Google Developer Expert

International Speaker


NG
CONF


AC
ANGULAR
CONNECT


Riga
DevDay


FullStack
The conference on JavaScript, Node and Internet of Things
13–15 JUL
2016


DON'T MISS:
droidcon
LONDON
27-28 OCT
2016
LONDON


VOXXEDDAYS
BELGRADE


js.talks();


QCon
LONDON


Jfokus


INTERNET &
MOBILE
WORLD 2016


{ }
Frontend
Union
Conf


ff


JAZOON
TechDays


GDG DevFest
The Netherlands


GDG DevFest
Ukraine 2016


GDG DevFest
London 2016

Master of Ceremonies


Angular 2 Trainer

Google Developers Experts Writer Edit

Blogger

gsans

Coding is fun | Coded something awesome today? | #Angular & #JavaScript are the new kings! M...

Aug 14 · 8 min read

GraphQL and the amazing Apollo Client

Explore an Application built using React and Angular 2

Community Leader

AKERS TICKETS SPONSORS CRUI

NG-CRUISE

— PRESENTED BY NG-CONF —

ANGULAR CRUISE

PRESENTED BY NG-CONF

As Jeff Cross' beard has stated,

we're officially jumping the shark

May 29, 2017!

SIGNUP

SPONSOR

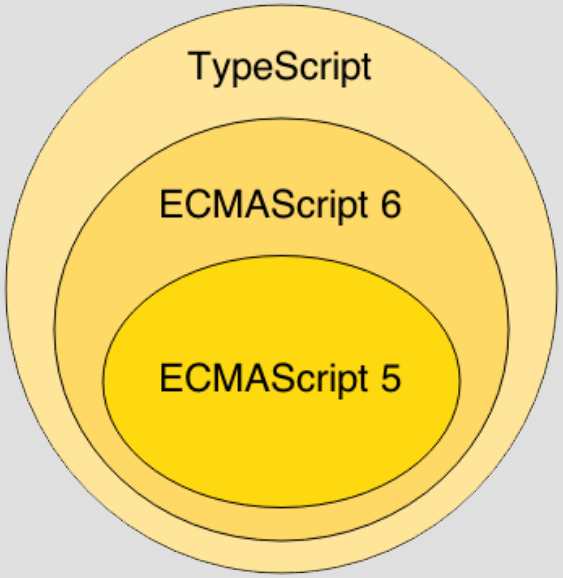
Some days before Jfokus...

Angular 2

Features

- Latest Web Standards
- Great Developer Experience
- Simple
- Lightning fast
- Works everywhere

TypeScript



ES6 (ES2015)

