Angular:

Introduction of the Angular/History of Angular

Why Angular:46

Features of Angular

Angular Architecture

Advantages of Angular

Limitations

Angular learning curve

Various companies that are angular

Components :

Services

Binding

Forms

Bootstrap

Routing

Route guards

Wild card routes

Directives:

Introduction of the Angular/History of Angular

Single page application: JS and TS: Front end framework: 46: FEF:

What is Angular?

1. AngularJS: was open source JS framework: JS
2. It is completely written in TS:
3. Primary aim is only to develop SPA
4. It uses HTML syntax to express your Application clearly:
5. It is maintained by Google:

Features of Angular:

1. DOM:
2. TYPESCRIPT:
3. Data Binding: user name: PW: login.html template: -> validated: TS(component)-processing : validate : TS send some response to the user:
4. Testing:

**History of Angular**:

1. AngularJS: 1.0/1.x: Oct 2010: completely written in JS: TS
2. Angular 2.0: 14th Sept 2016: TS: ES-6: TS, DART, ES-6 JavaScript:
3. Angular 3.0: created but never published or never released : versioning: @angular /core, @angular/compiler, router libraries:
4. 4.0: March 2017: Fastest compilation
   1. Better bug fixing
   2. TS 2.1 and 2.2:
   3. Else block for \*ngIf: (Structural directive In Angular)
5. 5.0: Nov 2017:
6. 6.0: May 2018: ng update and ng add:
7. Angular team decided to do one thing: after each and every 6 month: they are releasing the new version of Angular:

Angular Architecture:

Full-fledged MVC architecture:

Model: All the data related logic

View: UI logic

Controller: Brain of your application: interface between Model and view:

TV set : watch Set up: TataSky, Airtel:

TV set: view:

Dish / Setup box: What to show and what to not: 300 channel : Control : DB

Satellite: Model: fetch UN and PW: DB: try match : response to Controller: use:

Advantages of Angular

1. Modularity
2. Testability
3. SIP: **Lazy Loading:**

Limitations

Angular’s learning curve:

Migration:

Angular: project: 14: migration Underflow:

Angular fw is much complex as :

Various companies that are angular

Google: NIKE, HBO, Sony, UpWork, Forbes::: General Motors:

Components based things:

Common features of Component:

1. They are building block of angular application
2. We have a root component: app.component.ts
   1. I can have any no of components in my App.
   2. 1000 of components but only one root component is there: per component only one html file can be there
3. Component is nothing but a TS class: Decorator: @Component decoroators:

Angular Application: you need installation on system: TSC :

Angular:

Injector service in Angular:

**Angular Forms:**

**Template Driven Form**: **ngModel** input type=”text/password”: validation

Validation: 9999999:

Name: Remi

Email; [nil@cbb.co](mailto:nil@cbb.co): maximum coding we do on html: simplest way:

**Reactive Form**: maximum coding we do on TS: if

**Template Driven Form:**

**Ng new SLDemo**

Bootstrapping: Angular: using different components:

Integrate Bootstrap in our Angular: //installed in our project; repetitive:

Bootstrap: npm install bootstrap

jquery : npm install jquery

3 : group: under same folder

We need to add references for javascript and CSS:

**Routing**:

Home, Login , Registration

Step1: index.html: optional:

  <base href="/">

Step : 2: Optional: open app.module.ts:

AppRoutingModule

Step 3 : **Compulsory**:

app.routing.module.ts

Step 4: : app.routing.module.ts: compulsory:

const routes: Routes = [

{path:'home', component:MainpageComponent },

{path:'login',component:LoginComponent},

{path:'registration',component:RegistrationComponent}];

Step: 5: Compulsory

In nav bar; add 2 attributes in anchor tag:

routerLink="home" routerLinkActive="active"

Step: 6: Compulsory

App.component.html:

<router-outlet></router-outlet>

Route guard:

Amazon:

Login: UN and PW:

**Normal user** or **Admin**:

Authenticated ; Correct Un and PW: nilesh and 12345

User or Admin:

Authorization: : see the product: buy: cancel order:

Admin: to perform more than : Add product: update product:

Reactive form:

Maximum in the ts file:

\*ngIf: in template

Steps:

1. Define normal form: form tag
2. **Import ReactiveFormsMudule**: app.module.ts: Import array
3. **Component file**: .ts: import { FormGorup, FormControl }
   1. FormGorup:
   2. FormControl:
4. **Create FormsGroup object**: add FormControls:
5. **Template**
6. Ts. File we need to do some validations:
   1. Import Validator
   2. Apply validators
   3. Define getter method:
   4. Show error message: disable button
7. Reactive form:
8. **Route guard: prohibit the user from accessing a particular: front end in which restricting the user: Spring Security:**
9. **Types of Route guards : Interfaces:** 
   1. **CanActivate**
   2. **CanActivateChild**
   3. **CanDeactivate**
   4. **CanLoad**
10. Wild card Routes: any link: redirect user to the specific :
11. **Directives in Angular**:
    1. **1) Component directive**: 1 )one way binding 2) two way binding
       1. Ng g c emp : providing /giving some help to render something on screen html
       2. They are providing some data to the templates: functions, values
       3. How they provide data to the template;
          1. Binding :
             1. One way binding :

In one way binding data flows in one direction

Component to html:

Interpolation: {{}}

Property binding

Html to component:

Event binding

* + - * 1. Two way binding:

~~Component directive~~: We are not using: but we are using Attribute Directive: **[(ngModel)] banana expression**

* + - 1. Between ts file to html or Vice versa:

2) Structural directives:

1) \*ngFor

2) \*ngIf

3)\*ngSwitchCase:

<div [ngSwitch]=expression>

<p \*ngSwitchCase 1> statements</p>

<p \*ngSwitchDefault> statements</p>

<p \*ngSwitchCase 2> statements</p>

<p \*ngSwitchCase 3> statements</p>

</div>

3) Attribute directive :

Attribute: for html tags:

1. [(ngModel)]// 2 way binding
2. ngStyle: Angu
3. ngClass<p class=”primary”>n dfdsf hkjfsdkfh dkfh</p>

inline CSS:

2hrs;

Excerise: only one emp: Array: pushing data in that array: do same appl for 3 emp:

Bootstrap and routing:

Login functionality

1. Pipes in angular: Inbuilt pipes and custom pipes
2. http: How you can send a request from you Angular to SB: SB: localhost:8080:
3. Project:

SB Application: Cross Origin: Dummy Service: fetching some data from Your DB: inserting some data into your DB:

Entity : Employee: id,

Controller: user:

Service: CRUD:

Angular: How you can send a request from Angular to the SB: 4200: