# Routing PoC

# Introduction

In this app, there are three sections

1. Home
2. Servers
3. View and Edit Servers
4. A Service is used to load and update Servers
5. Users
6. View Users

This app will display above three sections using Routing

## Routes

Represent a route configuration.

Routes = Route[ ]

## Route

1. Simple Configuration
2. Multiple Outlet
3. Wild Card
4. Redirects
5. Empty Path
6. Matching Strategy
7. Component less Routes
8. Lazy Loading

Package –

**@angular/router**

const routes: Routes = [

{

path: ‘team/:id/player/:playerId’, **-- http://localhost:4200/team/23/player/123?x =y#placeholder**

pathMatch: ‘prefix’ | ‘full’ (Default is prefix), // needed while redirecting

redirectTo: ‘path’,

component: TeamComponent,

loadComponent: () => {},

canActivate: Array<CanActivateFn | any>, //if the user allowed to see the component

canMatch: Array<CanMatchFn>, // if the user allowed to match this path,

canActivateChild: Array<CanActivateChildFn | any>, // access right on child component

resolve?: ResolveData, // look up data resolvers

children: Routes, // array of child routes

loadChildren: LoadChildren

} ]

# How to trigger routing?

1. User click some link - RouterLink

<a routerLink=”/user”, routerLinkActive=”[class1, class2]> ….</a> // UrlTree

1. Programmatically –

Router.navigate(

[‘servers’, id, ‘edit’],

{queryParams: {allowEdit: 1}, fragment: ‘loading’}

Router

A service that provides navigation among views and URL manipulation

ActivatedRoute

Inject currently loaded route.

access to parameter, path variables, current path etc…

* activatedRoute.snapshot.params[‘id’]
* activatedRoute.snapshot.queryParams[‘id’]
* activatedRoute.params.subscribe 🡪 if user remain on same component after redirect then component won’t be created again
* activatedRoute.queryParams.subscribe 🡪