- Classes, modules, arrow functions

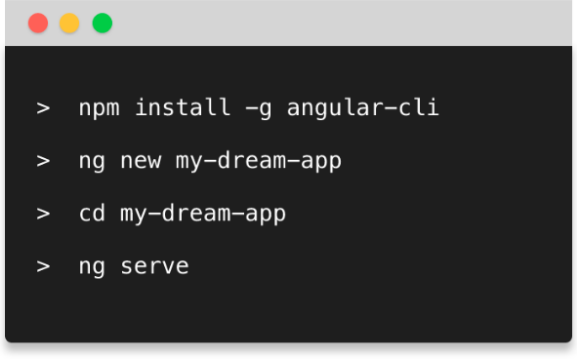
TypeScript

- Types, decorators, generics, interfaces
- Great editor support

TypeScript IDE Support

ATOM

Modern Tooling



Angular CLI

Protractor

Angular Augury

by RANGLE.IO

Language Services

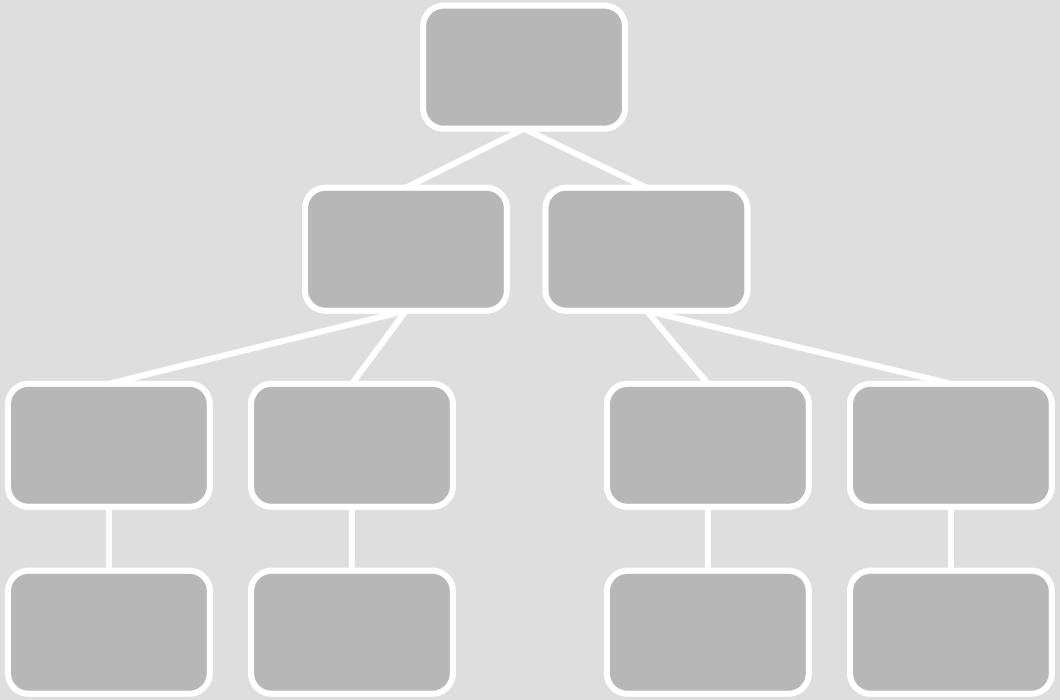
angularexpo.com

Application Design

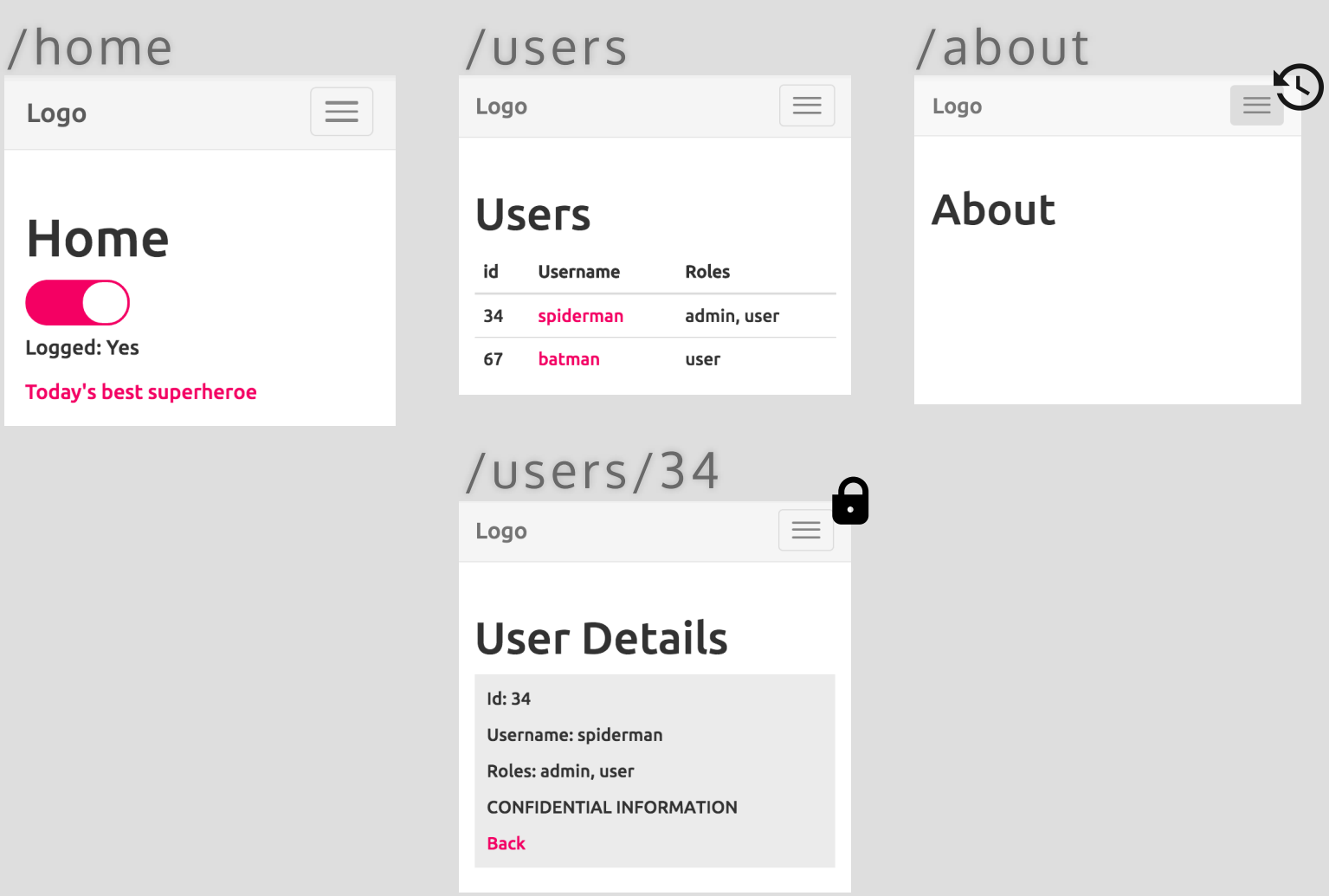
Requirements

- Network resilience
- Ask before navigating
- Restrict access
- Lazy loading

Components Tree



source: blog


/home
Logo
Home
Logged: Yes
Today's best superheroe
/users
Logo
Users
id
Username
Roles
34
spiderman
admin, user
67
batman
user
/about
Logo
About
/users/34
Logo
User Details
Id: 34
Username: spiderman
Roles: admin, user
CONFIDENTIAL INFORMATION
Back

demo

Bootstrap

Bootstrapping

- Angular Application instantiation
- Root Module (AppModule)
- Global Dependencies
  - Router, Http, Services
  - Global Values
  - Vendor dependencies

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <script src="https://unpkg.com/zone.js/dist/zone.js"></script>
    <script src="https://unpkg.com/zone.js/dist/long-stack-trace-zone.js
