ANGULAR ARCHITECTS
INSIDE KNOWLEDGE

SOFTWARE ARCHITECT

Adv. Routing

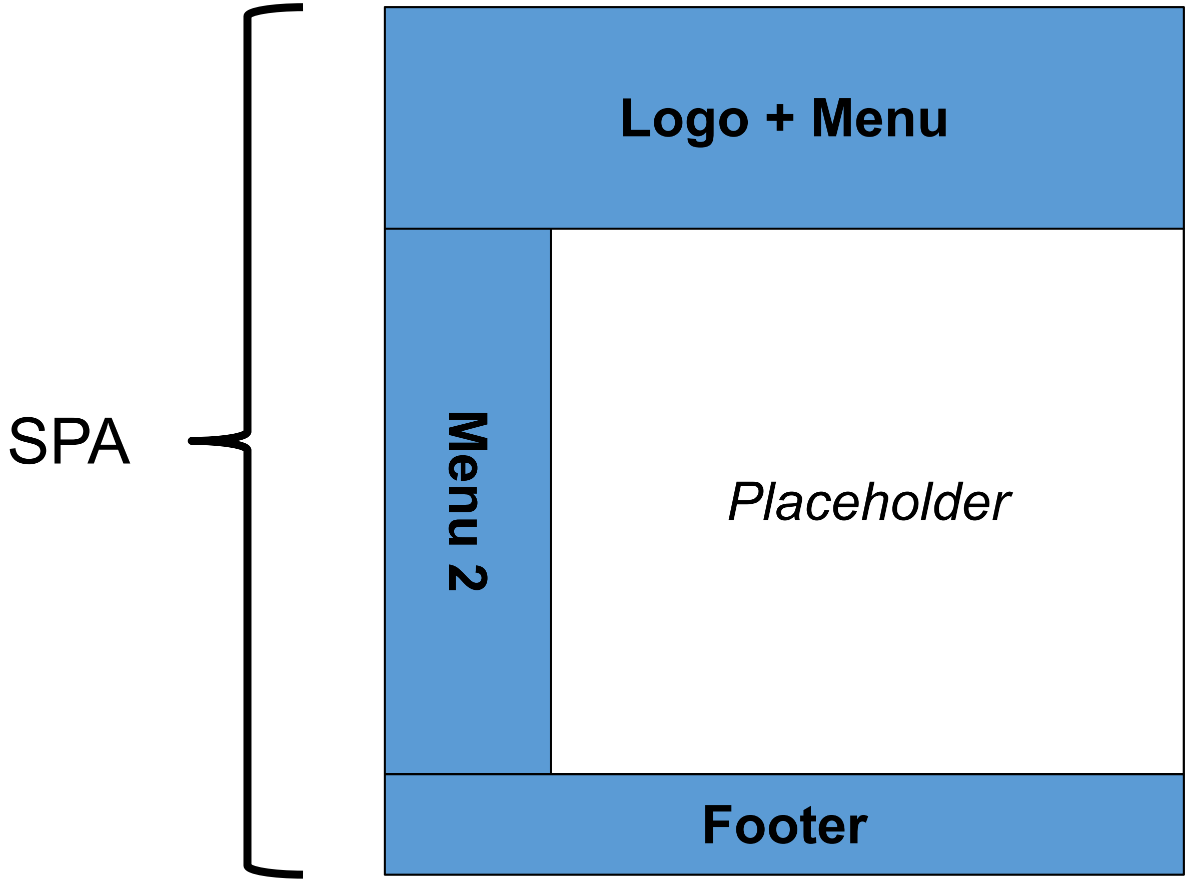
Alex Thalhammer

Routes are **the key points** of your Angular application!

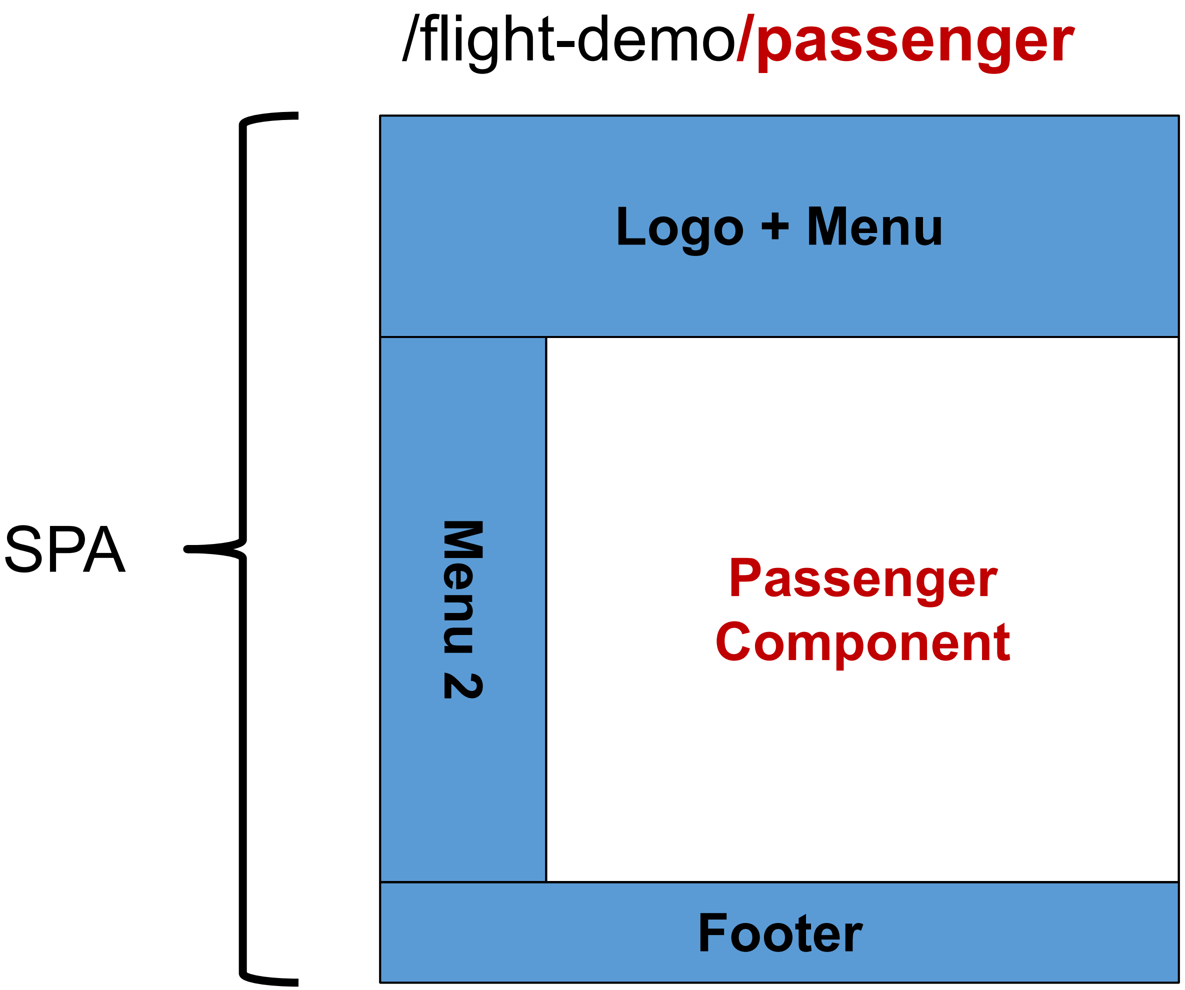
Outline

- Basics
- Hierachrichal Routing
- Aux Routes
- Guards
- Resolver
- Router Events
- Lazy Loading & Preloading

Routing in Angular



Routing in Angular



Config

```
const APP_ROUTES: Routes = [
    {
        path: 'home',
        component: HomeComponent
    },
    {
        path: 'flight-search',
        component: FlightSearchComponent
    }
]
```

Config

```
const APP_ROUTES: Routes = [
    {
        path: '',
        redirectTo: 'home',
        pathMatch: 'full'
    },
    {
        path: 'home',
        component: HomeComponent
    },
    {
        path: 'flight-search',
        component: FlightSearchComponent
    }
]
```

Config

```
const APP_ROUTES: Routes = [
    {
        path: '',
        redirectTo: 'home',
        pathMatch: 'full'
    },
    {
        path: 'home',
        component: HomeComponent
    },
    […]
    {
        path: '**',
        redirectTo: 'home'
    }
]
```

Config

```
// app.module.ts
@NgModule({
    imports: [
        BrowserModule,
        HttpModule,
        FormsModule,
        RouterModule.forRoot(APP_ROUTES);
    ],
    […]
})
export class AppModule {
}
```

for root module

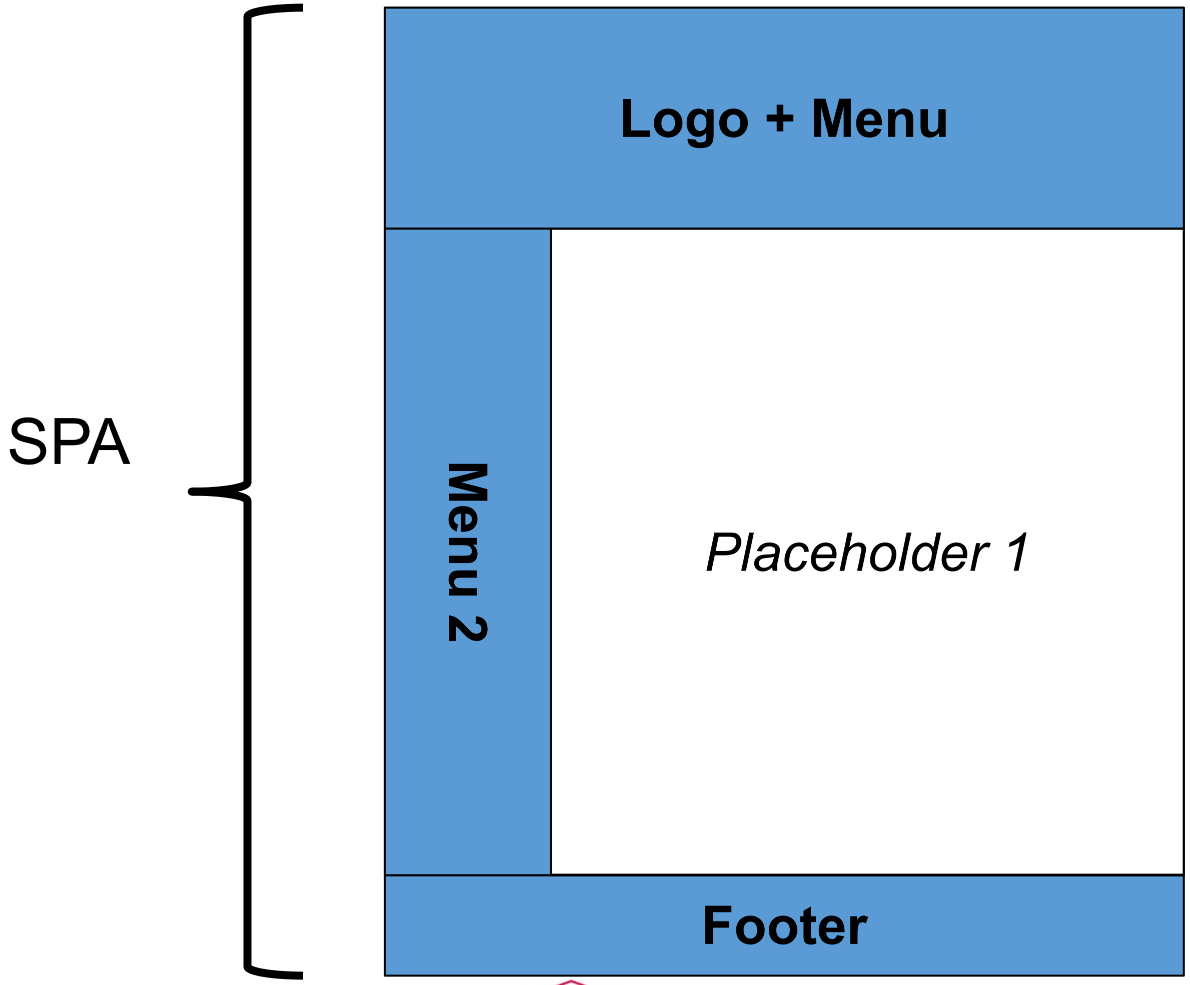
