# Forms

## Template Forms

Import FormsModeule

ngModel, ngModelGroup, ngForm, ngFormControl are built in and we can use validation templates like required, minLenght , maxlenght etc.

## Reactive Templates

Import ReactiveFormsModeule

Need to define the Form, Form Group, Form Control are to be defined in our own component.

Can be defined either using FormControl or FormBuilder(reduce code)

Http Module:

Import httpModule

Fake HTTP Service: Jsonplaceholder.typicode.com

**Optimistic** – Observable – Will not invoke the service until it is subscribed – lazy fetching.

**Pessimistic** -> promises – hit the backed calls and wait for response – eager fetching

**Snapshot –** is observable without subscribe and only the current object values will be loaded.

# Routing:

Import – RouterModule.forRoot() – forRoot is static method in RouterModule

Imports [

ROuterModule.forRoot()([

{ path: <path>, component: <ComponentName>

])

]

Import – RouterModule.Child() –

More specific routing At the bottom and more specific routes at the top.

Don’t use <a href=””> for routing – it re-download all the scripts and page referesh happen – Use RouterLink instead <a routerLink=””>

Use <a [routerLink]=”[path, param]”> for dynamic link generation.

Use ParamMap to get parameters and queryParameter is get query params.

Use “rxjs/add/observable/combineLatest” for combining multiple observable and subscribe. And use switch map for object conversion.

combineobs = Observable.combineLatest([

This.route.ParamMap,

This.route.queryParamMap ]);

Combineobs.subscribe(combined => {

Let var1 = combined[0].get(<paramMapAttribute>);

Let var1 = combined[1].get(<QueryparamMapAttribute>);

});

# Authorization & Authentication

Install Angular2-jwt for JWT support

Use localStorage.putItem/getItem to store token and access anywhere in the application

TokenNotExpired() – will automatically get the token from local storage name called “token” and checks it expiration. Returns true/false.

Let jwtHelper = new JwtHelper() – provides decodeTOken, isExpired, getExpirationDate etc.

## CanActivate

We can disable the links in the UI and user type the route URI page will get displayed

To fix the vulnerability to direct access route through browser.

Create service andimplement this CanActivate interface and override method canActivate() and return true if authorized otherwise you can redirect the use to login page using router module. And multiple CanActivate services can be added.

Also each component where that should be protected should have canActivate: [new Service which implemented CanActivate].

## AuthHttp:

The angular http we need to create header and add Authoorization header and create httpoptions to add header then we need to send this options to http method calls to the backend API;s.

TO avoid all those things simply AuthHttp from angular2-jwt which inbuilt add Authorization heady by reading token from local storage.

# Deployment package Optimization

1. Minification – removes comments and extra spaces
2. Uglification – change long variable & class names.
3. Bundling –
4. Deadcode Elimination –
5. AOT compilation – Ahead of Time compilation

All the above techniques are applied in ng build –prod

JIT – is not suitable for nod development enviroronments and it every users it compile when needed and lead to performance issues. It need vendor-bundle which contains 200+MB for support JIT. So AOT is good for production.

## TSLINK

Used for reporting code quality issues. Nglint is command or install vscode plugin to fix the issues while coding itself

Animation:

Animate.css has predefined animations, install animate.css to use it in angular.

PolyFills:

Helps to use modern features to use in old browsers – configuration in polyfills.js