    <script src="https://unpkg.com/reflect-metadata@0.1.3/Reflect.js"></
    <script src="https://unpkg.com/systemjs@0.19.31/dist/system.js"></sc
    <script src="systemjs.config.js"></script>
    <script>System.import('app');</script>
  </head>
  <body>
    <my-app>
      Loading...
    </my-app>
  </body>
</html>
```

Bootstrap

```
// main.ts
import { platformBrowserDynamic } from '@angular/platform-browser-dynam
import { AppModule } from './app.module';

platformBrowserDynamic().bootstrapModule(AppModule);
```

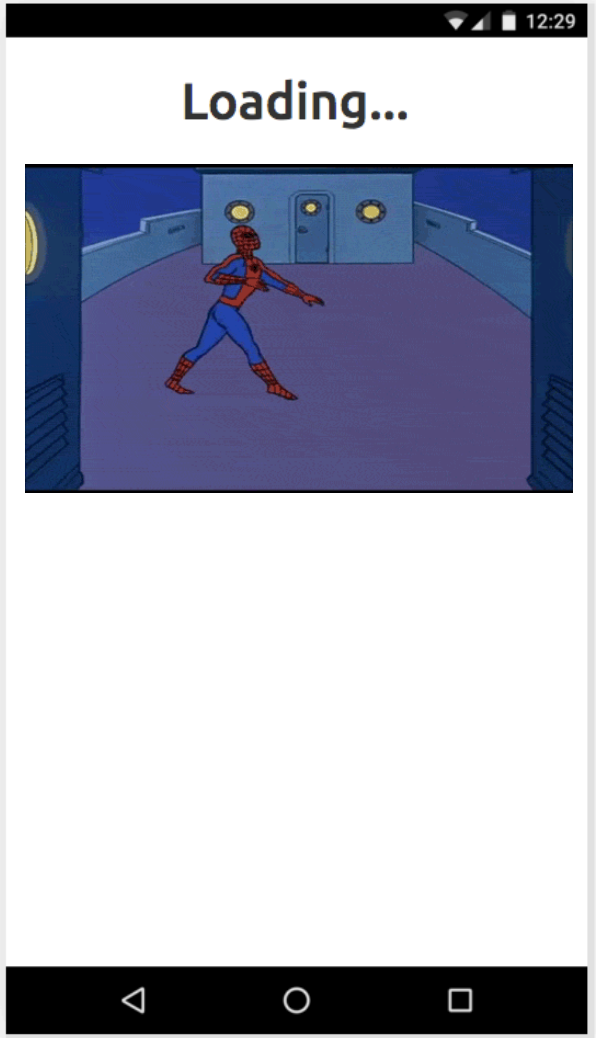
Bootstrap

```
// app.module.ts
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent } from './app.component';

@NgModule({
  imports: [ BrowserModule, UsersModule, appRouting ],
  declarations: [ AppComponent, Home, ... ],
  providers: [ LoginService, AuthCanActivate ],
  bootstrap: [ AppComponent ]
})
export class AppModule { }
```

app.component.ts

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

@Component({
  selector: 'my-app', // <my-app>Loading...</my-app>
  template: `...`
})
export class App {
  constructor() { }
}
```