for feature module: forChild

View von AppComponent

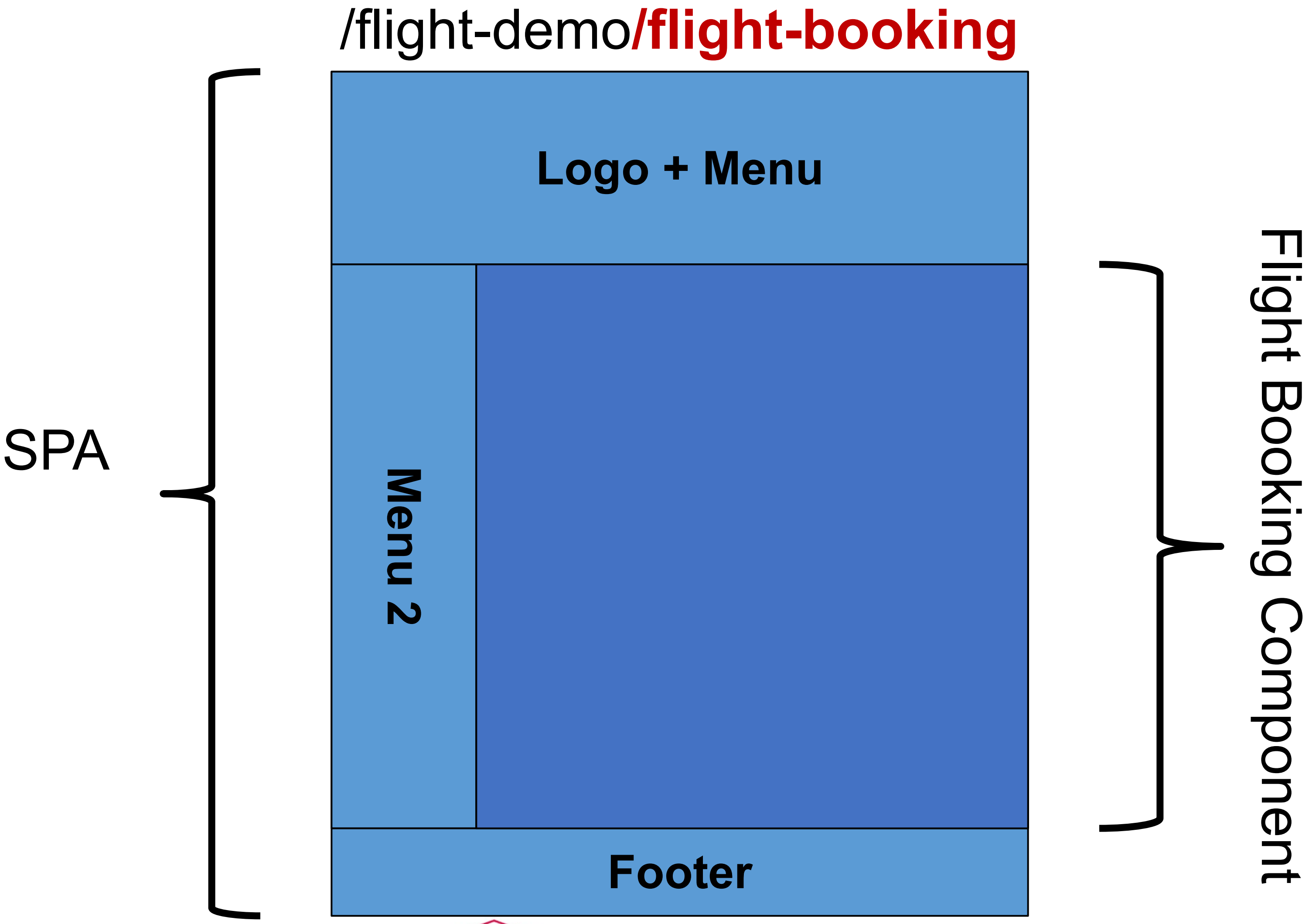
```
<a routerLink="/home">Home</a>
<a routerLink="/flight-search">Search flight</a>

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

Hierarchical Routing

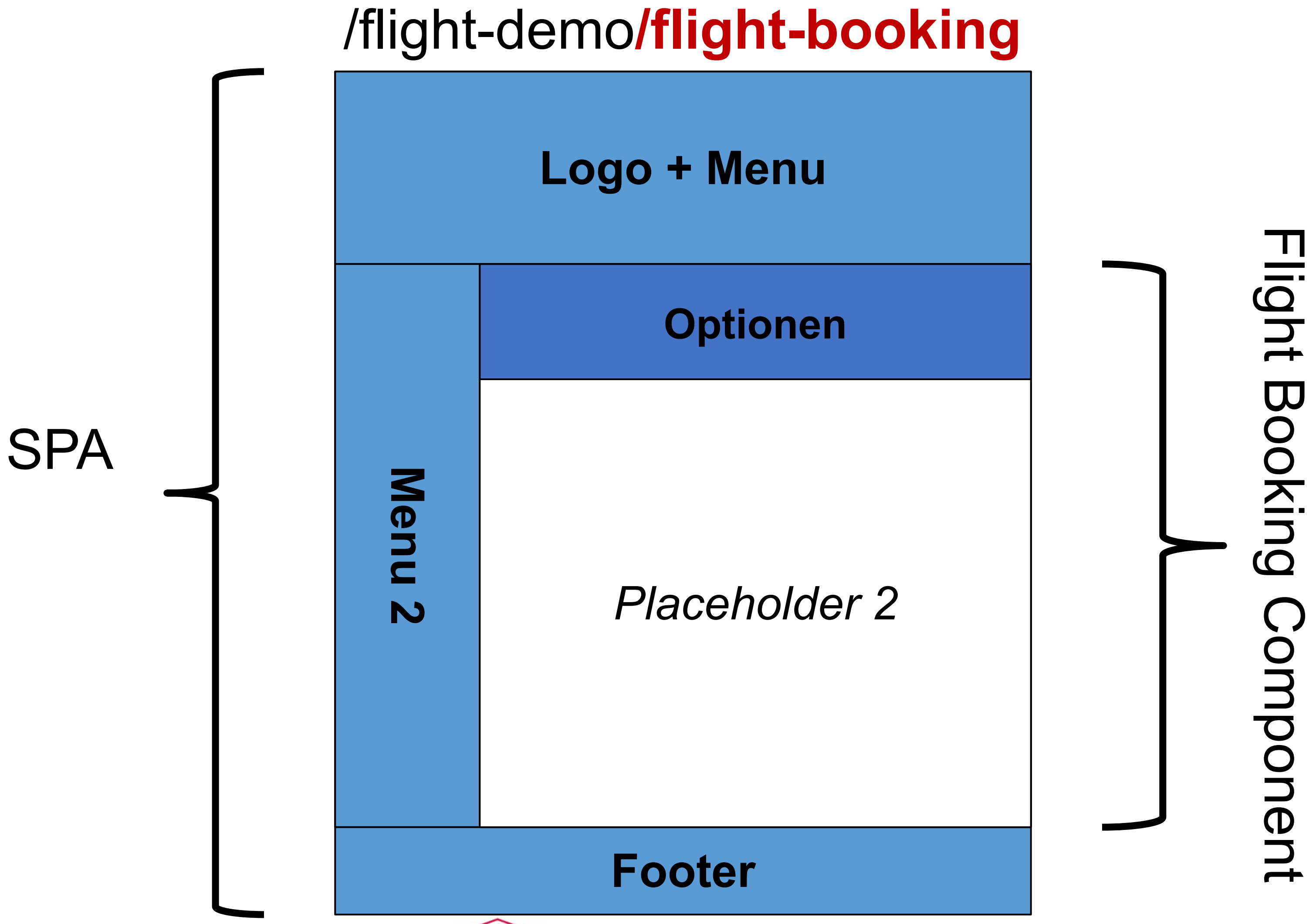
Hierarchical Views



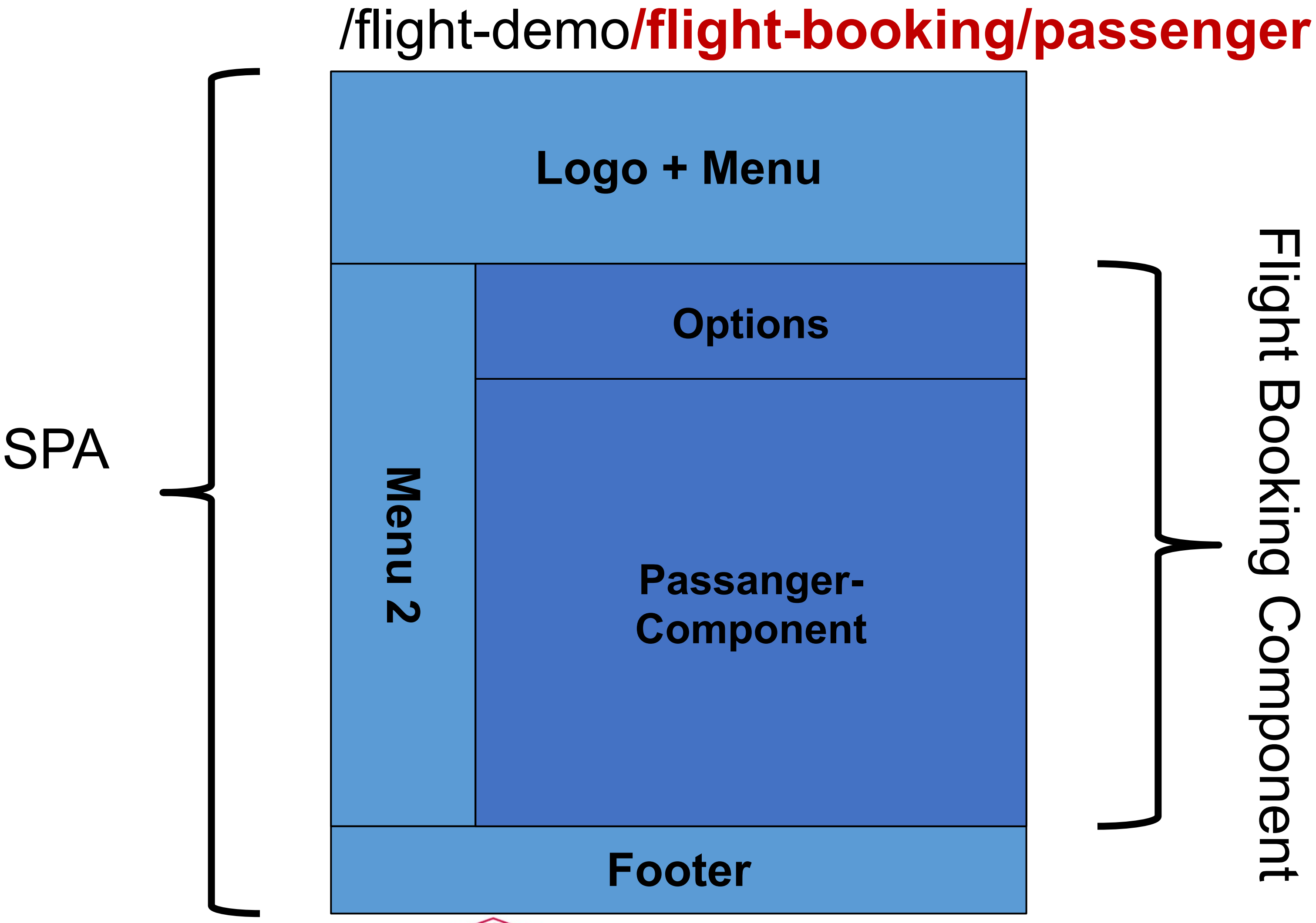
Hierarchical Views



Hierarchical Views

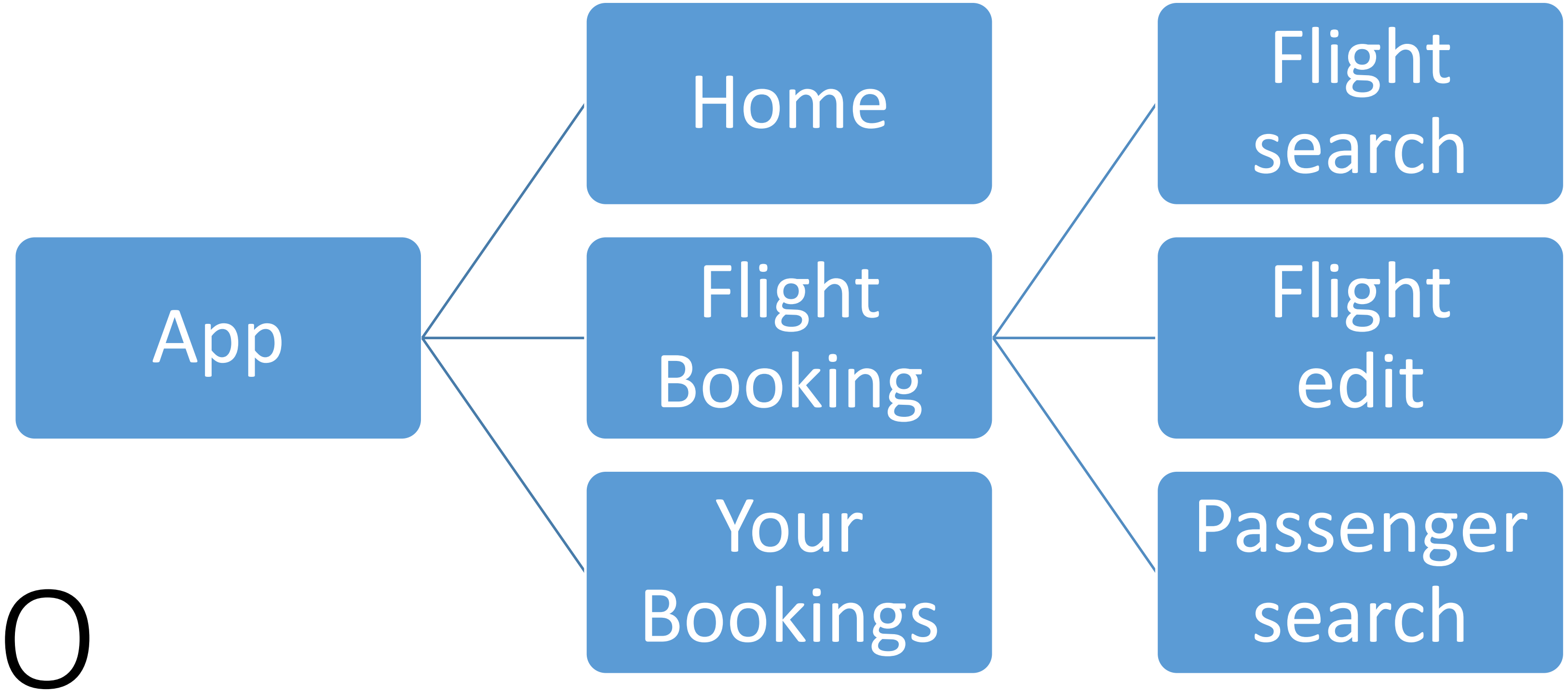


Hierarchische Views

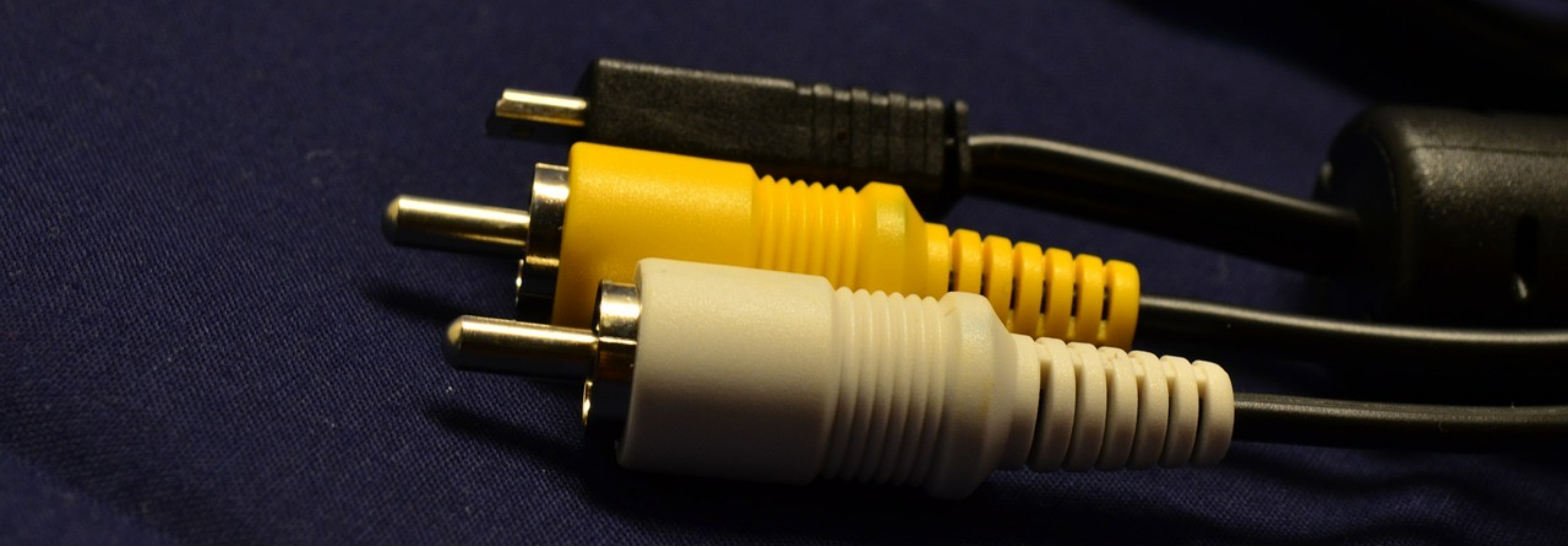


Config

```
const APP_ROUTES: Routes = [
    {
        path: 'flug-buchen',
        component: FlugBuchenComponent,
        children: [
            {
                path: 'flug-suchen',
                component: FlugSuchenComponent
            },
            […]
        ]
    }
];
```

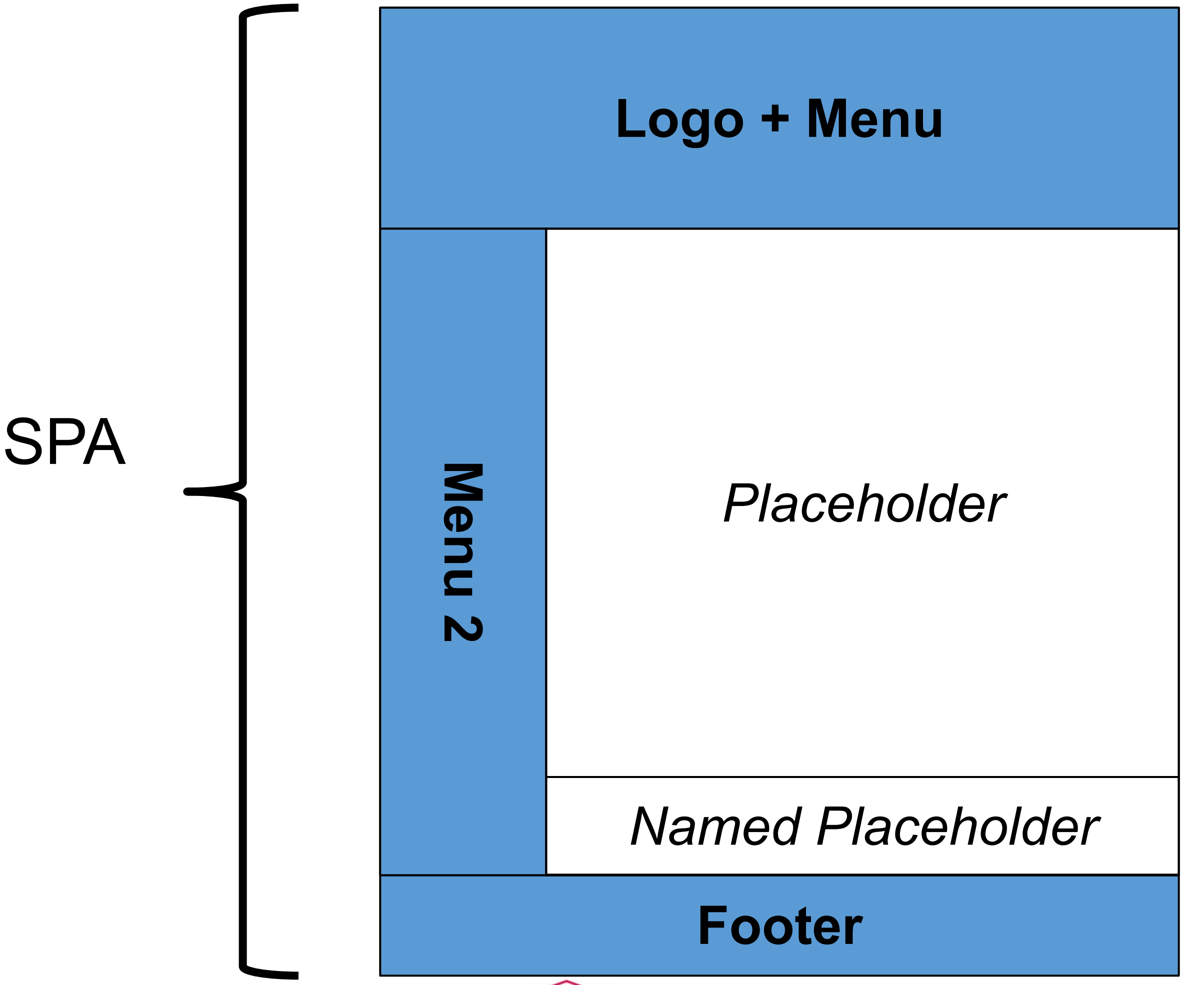

App
Home
Flight Booking
Your Bookings
Flight search
Flight edit
Passenger search

DEMO

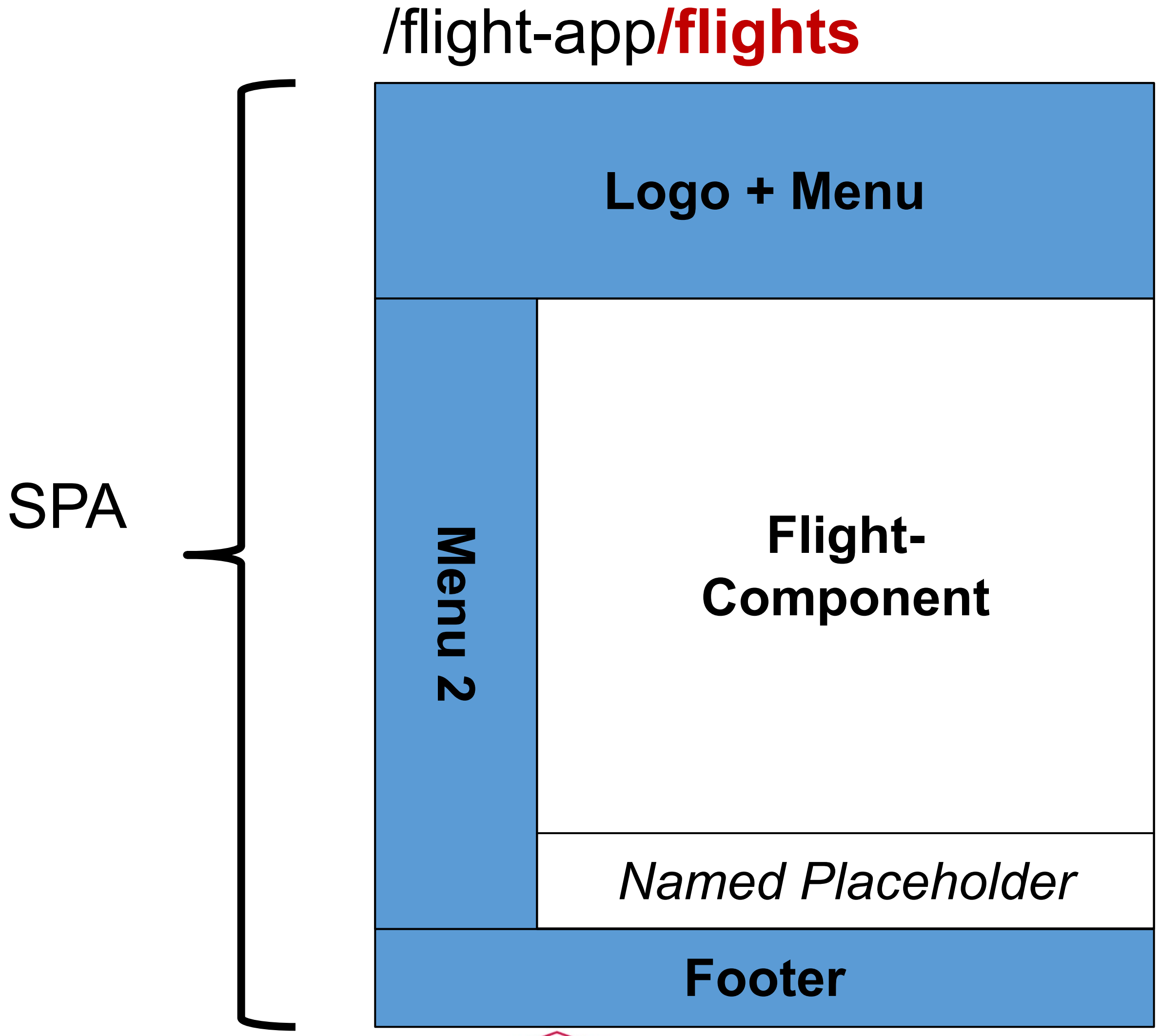


Aux-Routes

Aux-Routes

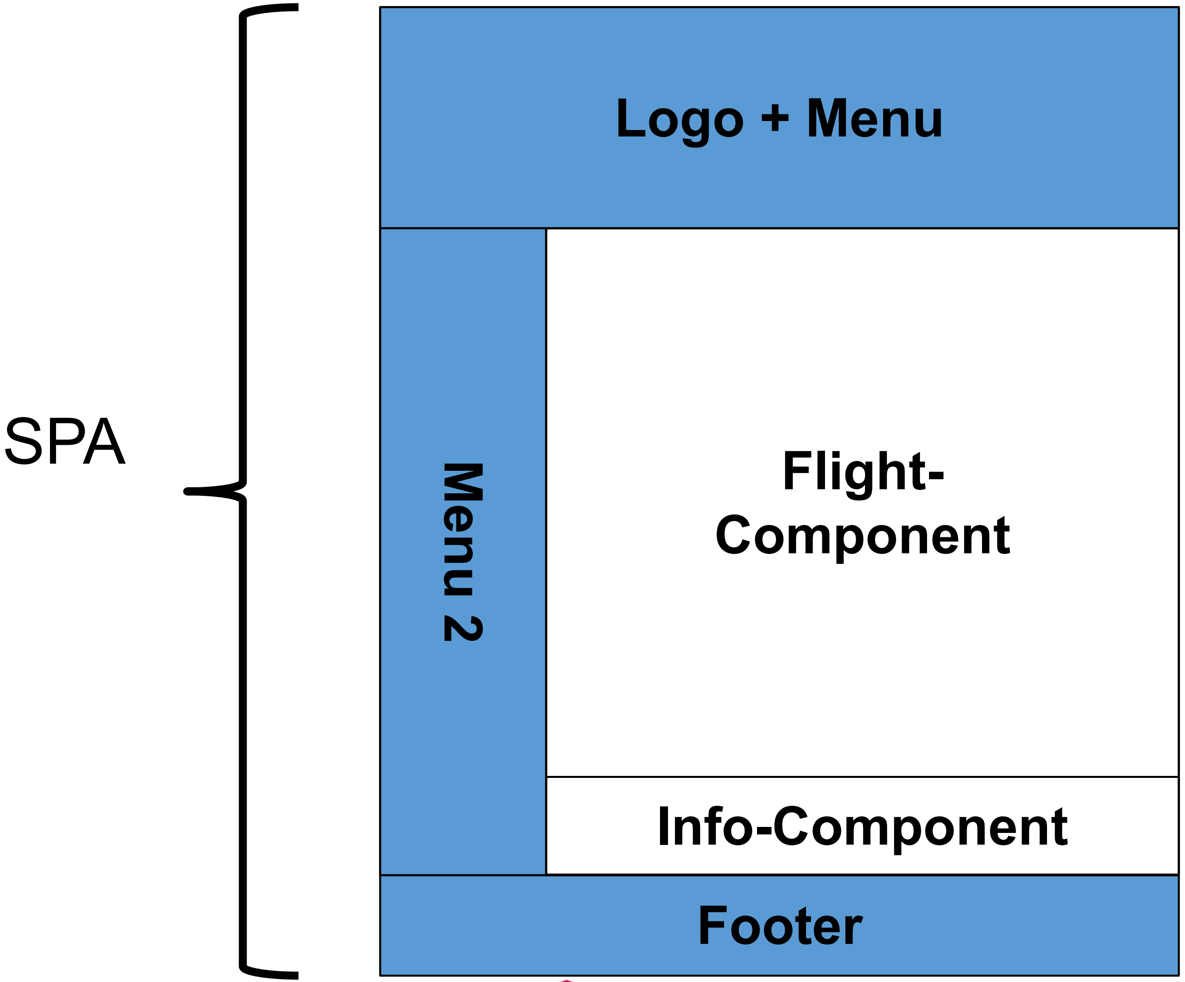


Aux-Routes


/flight-app/flights
SPA
Logo + Menu
Menu 2
Flight-Component
Named Placeholder
Footer

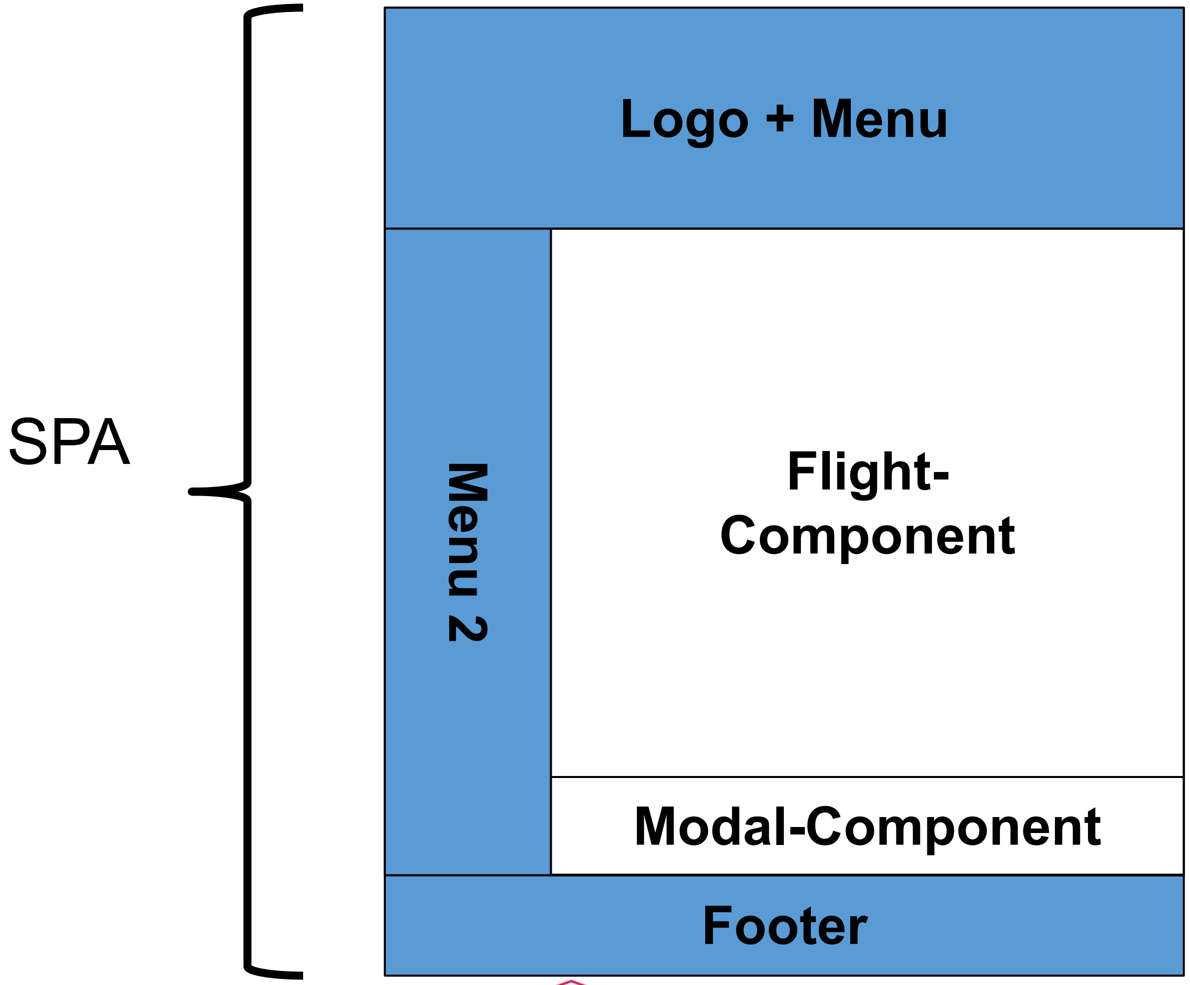
Aux-Routes

/flight-app/**flights(aux:info)**



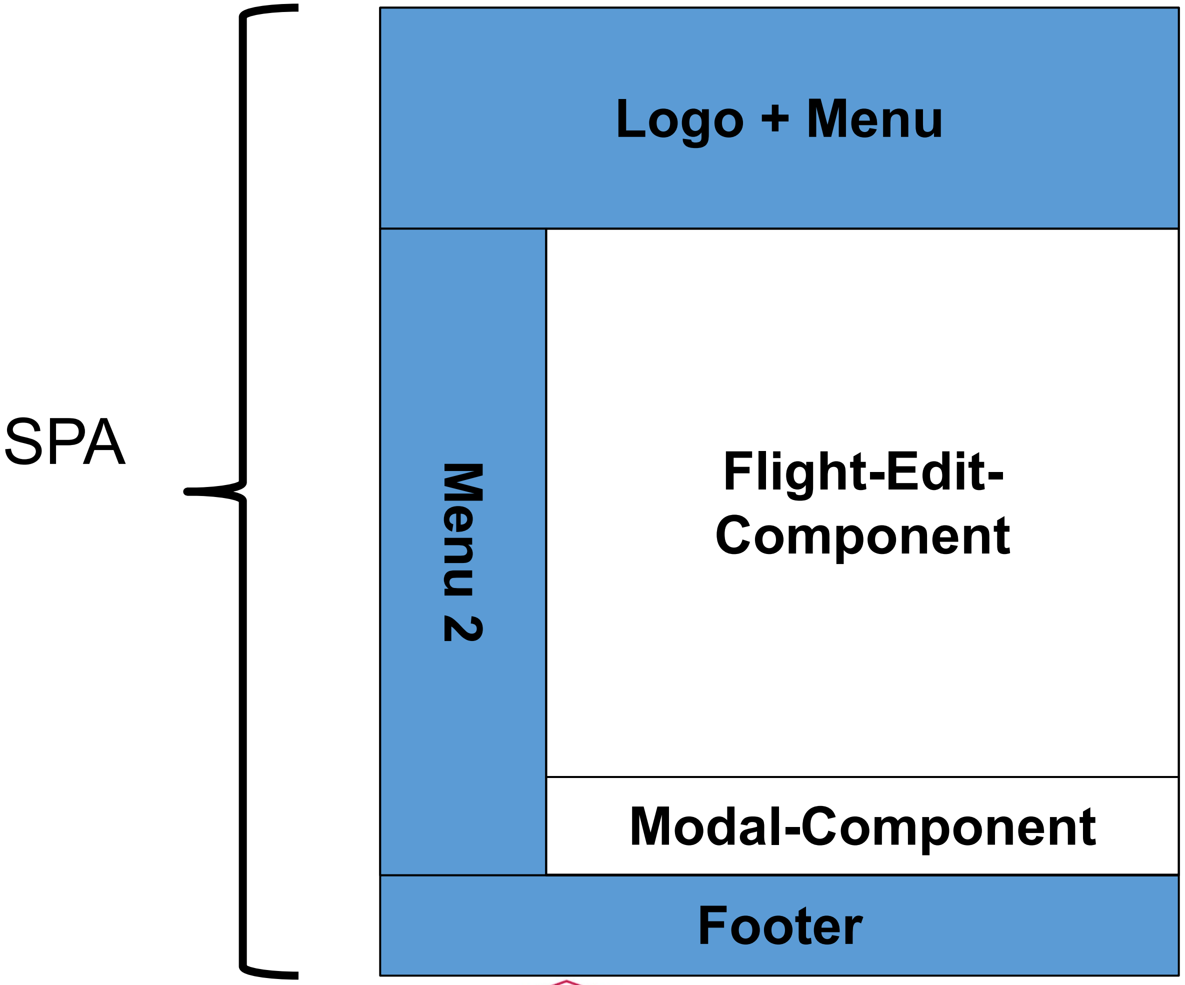
Aux-Routes

/flight-app**/flights(aux:info/modal)**



Aux-Routes

/flight-app**/flights(aux:info/modal)/17**



Use Cases

- Modal Dialogs
- “Applets” & autonomous areas
- Commander-Style

Define placeholder

Standard-Name: primary

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

<hr>

<router-outlet name="aux"></router-outlet>
```

