- Kreiranje nove Angular aplikacije u nekom folderu

  ng new my-dream-app
- Ako ima ista greška više puta onda ide:

  npm cache clean --force
- Kada se aplikacija kreira, potrebno je otvoriti folder gdje je kreirana

  cd my-dream-app
- Nakon toga aplikacija se može pokrenuti

  ng serve --open
- Kada se pokrene, aplikaciji se pristupa u web browseru na:

  http://localhost:4200

Angular Install

```
E:\Enviroment\Y27-Angular>npm install -g @angular/cli
npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
npm WARN deprecated uuid@3.4.0: Please upgrade  to version 7 or higher.  Older versions may use Math.random() in certain
 circumstances, which is known to be problematic.  See https://v8.dev/blog/math-random for details.
C:\Users\MV\AppData\Roaming\npm\ng -> C:\Users\MV\AppData\Roaming\npm\node_modules\@angular\cli\bin\ng

> @angular/cli@12.2.0 postinstall C:\Users\MV\AppData\Roaming\npm\node_modules\@angular\cli
> node ./bin/postinstall/script.js

? Would you like to share anonymous usage data with the Angular Team at Google under
Google's Privacy Policy at https://policies.google.com/privacy? For more details and
how to change this setting, see https://angular.io/analytics. Yes

Thank you for sharing anonymous usage data. If you change your mind, the following
command will disable this feature entirely:

    ng analytics off

+ @angular/cli@12.2.0
added 237 packages from 181 contributors in 62.405s
```

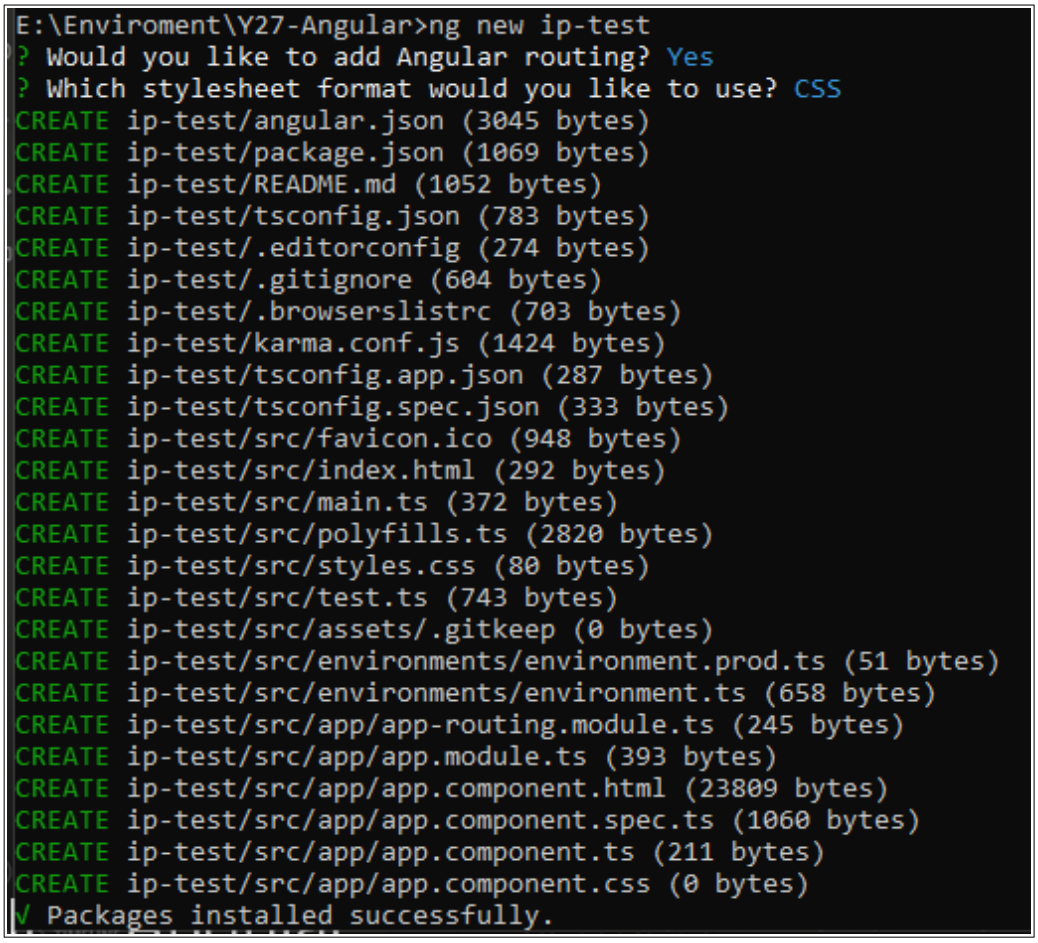
Typescript Install

```
E:\Enviroment\Y27-Angular>npm install -g typescript
C:\Users\MV\AppData\Roaming\npm\tsc -> C:\Users\MV\AppData\Roaming\npm\node_modules\typescript\bin\tsc
C:\Users\MV\AppData\Roaming\npm\tsserver -> C:\Users\MV\AppData\Roaming\npm\node_modules\typescript\bin\tsserver
+ typescript@4.3.5
added 1 package from 1 contributor in 24.188s

E:\Enviroment\Y27-Angular>
```

Ng

```
E:\Enviroment\Y27-Angular>ng
Available Commands:
  add Adds support for an external library to your project.
  analytics Configures the gathering of Angular CLI usage metrics. See https://angular.io/cli/usage-analytics-gathering.
  build (b) Compiles an Angular app into an output directory named dist/ at the given output path. Must be executed from
 within a workspace directory.
  deploy Invokes the deploy builder for a specified project or for the default project in the workspace.
  config Retrieves or sets Angular configuration values in the angular.json file for the workspace.
  doc (d) Opens the official Angular documentation (angular.io) in a browser, and searches for a given keyword.
  e2e (e) Builds and serves an Angular app, then runs end-to-end tests.
  extract-i18n (i18n-extract, xi18n) Extracts i18n messages from source code.
  generate (g) Generates and/or modifies files based on a schematic.
  help Lists available commands and their short descriptions.
  lint (l) Runs linting tools on Angular app code in a given project folder.
  new (n) Creates a new workspace and an initial Angular application.
  run Runs an Architect target with an optional custom builder configuration defined in your project.
  serve (s) Builds and serves your app, rebuilding on file changes.
  test (t) Runs unit tests in a project.
  update Updates your application and its dependencies. See https://update.angular.io/
  version (v) Outputs Angular CLI version.

For more detailed help run "ng [command name] --help"

E:\Enviroment\Y27-Angular>
```

ng new ip-test



Uvod u TypeScript

- Proširenje JavaScript-a u kojem se kod kompajlira
- Instaliranje: npm install -g typescript
- Fajlovi se završavaju sa .ts
- TypeScript zahtjeva definisanje tipova podataka:
  - let s: string = “ETF”;
  - let podaci: Array = [];
  - let osobe: Array<Osoba> = [];

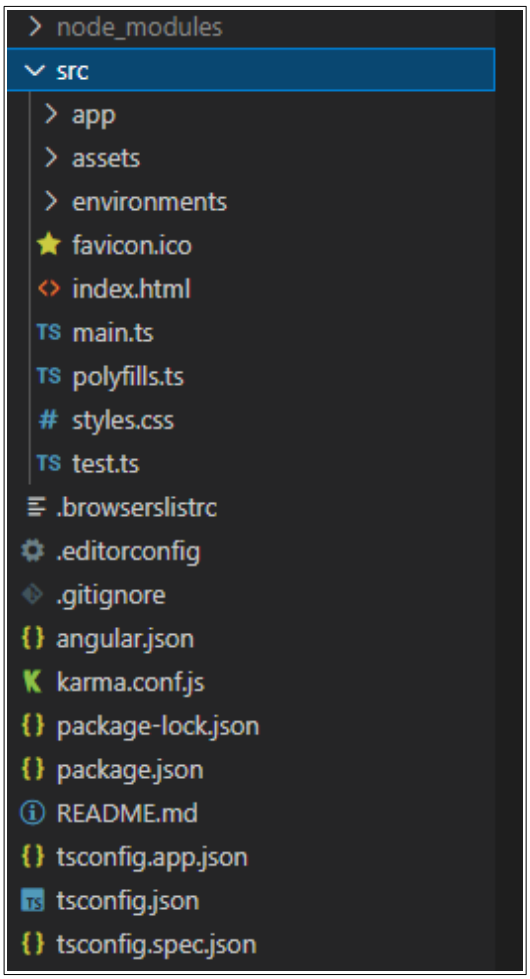
Uvod u TypeScript klase

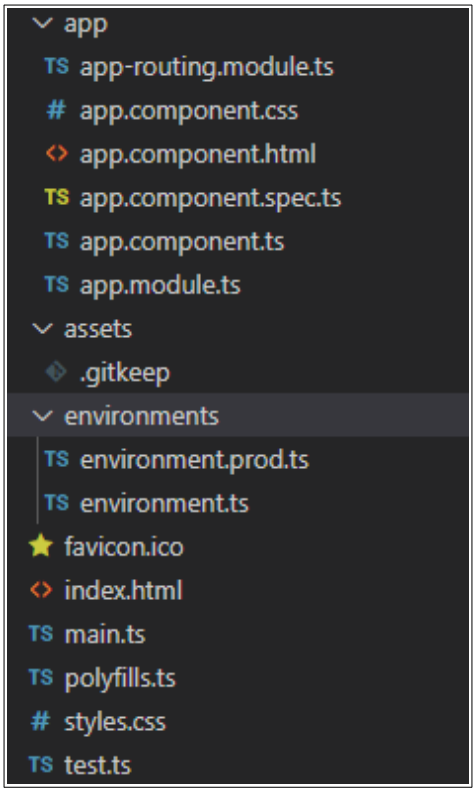
```
class User {
  private name: string;
  constructor(name) {
     this.name = name;
  }
  sayHi() {
     alert(this.name);
  }
}
```

- let osoba: User = new User("test");
- osoba.sayHi();