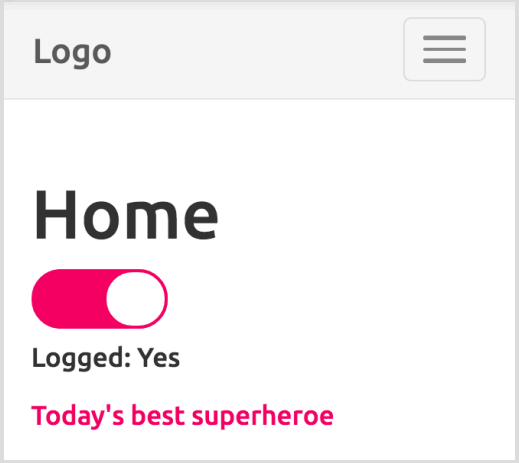

12:29
Loading...

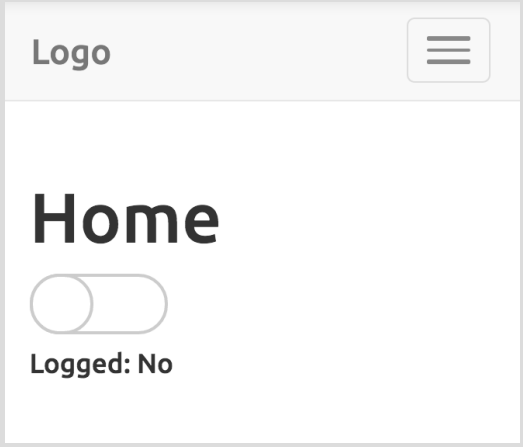

12:29
Loading...


NEAT

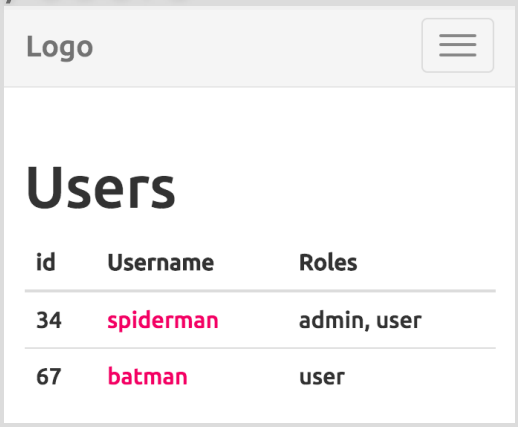
Services

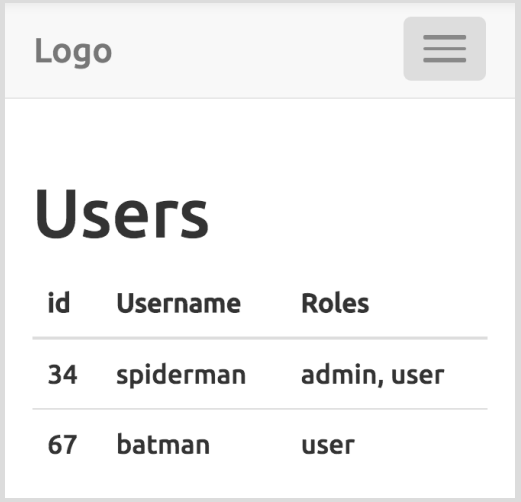
/home


Logo
Home
Logged: Yes
Today's best superheroe


Logo
Home
Logged: No

/users


Logo
Users
id
Username
Roles
34
spiderman
admin, user
67
batman
user


Logo
Users
id
Username
Roles
34
spiderman
admin, user
67
batman
user

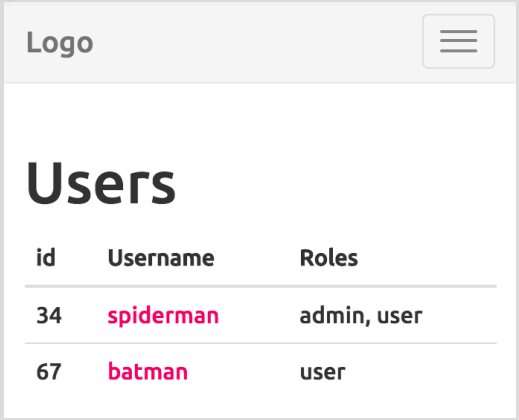
login.service.ts

```
import { Injectable } from '@angular/core';

@Injectable()
export class LoginService {
  authorised = false;

  authorise(value) {
    this.authorised = value;
  }
}
```

home.component.ts

```
// home.component.ts
import { Component } from '@angular/core';
import { LoginService } from './login.service';

@Component({
  template: `
    <h1>Home</h1>
    <input #switch type="checkbox"
      [checked]="loginService.authorised"
      (click)="change(switch.checked)">
    <div *ngIf="loginService.authorised">
      <a>Today's best superheroe</a>
    </div>`
})
export class Home {
  constructor(private loginService:LoginService) { }

  change(checked){
    this.loginService.authorise(checked);
  }
}
```

/users


Logo
Users
id
Username
Roles
34
spiderman
admin, user
67
batman
user

users.json


```
[{
  "id": 34,
  "username": "spiderman",
  "roles": ["admin", "user"]
}, {
  "id": 67,
  "username": "batman",
  "roles": ["user"]
}]
```

users.service.ts

```
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';
import 'rxjs/add/operator/map';
import 'rxjs/add/operator/retryWhen';
import 'rxjs/add/operator/delay';

@Injectable()
export class UsersService {
  constructor(private http:Http) { }

  get(){
    return this.http.get('api/users.json')
      .map(response => response.json())
      .retryWhen(errors => errors.delay(2000));
  }
}
```

users.component.ts

```
import { Component } from '@angular/core';
import { UsersService } from './users.service';

@Component({
  selector: 'users',
  template: `
    <h1>Users</h1>
    <table class="table">
      <tr *ngFor="let user of users">
        <td>{{user.username}}</td>
      </tr>
    </table>`
})
export class Users {
  constructor(private service:UsersService){
    service.get().subscribe(users => this.users = users);
  }
}
```

Router

Main Contributor

@victorsavkin

Main Features

- Url-based navigation
- Flexible routing
  - Child and auxiliary routes
- Navigation Guards
- Lazy loading and preloading
- Resolve

ANGULAR 2 ROUTER

VICTOR SAVKIN

Setup Router

Dependencies

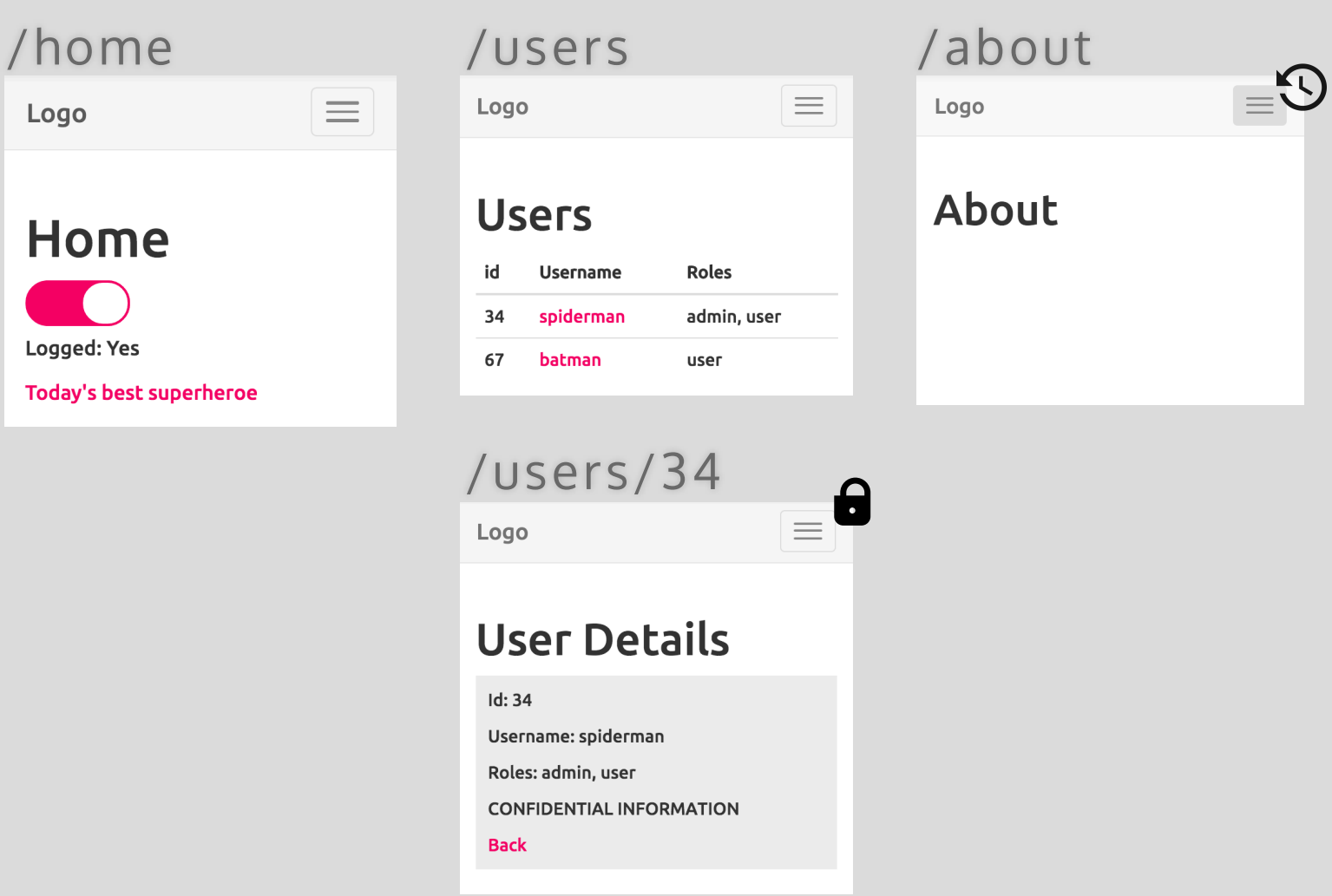
- Choose Location Strategy
- Setup routes
- Include providers and directives
  - *`RouterModule.forRoot(routes)`*
  - *`RouterModule.forChild(routes)`*

Location Strategies

- Hash Location
  - Eg: #/home, #/users/34
- Path Location (default)
  - Requires <base href=.../>
  - Eg: /home, /users/34

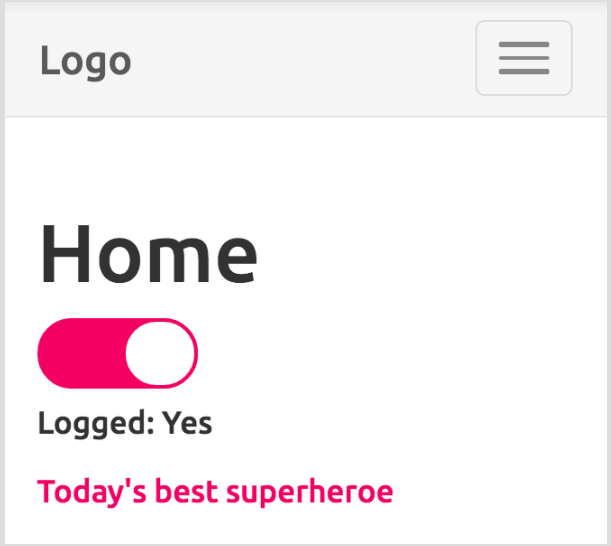
Bootstrap

```
// app.module.ts
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppRouting } from './app.routing';

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


/home
Logo
Home
Logged: Yes
Today's best superheroe
/users
Logo
Users
id
Username
Roles
34
spiderman
admin, user
67
batman
user
/about
Logo
About
/users/34
Logo
User Details
Id: 34
Username: spiderman
Roles: admin, user
CONFIDENTIAL INFORMATION
Back

Defining Routes

```
// app.routing.ts
import { Routes, RouterModule } from '@angular/router';
import { Home } from './home.component';
import { About } from './about.component';
import { NotFound } from './notfound.component';

const appRoutes: Routes = [
  { path: '', redirectTo: 'home', pathMatch: 'full' },
  { path: 'home', component: Home },
  { path: '**', component: NotFound }, //always last
];