Struktura Angular projekta

- **src** folder – sadrži glavni kod aplikacije
  - **app** folder – fajlovi koji se koriste za komponente
    - **app-component.css** – CSS stilovi za app komponentu
    - **app-component.html** – HTML template za povezivanje
    - **app-component.ts** – fajl koji definiše logiku komponenti
    - **app-component.spec.ts** – fajl za Unit test
    - **app.module.ts** – fajl u kojem se definišu moduli, komponente i ostala podešavanja aplikacije kako bi se mogla koristiti


node_modules
src
app
assets
environments
favicon.ico
index.html
main.ts
polyfills.ts
styles.css
test.ts
.browserslistrc
.editorconfig
.gitignore
angular.json
karma.conf.js
package-lock.json
package.json
README.md
tsconfig.app.json
tsconfig.json
tsconfig.spec.json


app
app-routing.module.ts
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
assets
.gitkeep
environments
environment.prod.ts
environment.ts
favicon.ico
index.html
main.ts
polyfills.ts
styles.css
test.ts

main.ts

```ts
import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

import { AppModule } from './app/app.module';
import { environment } from './environments/environment';

if (environment.production) {
  enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.error(err));
```

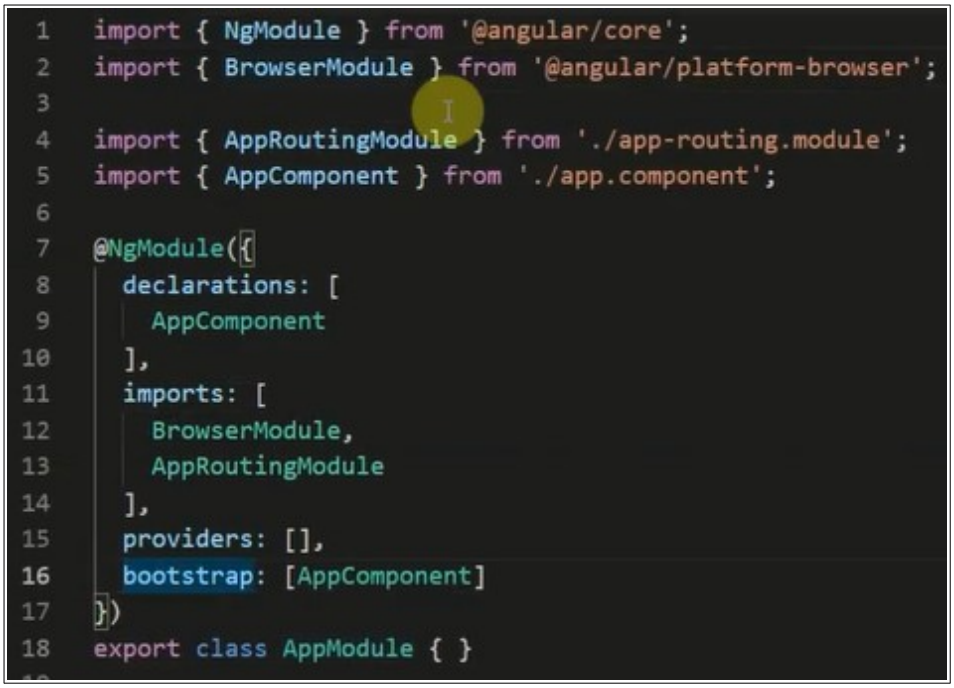
Почетни модуо – кориштење

index.html

```html
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>IpTest</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

Општи шаблон за почетне странице

app.module.ts



```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';

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

Дефиниција основног модула модула

- Компоненте које се налазе у модулу
- Укључења других модула
- Сервиси који се користе
- Прве компоненте које ће се извршити

app.component.html

```
  <svg id="clouds" xmlns="http://www.w3.org/2000/svg" width="2611.084" h
    <title>Gray Clouds Background</title>
    <path id="Path_39" data-name="Path 39" d="M2379.709,863.793c10-93-7
  </svg>

</div>

<!-- * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * -->
<!-- * * * * * * * * * * The content above * * * * * * * * * * * * * * * -->
<!-- * * * * * * * * * * is only a placeholder * * * * * * * * * * * * * -->
<!-- * * * * * * * * * * and can be replaced. * * * * * * * * * * * * * -->
<!-- * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * -->
<!-- * * * * * * * * * * End of Placeholder * * * * * * * * * * * * * * -->
<!-- * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * -->

<router-outlet></router-outlet>
```

Дио који укључује различите садржаје у страницу у зависности од УРЛ адресе

app.componnent.ts

```
1   import { Component } from '@angular/core';
2
3   @Component({
4     selector: 'app-root',
5     templateUrl: './app.component.html',
6     styleUrls: ['./app.component.css']
7   })
8   export class AppComponent {
9     title = 'ip-test';
10  }
```

Компонента која се односи на основни облик странице

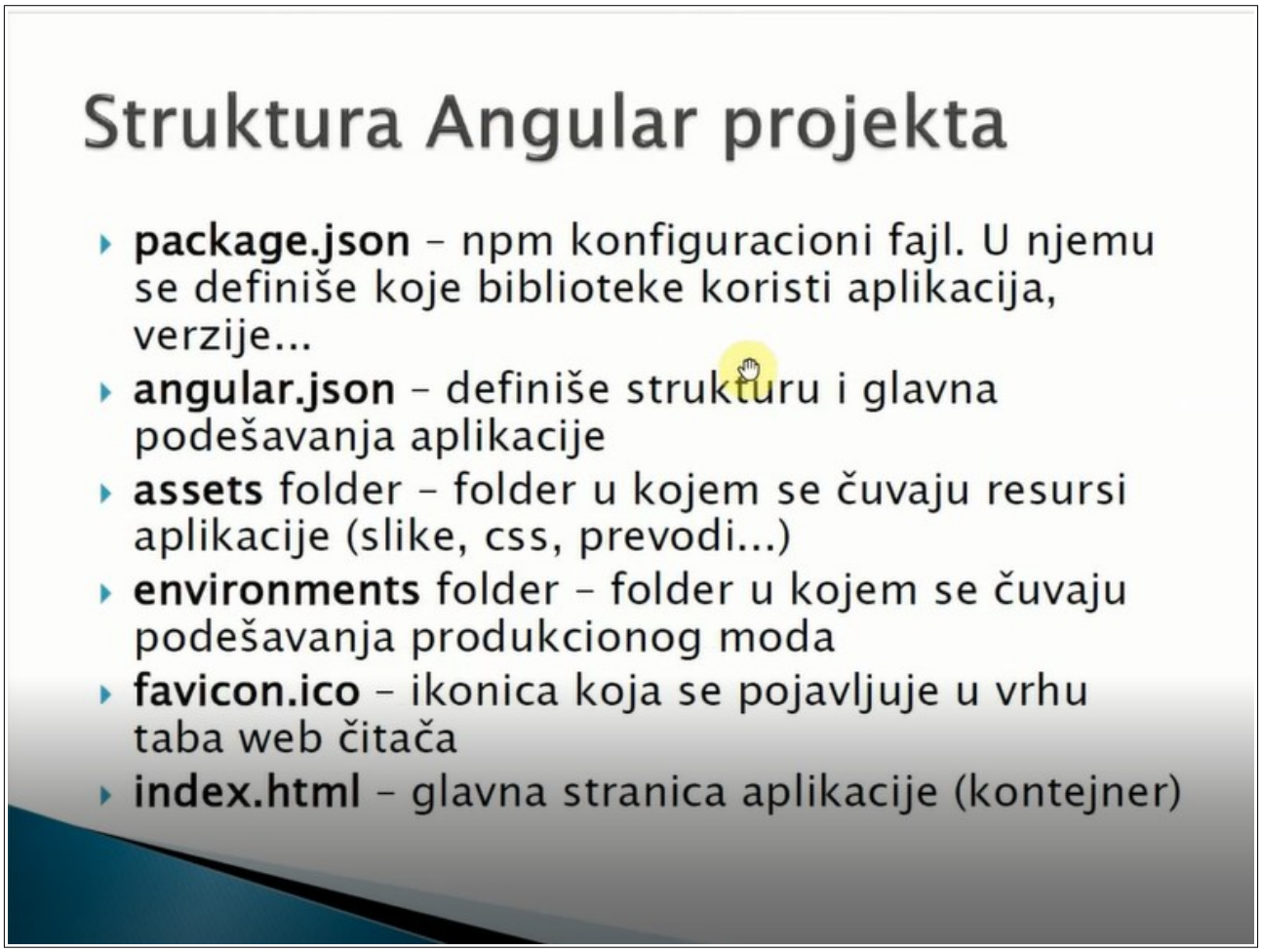
- Селектор, користи се за укључење компоненти помоћу тагова, као на пријер у индексу.
- Адресе шаблон за садржај
- Стилови који се примјењују за компоненте

app.routing.module.ts

```
1   import { NgModule } from '@angular/core';
2   import { RouterModule, Routes } from '@angular/router';
3
4   const routes: Routes = [];
5
6   @NgModule({
7     imports: [RouterModule.forRoot(routes)],
8     exports: [RouterModule]
9   })
10  export class AppRoutingModule { }
```

Модуо и код задужен за рутирање и управљање рутама

- Руте које су специфичне за апликацију дефинисане преко JCOH подобјеката
- Позивање и прослеђивање рута кроз апликацију
- Јавно доступни модули и код.



Uvod u TypeScript klase

- Koncept iz objektno orijentisanog programiranja (OOP)
- Definiše strukturu objekta (atribute i funkcije)
- Svaka klasa ima konstruktor koji je zadužen da napravi objekat na osnovu klase
- Objekat je konkretan entitet neke klase

Struktura Angular projekta

- **main.ts** – fajl koji se prvi izvršava u aplikaciji. Koristi se za definisanje globalne konfiguracije
- **styles.css** – globalni CSS fajl

Biblioteke

- Dodavanje u *dependencies* dio package.json fajla
- Instaliranje svih biblioteka iz package.json fajla
  - npm install
- Ažuriranje biblioteka
  - npm update
- Dodavanje neke biblioteke
  - npm install naziv [-g -s]
    - -g - globalno
    - -s - dodaje u package.json

Arhitektura

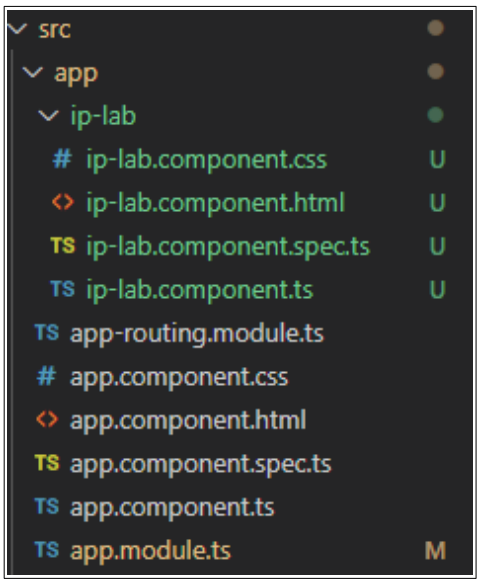
- Angular aplikacija je skup **modula** (ngModules)
- Uvijek ima bar jedan root modul koji omogućava pokretanje aplikacije
- **Komponente** definišu pogled (HTML elemente koji omogućavaju korištenje aplikacije)
- Komponente koriste **servise**, koji obezbjeđuju funkcionalnost koja nije vezana samo za pogled

Arhitektura – komponente

- Komponenta ima class fajl koji služi za čuvanje podataka i logike koja je vezana za HTML templejt.
- HTML templejt se koristi za prikaz podataka komponente, interakciju sa korisnikom i pozivanje novih funkcionalnosti od strane korisnika.
- Svaka komponenta je vezana za neki HTML fajl
- Kreiranje nove komponente
  - ng generate component naziv
  - ili ng g c naziv

ng g c ip-lab

```
E:\Enviroment\Y27-Angular>cd ip-test

E:\Enviroment\Y27-Angular\ip-test>ng g c ip-lab
? Would you like to share anonymous usage data about this project with the Angular Team at
Google under Google's Privacy Policy at https://policies.google.com/privacy? For more
details and how to change this setting, see https://angular.io/analytics. Yes

Thank you for sharing anonymous usage data. Would you change your mind, the following
command will disable this feature entirely:

    ng analytics project off

CREATE src/app/ip-lab/ip-lab.component.html (21 bytes)
CREATE src/app/ip-lab/ip-lab.component.spec.ts (620 bytes)
CREATE src/app/ip-lab/ip-lab.component.ts (274 bytes)
CREATE src/app/ip-lab/ip-lab.component.css (0 bytes)
UPDATE src/app/app.module.ts (473 bytes)

E:\Enviroment\Y27-Angular\ip-test>
```



app.module.ts

```ts
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { IpLabComponent } from './ip-lab/ip-lab.component';

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

```

ip-lab.component.html

```
<p>ip-lab works!</p>
```

Садржај новокреиране компоненте

ip-lab.component.ts

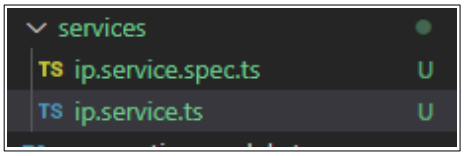
```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-ip-lab',
  templateUrl: './ip-lab.component.html',
  styleUrls: ['./ip-lab.component.css']
})
export class IpLabComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {
  }

}
```

Управљачки код новоизабране компоненте

- Догађаји при иницијализацији

ng g s services/ip

```
E:\Enviroment\Y27-Angular\ip-test>ng g s services/ip
CREATE src/app/services/ip.service.spec.ts (337 bytes)
CREATE src/app/services/ip.service.ts (131 bytes)

E:\Enviroment\Y27-Angular\ip-test>
```



ip.service.ts

```
1  import { Injectable } from '@angular/core';
2
3  @Injectable({
4    providedIn: 'root'
5  })
6  export class IpService {
7
8    constructor() { }
9  }
```

Сервисна класа – примјер импровизованог сервиса

Dependency Injection
Глобално декларисан сервис

ip.service.ts

```
1  import { Injectable } from '@angular/core';
2
3  @Injectable({
4    providedIn: 'root'
5  })
6  export class IpService {
7
8    constructor() { }
9
10   showInfo(){
11     console.log('IP 2021');
12   }
13
14 }
15
```

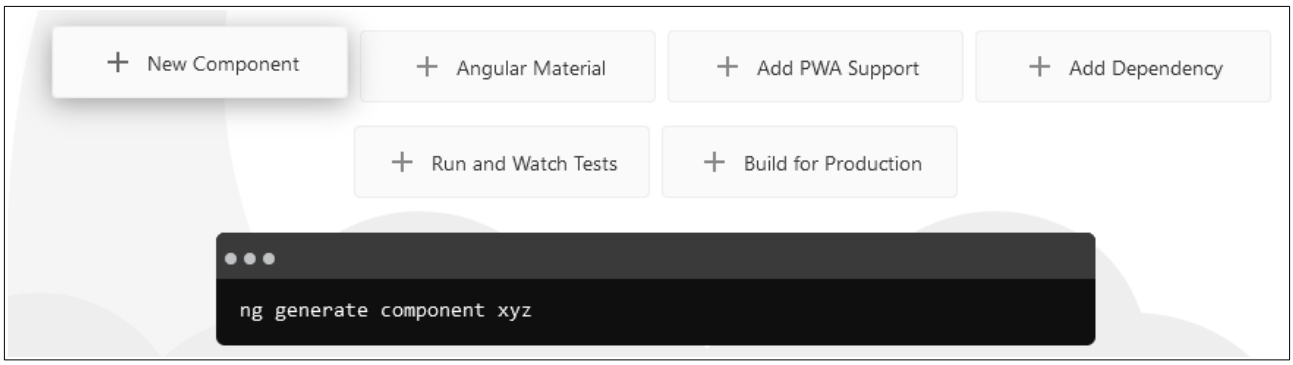
- show info method/function

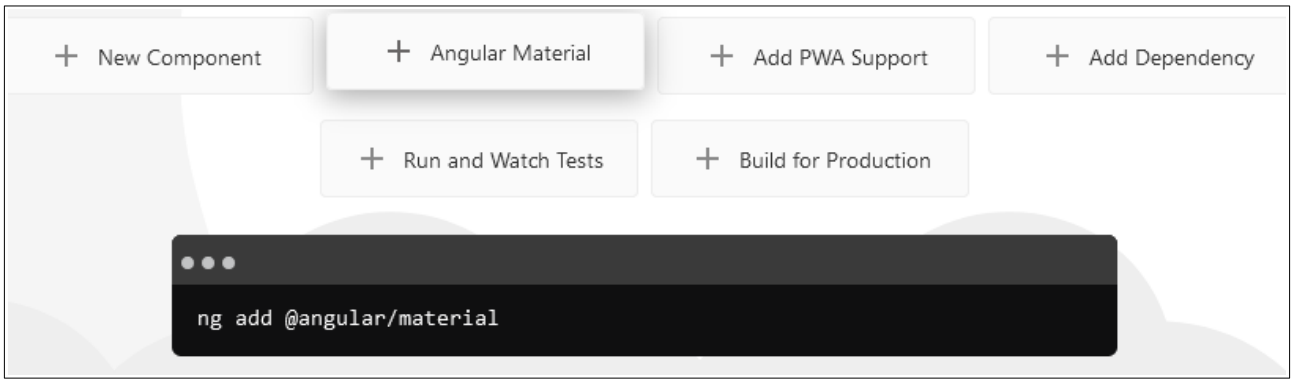
ip-lab.component.ts

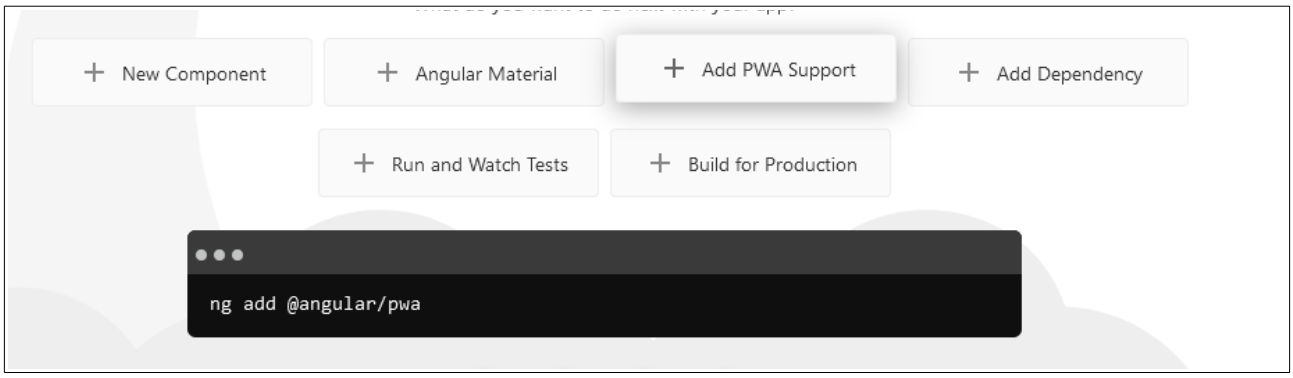
```ts
import { Component, OnInit } from '@angular/core';
import { IpService } from '../services/ip.service';

@Component({
  selector: 'app-ip-lab',
  templateUrl: './ip-lab.component.html',
  styleUrls: ['./ip-lab.component.css']
})
export class IpLabComponent implements OnInit {

  constructor(private service: IpService) { }

  ngOnInit(): void {
    this.service.showInfo();
  }

}
```