export const AppRouting = RouterModule.forRoot(appRoutes, { useHash: tr
```

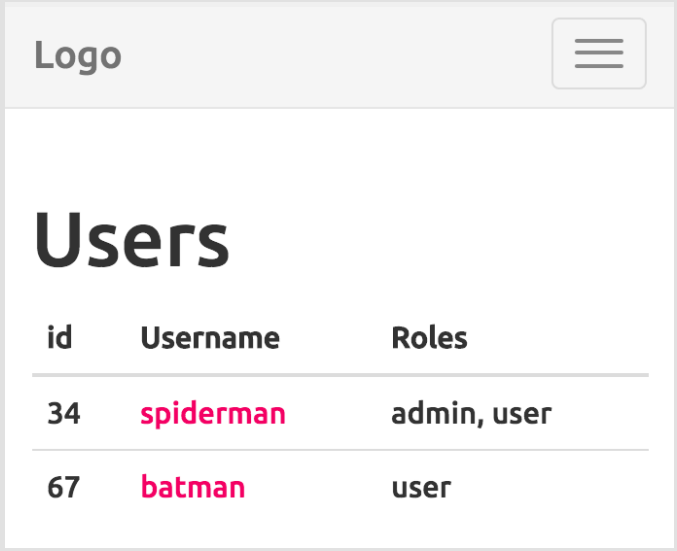

Logo
Home
Logged: Yes
Today's best superheroe

router-outlet

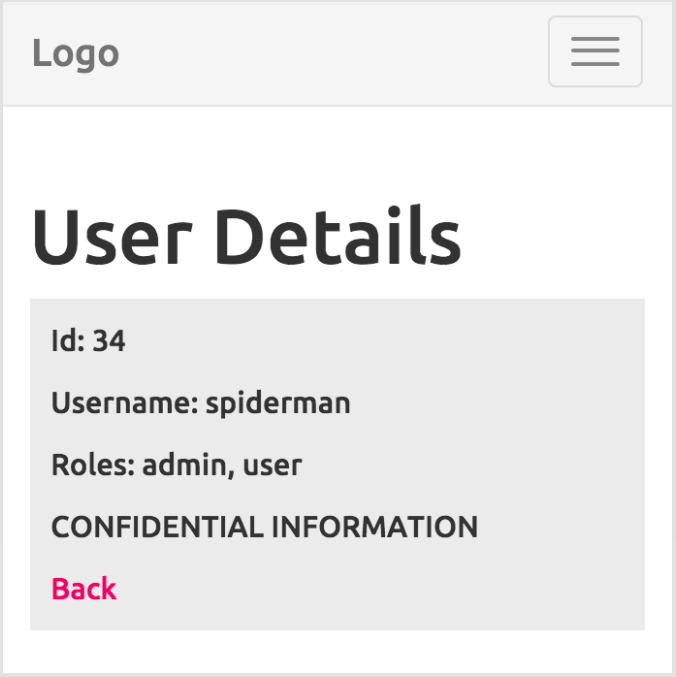
```
// app.component.ts
@Component({
  selector: 'my-app',
  template: `
    <nav></nav>
    <main>
      <router-outlet></router-outlet>
    </main>`
})
export class App { }
```

Child Routes

users
#/users
Users



users/:id
#/users/34
User



Child Routes

```
// users.routing.ts
import { Routes, RouterModule } from '@angular/router';
import { Users } from './users.component';
import { User } from './user.component';

const usersRoutes: Routes = [
  { path: 'users', component: Users },
  { path: 'users/:id', component: User }
];

export const UsersRouting = RouterModule.forChild(usersRoutes);
```

Child Routes - refactored

```
// users.routing.ts
import { Routes, RouterModule } from '@angular/router';
import { Users } from './users.component';
import { User } from './user.component';

const usersRoutes: Routes = [
  { path: 'users',
    children: [
      { path: '', component: Users },
      { path: ':id', component: User }
    ]
  }
];

export const UsersRouting = RouterModule.forChild(usersRoutes);
```

Navigation

- Using regular links with hash
  - `#/home`
- Using **routerLink** directive
  - `['home']`
  - `['users', 34] /users/:id`
- Programatically
  - `router.navigate(['users', 34])`
  - `router.navigateByUrl('/users/34')`

routerLink

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

@Component({
  selector: 'users',
  template: `
    <h1>Users</h1>
    <tr *ngFor="let user of users">
      <td>
        <a [routerLink]="['/users', user.id]">{{user.username}}</a>
      </td>
    </tr>
  `
})
export class Users { }
```

Some examples

```
<!-- Top Routes -->
<a href="#/home">Home</a>
<a routerLink="home">Home</a>

<!-- Child routes -->
<a href="#/users/34">Spiderman</a>
<a [routerLink]="['users', 34]">Spiderman</a>
```

Passing data

```
<!-- params -->
<a href="#/users;flag=true/34">Spiderman</a>
<a [routerLink]="['users', { flag: true }, 34]">Spiderman</a>

<!-- hash fragments -->
<a href="#/users/34#section">Spiderman</a>
<a [routerLink]="['users', 34]" fragment="section">Spiderman</a>

<!-- params and queryParams -->
<a href="#/users/34;flag=true?q=1">Spiderman</a>
<a [routerLink]="['users', 34, {flag: true}]" [queryParams]="{q: 1}">Sp
```

Current Url

```
// { path: 'users/:id', component: User } Url: #/users/34
import { Router } from '@angular/router';

@Component({
  selector: 'user-details'
})
export class User implements OnInit {
  constructor(private router: Router){ }

  ngOnInit() {
    console.log(this.router.url); // /users/34
  }
}
```