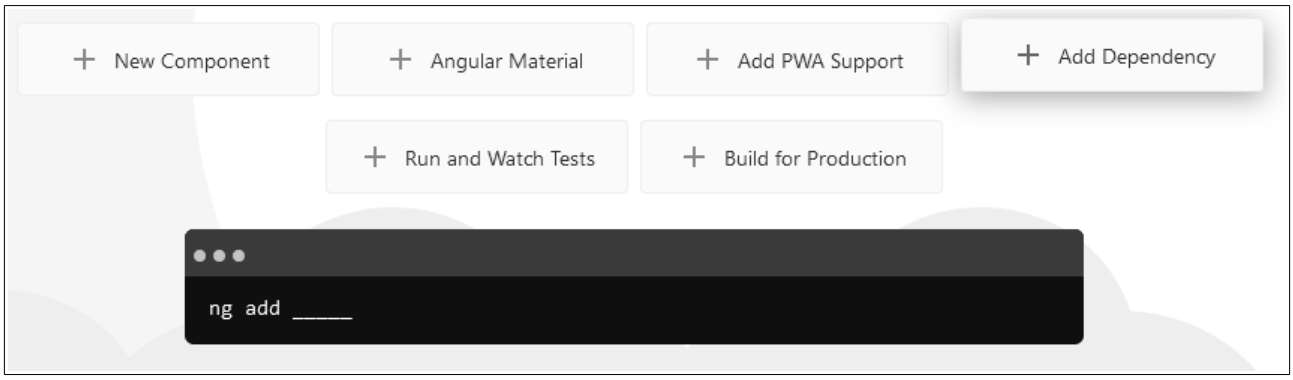
ng serve

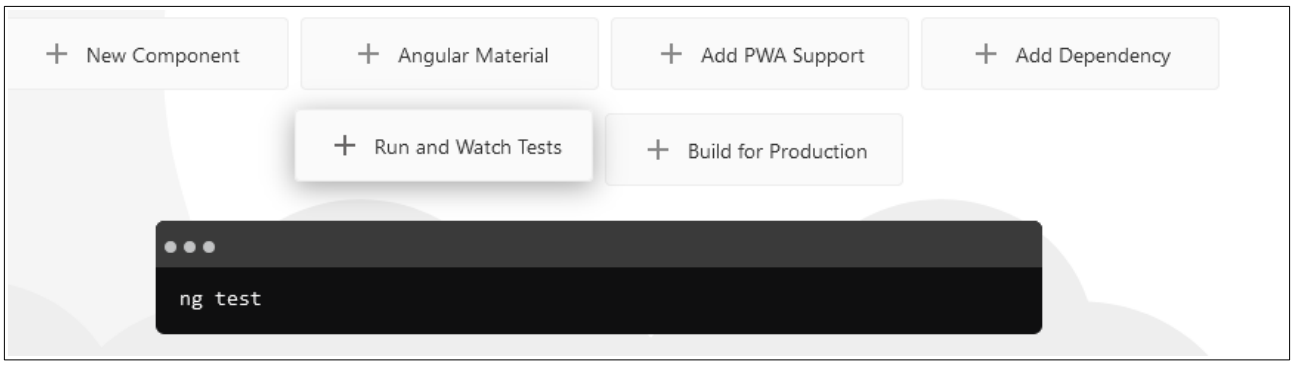
```
E:\Enviroment\Y27-Angular\ip-test>ng serve
- Generating browser application bundles (phase: setup)...Compiling @angular/core : es2015 as esm2015
Compiling @angular/common : es2015 as esm2015
Compiling @angular/router : es2015 as esm2015
Compiling @angular/platform-browser : es2015 as esm2015
Compiling @angular/platform-browser-dynamic : es2015 as esm2015
√ Browser application bundle generation complete.

Initial Chunk Files   | Names         |      Size
vendor.js             | vendor        |   2.38 MB
polyfills.js          | polyfills     | 508.73 kB
styles.css, styles.js | styles        | 380.92 kB
main.js               | main          |  57.70 kB
runtime.js            | runtime       |   6.57 kB

                      | Initial Total |   3.31 MB

Build at: 2021-08-10T19:16:28.765Z - Hash: 2667e9f5ad9f6b7e1b1e - Time: 24719ms

** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **


√ Compiled successfully.
√ Browser application bundle generation complete.

5 unchanged chunks

Build at: 2021-08-10T19:16:30.430Z - Hash: 81f3b128079d2db4153a - Time: 1117ms

√ Compiled successfully.
```

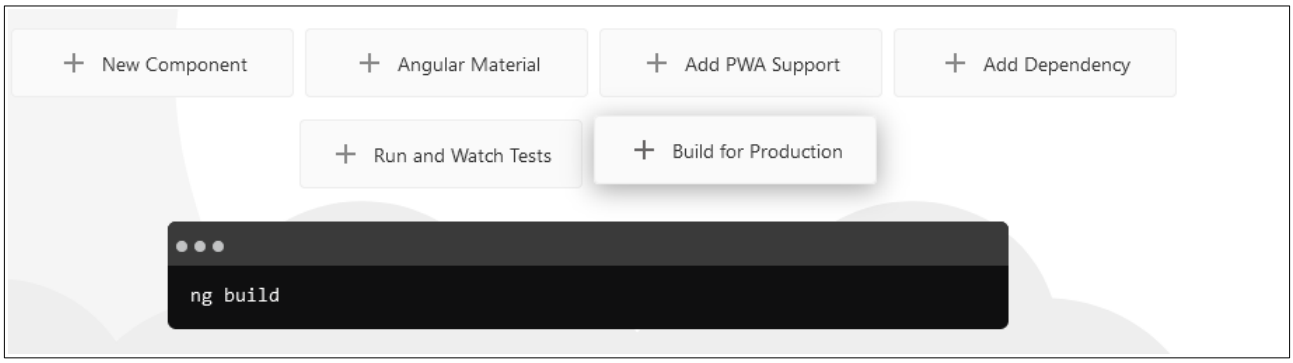

New Component
Angular Material
Add PWA Support
Add Dependency
Run and Watch Tests
Build for Production
ng generate component xyz

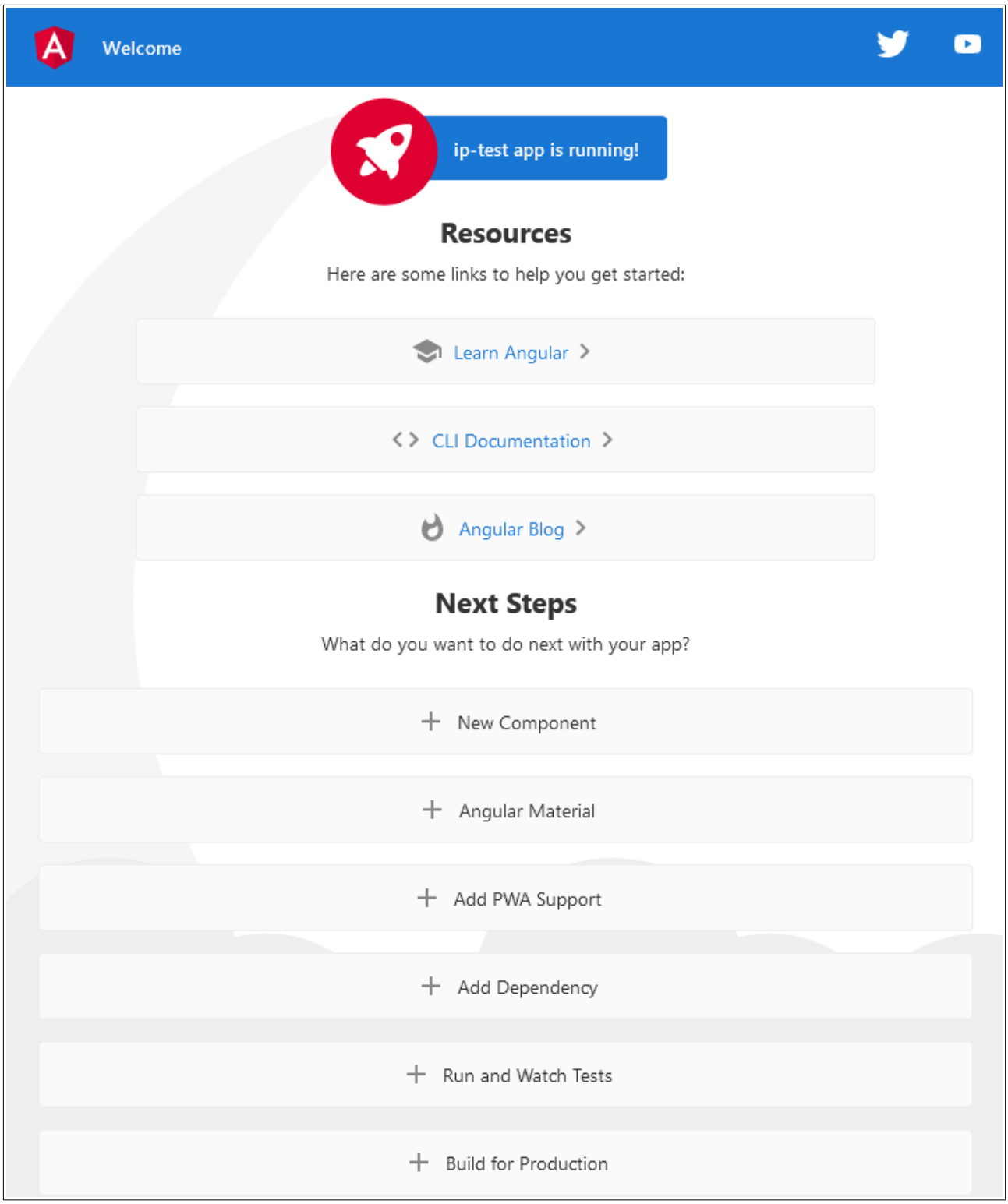

New Component
Angular Material
Add PWA Support
Add Dependency
Run and Watch Tests
Build for Production
ng add @angular/material


New Component
Angular Material
Add PWA Support
Add Dependency
Run and Watch Tests
Build for Production
ng add @angular/pwa


New Component
Angular Material
Add PWA Support
Add Dependency
Run and Watch Tests
Build for Production
ng add _____


New Component
Angular Material
Add PWA Support
Add Dependency
Run and Watch Tests
Build for Production
ng test

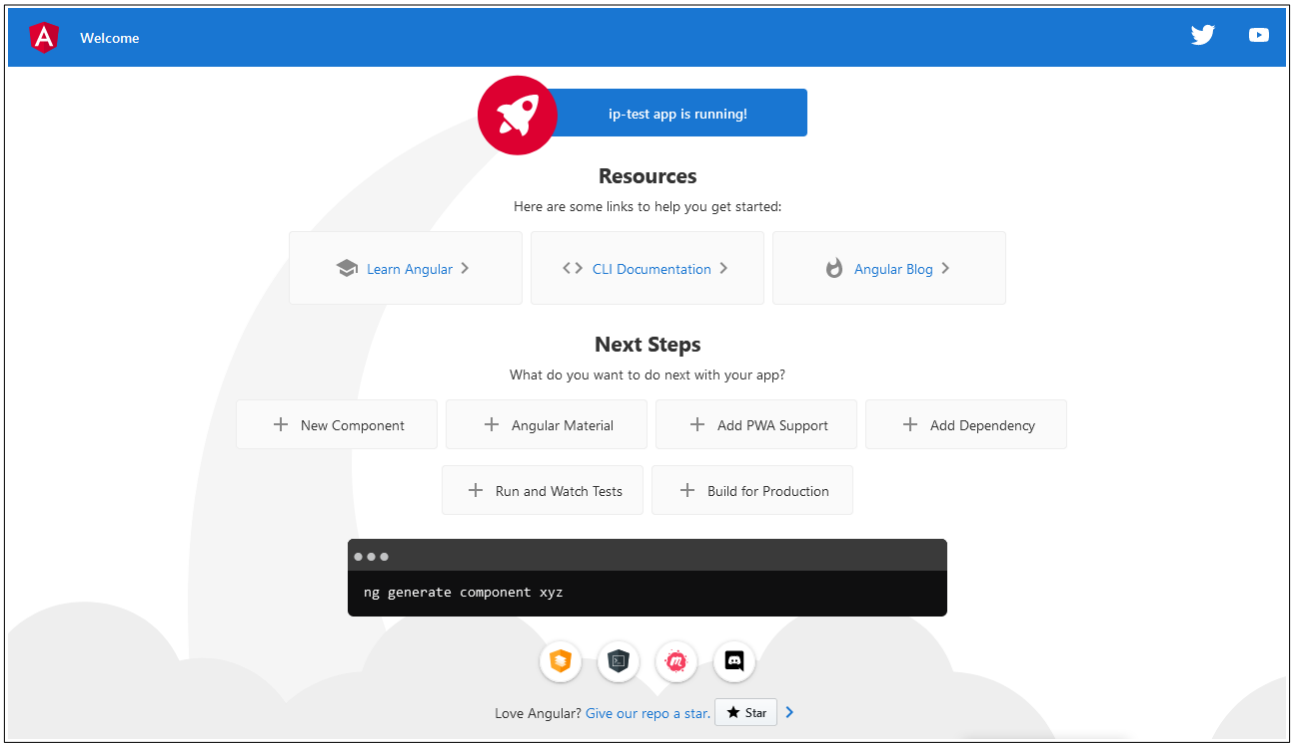

New Component
Angular Material
Add PWA Support
Add Dependency
Run and Watch Tests
Build for Production
ng build


Welcome
ip-test app is running!
Resources
Here are some links to help you get started:
Learn Angular
CLI Documentation
Angular Blog
Next Steps
What do you want to do next with your app?
New Component
Angular Material
Add PWA Support
Add Dependency
Run and Watch Tests
Build for Production

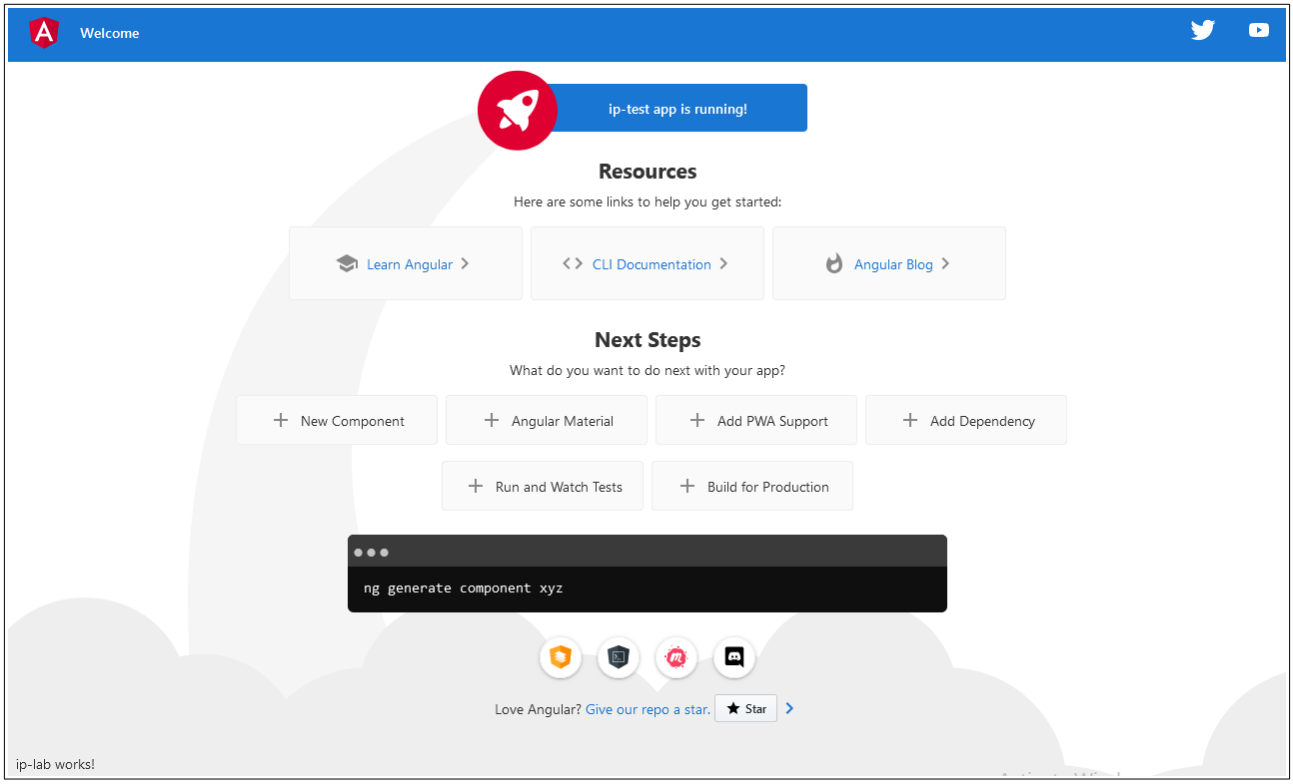
app-routing-module.ts

```ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { IpLabComponent } from './ip-lab/ip-lab.component';

const routes: Routes = [
  {
    path: 'etf',
    component: IpLabComponent
  }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }

```

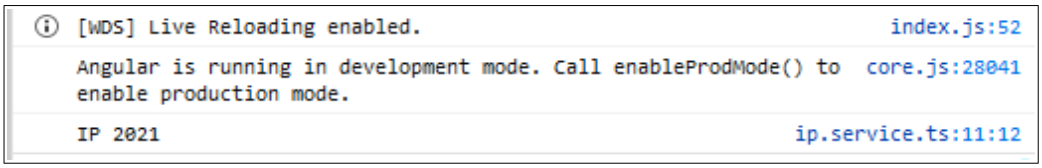
localhost:4200/



localhost:4200/etf



Firefox console
localhost:4200/etf



Arhitektura – moduli

- AppModule – root modul aplikacije koji obezbjeđuje mehanizam za pokretanje aplikacije
- Ostali moduli služe za "grupisanje" dijelova aplikacije u cjeline u zavisnosti od toga šta ti dijelovi rade. Poželjno je da komponente, servisi... koji su vezani za istu grupu funkcionalnosti budu u istom modulu
- Moduli mogu uključivati druge module

Direktive

- Direktive su instrukcije DOM-u koje određuju kako će se neki element ponašati na stranici
- **Direktive komponenti** – koriste se u glavnoj klasi i određuju kako će se komponenta obraditi i koristiti tokom izvršavanja
- **Strukturne komponente** – počinju sa * i određuju stukturu DOM-a (*ngIf, *ngFor, ...)
- **Direktive atributa** – određuju kako će izgledati element (ngClass, ngStyle, ...)

Povezivanje podataka

- Databinding omogućava da programska logika (TypeScript dio) i korisnici (preko HTML dijela) komuniciraju
- **One-way databinding** (jednosmjerno) – omogućava da se HTML izmjeni na osnovu promjena sadržaja modela
- **Two-way databinding** (dvosmjerno) – omogućava da se sinhronizovano ažuriraju i model i pogled (HTML). Kad se napravi izmjena u modelu to se automatski prikazuje u HTML-u i obrnuto, kada se nešto promijeni na HTML to se ažurira u modelu

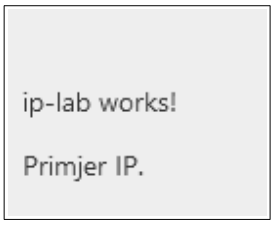
ip-lab.component.ts

```
1  import { Component, OnInit } from '@angular/core';
2  import { IpService } from '../services/ip.service';
3
4  @Component({
5    selector: 'app-ip-lab',
6    templateUrl: './ip-lab.component.html',
7    styleUrls: ['./ip-lab.component.css']
8  })
9  export class IpLabComponent implements OnInit {
10
11   constructor(private service: IpService) { }
12
13   public poruka : string = "Primjer IP.";
14
15   ngOnInit(): void {
16     this.service.showInfo();
17   }
18
19 }
```

ip-lab.component.html

```
1   <p>ip-lab works!</p>
2   {{ poruka }}
```

localhost:4200/etf



Povezivanje podataka

- String interpolation – ubacuje sadržaj iz modela u HTML element

```
import { Component } from '@angular/core';
@Component({
    selector: 'my-app',
    template: `<h1>{{ fullName }}</h1>`
})
export class AppComponent {
    fullName: string = 'ETF';
}
```

```
<h1>{{ fullName }}</h1>
```

Povezivanje podataka

- Property binding – dinamički povezuje sadržaj iz modela u HTML element

```
import { Component } from '@angular/core';
@Component({
    selector: 'my-app',
    template: `<h1 [innerHtml]='fullName'></h1>`
})
export class AppComponent {
    fullName: string = 'ETF';
}
```

```
<h1 [innerHtml]='fullName'></h1>
```

```
1  import { Component, OnInit } from '@angular/core';
2  import { IpService } from '../services/ip.service';
3
4  @Component({
5    selector: 'app-ip-lab',
6    templateUrl: './ip-lab.component.html',
7    styleUrls: ['./ip-lab.component.css']
8  })
9  export class IpLabComponent implements OnInit {
10
11   constructor(private service: IpService) { }
12
13   public poruka : string = "<b>Primjer IP.</b>";
14
15   ngOnInit(): void {
16     this.service.showInfo();
17   }
18
19 }
20
```

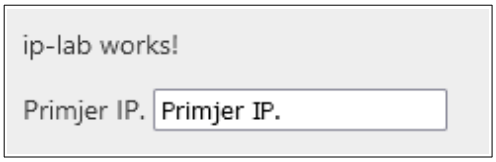
ip-lab.component.html

```
<p>ip-lab works!</p>
{{ poruka }}
<input type='text' [value]='poruka'/>

```

ip-lab.component.ts

```
import { Component, OnInit } from '@angular/core';
import { IpService } from '../services/ip.service';

@Component({
  selector: 'app-ip-lab',
  templateUrl: './ip-lab.component.html',
  styleUrls: ['./ip-lab.component.css']
})
export class IpLabComponent implements OnInit {

  constructor(private service: IpService) { }

  public poruka : string = "Primjer IP.";

  ngOnInit(): void {
    this.service.showInfo();
  }

}

```

localhost:4200/etf

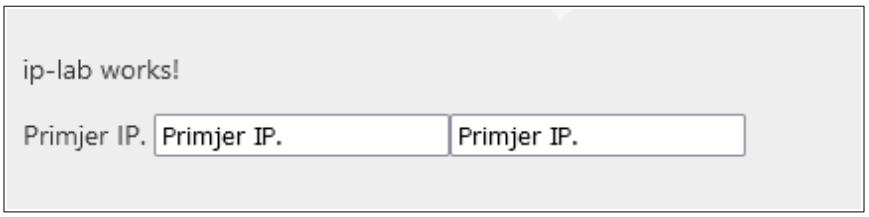


app.modul.ts

```ts
import { NgModule } from '@angular/core';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { IpLabComponent } from './ip-lab/ip-lab.component';

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

ip-lab.component.ts

```ts
<p>ip-lab works!</p>
{{ poruka }}
<input type='text' [value]='poruka'/>
<input type='text' [(ngModel)]='poruka' [ngModelOptions]='{standalone: true}'/>
```



- Промјенама на првом улазу не мења се текст са примјером ни на ком другом пољу, а промјенама на другом пољу мијењају се све три текста са примјером као што је и промењени текст и то на куцање. Мијења се стање и у самом моделу.

Povezivanje događaja

```
import { Component } from '@angular/core';
@Component({
    selector: 'my-app',
    template: `<button (click)="firstEvent($event)">
                Test</button>`
})
export class AppComponent {
    firstEvent(event){…}
}
```

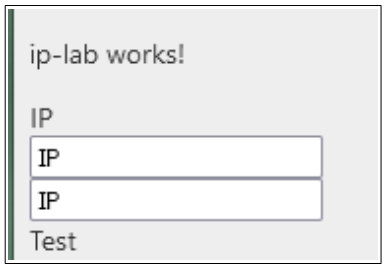
```
<button (click)="firstEvent($event)">Test</button>
```

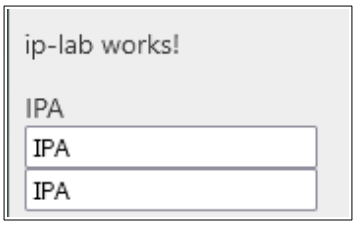
Direktiva *ngIf

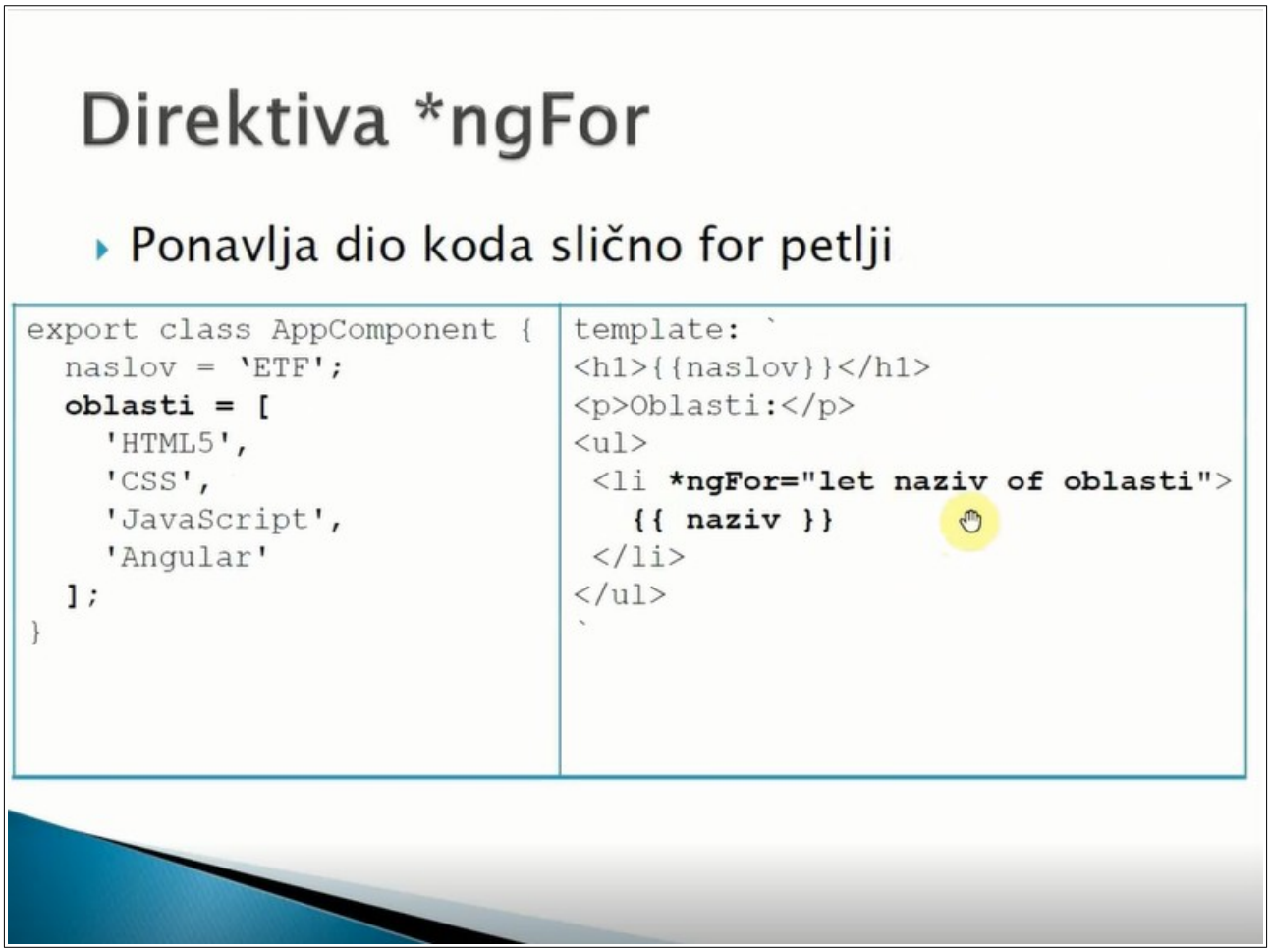
- Prikazuje HTML element ako je uslov ispunjen
- `<div *ngIf="condition">Sadržaj koji će biti prikazan kada je condition==true</div>`

ip-lab.component.html

```
1   <p>ip-lab works!</p>
2   {{ poruka }}<br>
3   <input type='text' [value]='poruka'/><br>
4   <input type='text' [(ngModel)]='poruka' [ngModelOptions]='{standalone: true}'/><br>
5   <div *ngIf='poruka=="IP"'>Test</div>
```





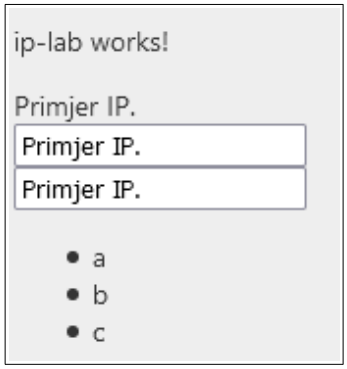


ip-lab.component.ts

```typescript
import { Component, OnInit } from '@angular/core';
import { IpService } from '../services/ip.service';

@Component({
  selector: 'app-ip-lab',
  templateUrl: './ip-lab.component.html',
  styleUrls: ['./ip-lab.component.css']
})
export class IpLabComponent implements OnInit {

  constructor(private service: IpService) { }

  public poruka : string = "Primjer IP.";
  public niz = ["a", "b", "c"];

  ngOnInit(): void {
    this.service.showInfo();
  }

}

```

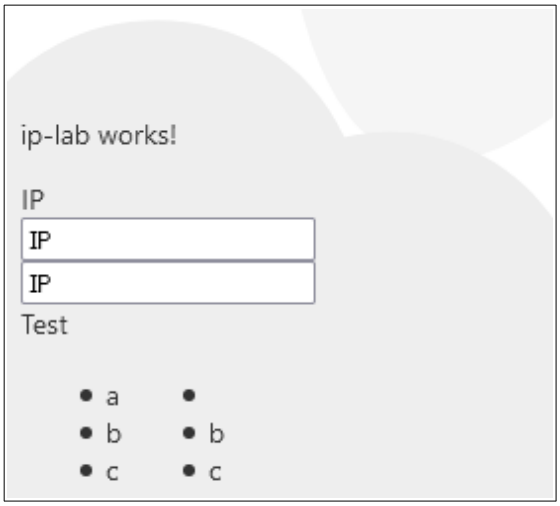
ip-lab.component.html

```html
<p>ip-lab works!</p>
{{ poruka }}<br>
<input type='text' [value]='poruka'/><br>
<input type='text' [(ngModel)]='poruka' [ngModelOptions]='{standalone: true}'/><br>
<div *ngIf='poruka=="IP"'>Test</div>
<ul><li *ngFor='let stavka of niz'>{{stavka}}</li></ul>
```



ip-lab.component.html

```
<p>ip-lab works!</p>
{{ poruka }}<br>
<input type='text' [value]='poruka'/><br>
<input type='text' [(ngModel)]='poruka' [ngModelOptions]='{standalone: true}'/><br>
<div *ngIf='poruka=="IP"'>Test</div>
<table>
    <tr>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span>{{stavka}}</span>
                </li>
            </ul>
        </td>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span *ngIf='stavka!="a"'>{{stavka}}</span>
                </li>
            </ul>
        </td>
    </tr>
</table>
```



Rutiranje

- Mehanizam koji omogućava prelazak sa jednog pogleda na drugi (stranice)
- Otvaranje sadržaja moguće je preko:
  - URL-a stranice
  - Linka
  - Back i Forward opcije u web browser-u
- Osnovna klasa je Router

Rutiranje

- Potrebno je uključiti klase za rutiranje

```
import { RouterModule, Routes } from
  '@angular/router';
```

- U AppModule klasi ili u odgovarajućoj routing klasi potrebno je definisati rute i konfigurisati ruter pomoću `RouterModule.forRoot()` metode

Rutiranje

```
const routes: Routes = [
  {
    path: 'pregled-korisnika',
    component: UsersListComponent
  },
  {
    path: 'korisnik/:id',
    component: UserDetailComponent
  },
  {
    path: '',
    redirectTo: '/pregled-korisnika',
    pathMatch: 'full'
  },
  {
    path: '**', component: PageNotFoundComponent
  } ];
```

Rutiranje

```
@NgModule({
  imports: [
     RouterModule.forRoot(appRoutes),
     //ostali import-i
  ]
  ...
})
export class AppModule { }
```

router-outlet

- Direktiva koja se koristi kao komponenta i određuje mjesto na kojem će se prikazati komponenta za traženu putanju
- <router-outlet></router-outlet>

Rutiranje – linkovi

```
<a routerLink="/pregled-korisnika"
 routerLinkActive="active">
    Korisnici
</a>

<a [routerLink]="['/korisnik', user.id]">
 {{ user.id}}
</a>
```

Rutiranje – parametri

```
constructor(
 private route: ActivatedRoute,
 ... ) { }

ngOnInit() {
 this.route.paramMap
    .subscribe(params => {
        this.userId = params.get('id');
    });
}
```

Angular Material

- https://material.angular.io/

Komponente

- Komponente su dijelovi stranica ili čitave stranice. Imaju ulaze i izlaze.
- Ulazi služe da komponenta dobije neke vrijednosti ili da se poveže sa promjenljivim iz komponente koja koristi datu komponentu (*data binding*)