Accessing Data

```
// { path: 'users/:id', component: User, data: { key: 1 } }
import { ActivatedRoute } from '@angular/router';

@Component({
  selector: 'user-details'
})
export class User implements OnInit {
  constructor(private route: ActivatedRoute){
    route.data.subscribe(data => {
      let key = data.key; // 1
    });
  }
  ngOnInit() {
    let key = this.route.snapshot.data.key; // 1
  }
}
```

Accessing Parameters

```
// { path: 'users/:id', component: User } Url: #/users/34;flag=true
import { ActivatedRoute } from '@angular/router';

@Component({
  selector: 'user-details'
})
export class User implements OnInit {
  constructor(private route: ActivatedRoute){
    route.params.subscribe(params => {
      let id = params.id; // 34
    });
  }
  ngOnInit() {
    let id = this.route.snapshot.params.id;  // 34
    let flag = this.route.snapshot.params.flag; // true
  }
}
```

Accessing hash fragment

```
// { path: 'users/:id', component: User } Url: #/users/34#section
import { Router } from '@angular/router';

@Component({
  selector: 'user-details'
})
export class User implements OnInit {
  constructor(private router: Router){
    router.routerState.root.fragment.subscribe(f => {
      let fragment = f; // section
    });
  }
  ngOnInit() {
    let fragment = this.router.routerState.snapshot.root.fragment; // s
  }
}
```

Accessing queryParams

```
// { path: 'users/:id', component: User } Url: #/users/34?q=1
import { Router } from '@angular/router';

@Component({
  selector: 'user-details'
})
export class User implements OnInit {
  constructor(private router: Router){
    router.routerState.root.queryParams.subscribe(params => {
      let q = params.q; // 1
    });
  }
  ngOnInit() {
    let q = this.router.routerState.snapshot.root.queryParams.q; // 1
  }
}
```

Navigation Guards

Ask before navigating

```
// users.routing.ts
import { Routes, RouterModule } from '@angular/router';
import { UserCanDeactivate } from './user.canDeactivate';

const usersRoutes: Routes = [
  { path: 'users', component: Users },
  { path: 'users/:id', component: User,
    canDeactivate: [UserCanDeactivate]
  }
];

export const UsersRouting = RouterModule.forChild(usersRoutes);
```

Ask before navigating

```
// user.canDeactivate.ts
import { Injectable } from '@angular/core';
import { CanDeactivate } from '@angular/router';

Injectable()
export class UserCanDeactivate implements CanDeactivate {
  canDeactivate() {
    return window.confirm('Do you want to continue?');
  }
}
```

Restrict Access

```
// users.routing.ts
import { Routes, RouterModule } from '@angular/router';
import { AuthCanActivate } from './auth.canActivate';

const usersRoutes: Routes = [
  { path: 'users', component: Users },
  { path: 'users/:id', component: User,
    canDeactivate: [UserCanDeactivate],
    canActivate: [AuthCanActivate]
  }
];

export const usersRouting = RouterModule.forChild(usersRoutes);
```

Restrict Access

```
// auth.canActivate.ts
import { Injectable } from '@angular/core';
import { Router, CanActivate } from '@angular/router';
import { LoginService } from './login.service';

export class AuthCanActivate implements CanActivate {
  constructor(private loginService: LoginService, private router: Route

  canActivate() {
    if (!this.loginService.authorised) {
      this.router.navigate(['/home']);
      return false;
    }
    return true;
  }
}
```

Lazy Loading

Lazy Loading

```
// app.routing.ts
const aboutRoutes: Routes = [
  { path: 'about', loadChildren: './app/about.module.ts' },
];
export const AboutRouting = RouterModule.forChild(aboutRoutes);

//about.module.ts
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { AboutRouting } from './about.routing';
import { About } from './about.component';

@NgModule({
  imports: [ CommonModule, AboutRouting ],
  declarations: [ About ]
})
export default class AboutModule { }
```

Preloading

default preloading

```
// app.routing.ts
import { Routes, RouterModule, PreloadAllModules } from '@angular/route

const appRoutes: Routes = [
  { path: 'about', loadChildren: './app/about.module.ts' },
];

export const AppRouting = RouterModule.forRoot(appRoutes, {
  useHash: true,
  preloadingStrategy: PreloadAllModules
});
```

Wrap-up

Requirements

- Network resilience
- Ask before navigating
- Restrict access
- Lazy loading

Thanks!