```
@Input("naziv") public nazivPromjenljive: tip;

<naziv-komponente naziv="promjenljiva">
</naziv-komponente>
```

bg g c ip-prikaz

```
E:\Enviroment\Y27-Angular\ip-test>ng g c ip-prikaz
CREATE src/app/ip-prikaz/ip-prikaz.component.html (24 bytes)
CREATE src/app/ip-prikaz/ip-prikaz.component.spec.ts (641 bytes)
CREATE src/app/ip-prikaz/ip-prikaz.component.ts (286 bytes)
CREATE src/app/ip-prikaz/ip-prikaz.component.css (0 bytes)
UPDATE src/app/app.module.ts (676 bytes)

E:\Enviroment\Y27-Angular\ip-test>
```

ip-prikaz.component.ts

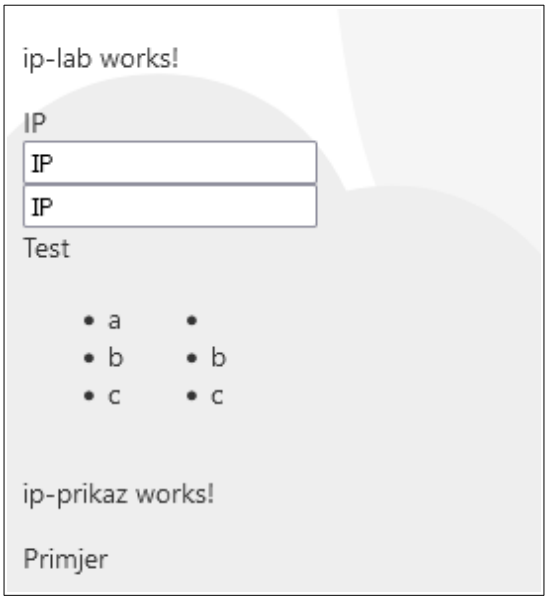
```ts
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-ip-prikaz',
  templateUrl: './ip-prikaz.component.html',
  styleUrls: ['./ip-prikaz.component.css']
})
export class IpPrikazComponent implements OnInit {
  public tekst: string = "Primjer";

  constructor() { }

  ngOnInit(): void {
  }

}
```

ip-prikaz.component.html

```html
<p>ip-prikaz works!</p>
{{ tekst }}
```

ip-lab.component.html

```html
<p>ip-lab works!</p>
{{ poruka }}<br>
<input type='text' [value]='poruka'/><br>
<input type='text' [(ngModel)]='poruka' [ngModelOptions]='{standalone: true}'/><br>
<div *ngIf='poruka=="IP"'>Test</div>
<table>
    <tr>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span>{{stavka}}</span>
                </li>
            </ul>
        </td>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span *ngIf='stavka!="a"'>{{stavka}}</span>
                </li>
            </ul>
        </td>
    </tr>
</table>
<app-ip-prikaz></app-ip-prikaz>
```

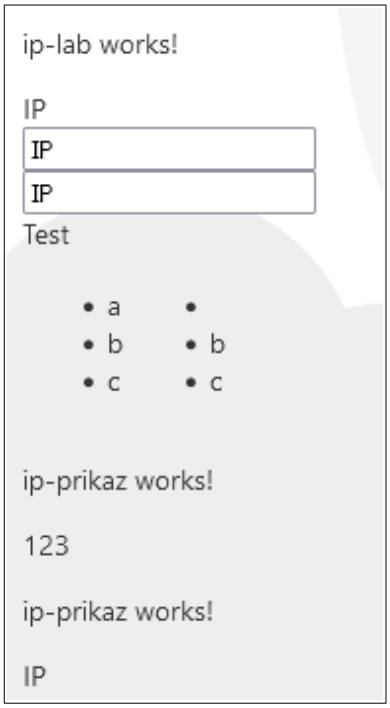


ip-prikaz.component.ts

```ts
import { Component, Input , OnInit } from '@angular/core';

@Component({
  selector: 'app-ip-prikaz',
  templateUrl: './ip-prikaz.component.html',
  styleUrls: ['./ip-prikaz.component.css']
})
export class IpPrikazComponent implements OnInit {
  constructor() { }

  @Input() public tekst: string = "Primjer";

  ngOnInit(): void {
  }

}
```

ip-prikaz.component.ts

```
<p>ip-lab works!</p>
{{ poruka }}<br>
<input type='text' [value]='poruka'/><br>
<input type='text' [(ngModel)]='poruka' [ngModelOptions]='{standalone: true}'/><br>
<div *ngIf='poruka=="IP"'>Test</div>
<table>
    <tr>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span>{{stavka}}</span>
                </li>
            </ul>
        </td>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span *ngIf='stavka!="a"'>{{stavka}}</span>
                </li>
            </ul>
        </td>
    </tr>
</table>
<app-ip-prikaz tekst='123'></app-ip-prikaz>
<app-ip-prikaz [tekst]='poruka'></app-ip-prikaz>
```

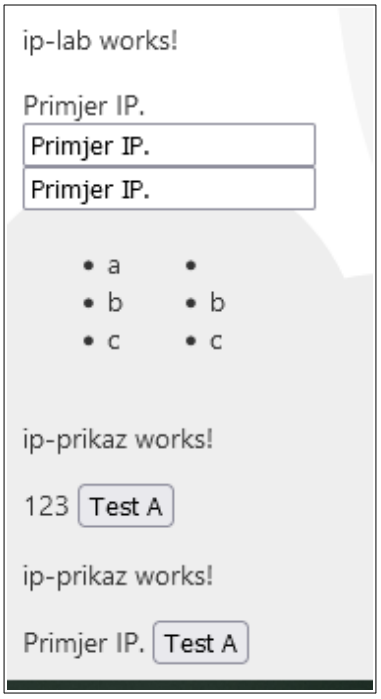


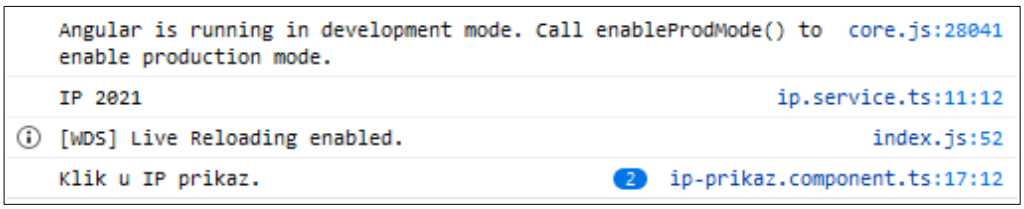
ip-prikaz.component.ts

```ts
import { Component, Input , OnInit } from '@angular/core';

@Component({
  selector: 'app-ip-prikaz',
  templateUrl: './ip-prikaz.component.html',
  styleUrls: ['./ip-prikaz.component.css']
})
export class IpPrikazComponent implements OnInit {
  constructor() { }

  @Input() public tekst: string = "Primjer";

  ngOnInit(): void {
  }

  onClick(): void{
    console.log('Klik u IP prikaz.')
  }
}

```

ip-prikaz.component.html

```html
<p>ip-prikaz works!</p>
{{ tekst }}
<button (click)='onClick()'>Test A</button>
```





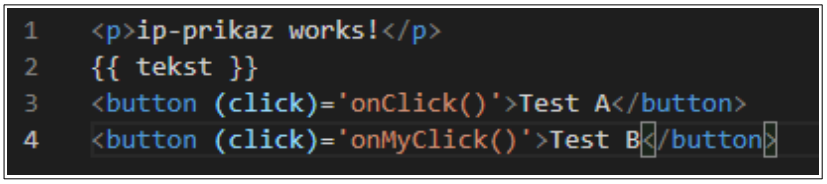
ip-lab.component.html

```
<p>ip-lab works!</p>
{{ poruka }}<br>
<input type='text' [value]='poruka'/><br>
<input type='text' [(ngModel)]='poruka' [ngModelOptions]='{standalone: true}'/><br>
<div *ngIf='poruka=="IP"'>Test</div>
<table>
    <tr>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span>{{stavka}}</span>
                </li>
            </ul>
        </td>
        <td valign='top'>
            <ul>
                <li *ngFor='let stavka of niz'>
                    <span *ngIf='stavka!="a"'>{{stavka}}</span>
                </li>
            </ul>
        </td>
    </tr>
</table>
<app-ip-prikaz tekst='123'></app-ip-prikaz>
<app-ip-prikaz [tekst]='poruka'></app-ip-prikaz>
<app-ip-prikaz tekst='My Event' (akcija)='onPrikazEvent($event)'></app-ip-prikaz>
```

ip-lab.component.ts

```ts
import { Component, OnInit } from '@angular/core';
import { IpService } from '../services/ip.service';

@Component({
  selector: 'app-ip-lab',
  templateUrl: './ip-lab.component.html',
  styleUrls: ['./ip-lab.component.css']
})
export class IpLabComponent implements OnInit {

  constructor(private service: IpService) { }

  public poruka : string = "Primjer IP.";
  public niz = ["a", "b", "c"];

  ngOnInit(): void {
    this.service.showInfo();
  }

  onPrikazEvent(event: any){
    console.log('IP Lab: ', event);
  }
}

```

ip-prikaz.component.html

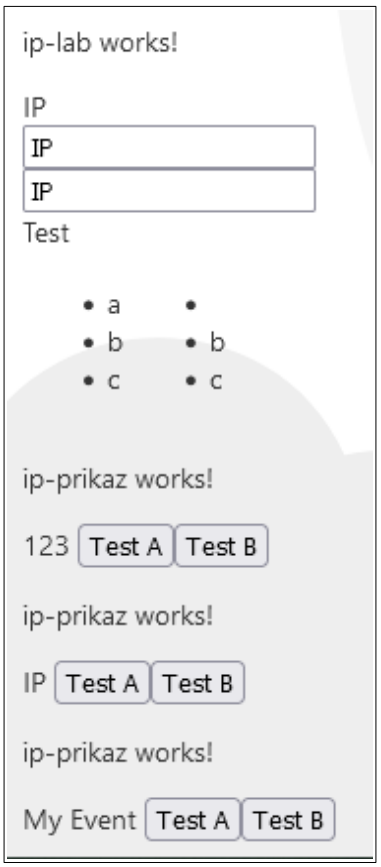


```html
<p>ip-prikaz works!</p>
{{ tekst }}
<button (click)='onClick()'>Test A</button>
<button (click)='onMyClick()'>Test B</button>
```

ip-prikaz.component.ts

```typescript
import { Component, Input , OnInit, Output, EventEmitter } from '@angular

@Component({
  selector: 'app-ip-prikaz',
  templateUrl: './ip-prikaz.component.html',
  styleUrls: ['./ip-prikaz.component.css']
})
export class IpPrikazComponent implements OnInit {
  constructor() { }

  @Input()  public tekst: string = "Primjer";
  @Output() public akcija: EventEmitter<any> = new EventEmitter();

  ngOnInit(): void {
  }

  onClick(): void{
    console.log('Klik u IP prikaz.')
  }

  onMyClick(): void{
    console.log('Klik u IP prikaz.')
    this.akcija.emit('Test');
  }
}

```

```
[WDS] Live Reloading enabled.                                   index.js:52
Angular is running in development mode. Call enableProdMode() to  core.js:28041
enable production mode.
IP 2021                                                  ip.service.ts:11:12
Klik u IP prikaz.                                  ip-prikaz.component.ts:22:12
IP Lab:  Test                                        ip-lab.component.ts:21:12
Klik u IP prikaz.                                2 ip-prikaz.component.ts:22:12
```


ip-lab works!
IP
IP
IP
Test
a
b
c
b
c
ip-prikaz works!
123 Test A Test B
ip-prikaz works!
IP Test A Test B
ip-prikaz works!
My Event Test A Test B

Komponente

- Izlazi služe da u komponenti koja koristi datu komponentu omogući obradu nekog događaja koji se desi u datoj komponenti. Događaj može poslati vrijednost komponenti u kojoj se nalazi.

```
@Output("dogadjaj") public dogadjaj = new
  EventEmitter<tip>();
...
dogadjaj.emit(vrijednost);

<naziv-komponente (dogadjaj)="obradiDogađaj($event)">
</naziv-komponente>
```

Forme

```
<form [formGroup]="form">
  <mat-form-field>
     <input matInput placeholder="Ime"
     required
     formControlName="firstName"
     autocomplete="off" />
  </mat-form-field>

...
</form>
```

Forme

```
public form: FormGroup;
…
constructor(public formBuilder:
  FormBuilder,
…
ngOnInit() {
  this.form = this.formBuilder.group({
    firstName: [this.user.firstName,
      Validators.required],
    //početna vrijednost, validatori. Može i bez ovoga.
    …
  });
}
```

Servisi

- Dependency injection (DI) omogućava da neka klasa dobije potrebne resurse (servise ili druge klase) kada zatrebaju, bez kreiranja vlastitih objekata tih klasa
- Angular servisi su zasnovani na DI
- Servisi se koriste za čuvanje biznis logike, pristupe vanjskim resursima ili za čuvanje ili dijeljenje podataka
- `ng generate service modul/ime`

Servisi

```
import { Injectable } from @angular/core';
@Injectable({
 providedIn: 'root',
})
export class TestService{
 getData() {
    return […];
 }
 //funkcije…
}
```

Servisi – upotreba

```
constructor(service: TestService) {
 this.data = service.getData();
}
```

- Svi potrebni servisi se dodaju u konstruktoru

HTTP servis

- https://angular.io/guide/http
- Služi za pristup resursima preko Interneta
- Baziran na XMLHttpRequest
- Potreban import modula:

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

- HTTPClient servis omogućava kreiranje HTTP zahtjeva
- Uključuje se u konstruktor

```
constructor(private http: HttpClient) { }
```

HTTP servis

- Upotreba HTTPClient:

```
getConfig() {
  return this.http.get(this.configUrl);
}

showConfig() {
  this.configService.getConfig()
    .subscribe((data) =>
      console.log(data)
  );
}
```

HTTP servis

- Slanje POST zahtjeva

```
addHero (hero: Hero): Observable<Hero> {
  return this.http.post<Hero>(
     this.heroesUrl, hero, httpOptions)
  .pipe(
     catchError(
          this.handleError('addHero', hero))
     );
}
this.heroesService.addHero(newHero)
  .subscribe(hero => this.heroes.push(hero));
```

Pipes

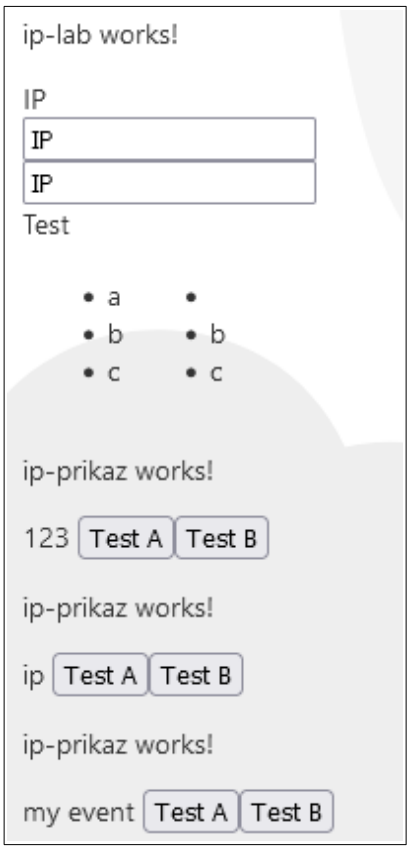
- Mehanizam formatiranja sadržaja

```
{{title | uppercase}}
```

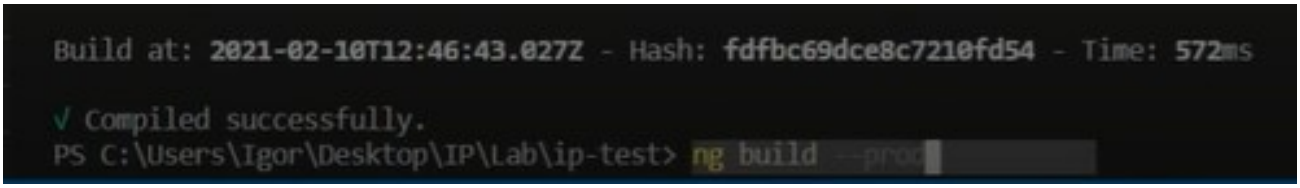
- Postoji 7 ugrađenih mehanizama, a mogu se napraviti i vlastiti:
  - Lowercasepipe – pretvara tekst u mala slova,
  - Uppercasepipe – pretvara tekst u velika slova,
  - Datepipe – formatira datum `{{todaydate | date:'d/M/y'}}`
  - Currencypipe – formatira vrijednost u valutu: `{{6589.23 | currency:"USD":true}}` – prikaže i znak valute)
  - Jsonpipe – formatira podatak kao JSON
  - Percentpipe – formatira podatak kao procente: `{{00.54565 | percent}}`
  - Decimalpipe – formatira broj: `{{454.78787814 | number: '3.4-4'}}`
  - Slicepipe – prikazuje dio vrijednosti: `{{months | slice:2:6}}`

ip-lab-component.ts

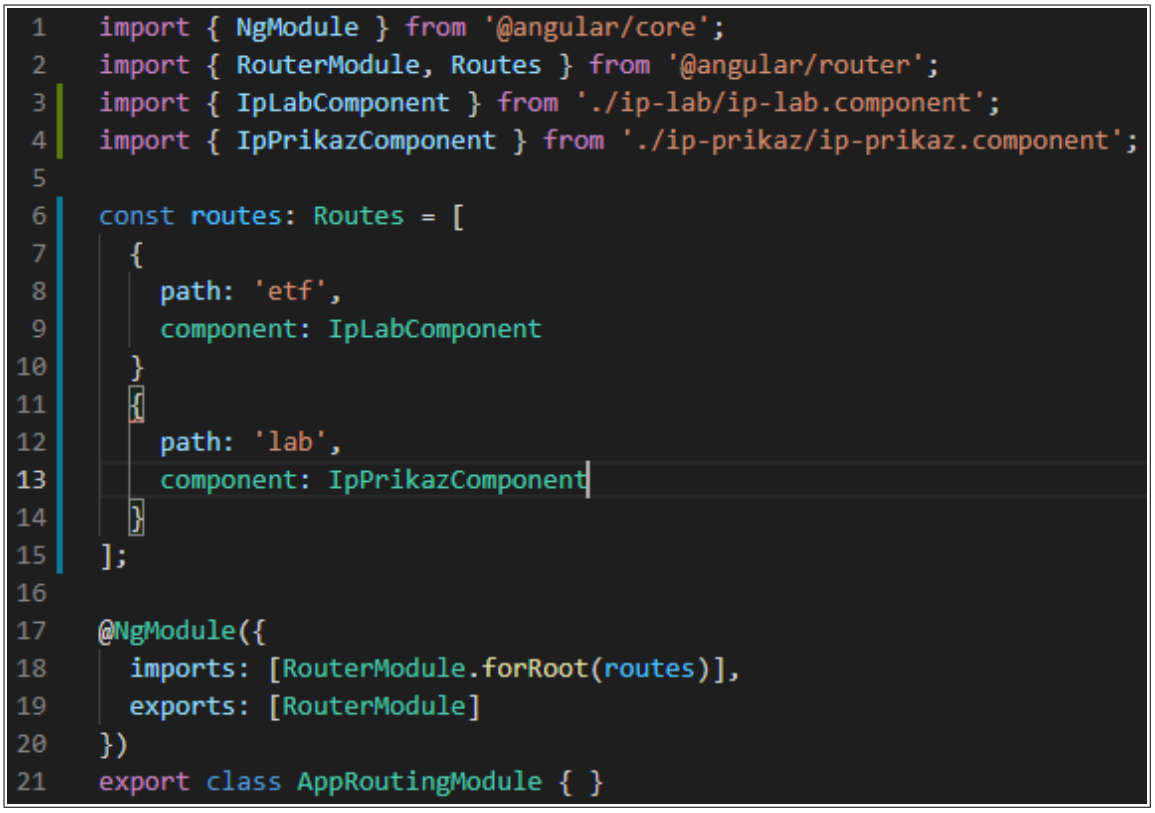
```
1   <p>ip-prikaz works!</p>
2   {{ tekst | lowercase }}
3   <button (click)='onClick()'>Test A</button>
4   <button (click)='onMyClick()'>Test B</button>
```



ng build -prod



app-routing-component.ts



```ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { IpLabComponent } from './ip-lab/ip-lab.component';
import { IpPrikazComponent } from './ip-prikaz/ip-prikaz.component';

const routes: Routes = [
  {
    path: 'etf',
    component: IpLabComponent
  }
  ,{
    path: 'lab',
    component: IpPrikazComponent
  }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

localhost:4200/lab