Config

```
export const ROUTE_CONFIG: Routes = [
    {
        path: 'home',
        component: HomeComponent
    },
    {
        path: 'info',
        component: InfoComponent,
        outlet: 'aux'
    },
    {
        path: 'dashboard',
        component: DashboardComponent,
        outlet: 'aux'
    }
]
```

Route Aux Routes

```
<a [routerLink]="[{outlets: { aux: 'info' }}]">
    Activate Info
</a>

<a [routerLink]="[{outlets: { aux: null }}]">
    Deactivate Info
</a>
```

Route multiple Outlets at once

```
<a [routerLink]="[{outlets: {
                     aux: 'basket',
                     primary: 'flight-booking/flight-search' }}]"> … </a>


<a [routerLink]="[{outlets: { aux: 'basket',
                     primary: ['flight-booking', 'flight-search'] }}]"> … </a>


<a [routerLink]="[{outlets: { aux: 'basket',
                     primary: ['flight-booking', 'flight-edit', 17] }}]"> … </a>
```

Route programatically

```
export class AppComponent {

    constructor(private router: Router) {
    }

    activateInfo() {
        this.router.navigate([{outlets: { aux: 'info' }}]);
    }

    deactivateInfo() {
        this.router.navigate([{outlets: { aux: null }}]);
    }
}
```

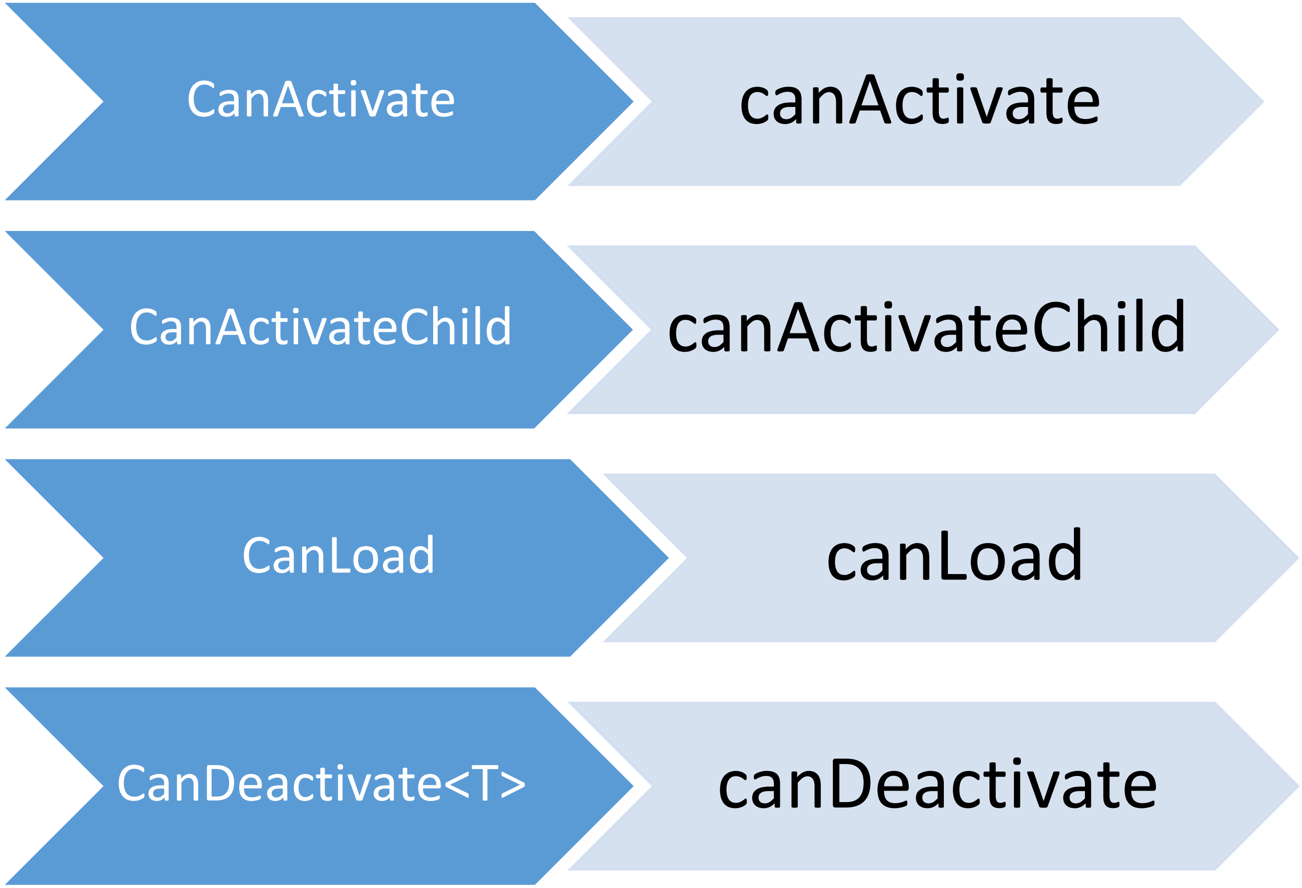
Guards

What are guards?

- Services

- Can avoid activate or deactivate of a route

Guards



CanActivate

canActivate

CanActivateChild

canActivateChild

CanLoad

canLoad

CanDeactivate<T>

canDeactivate

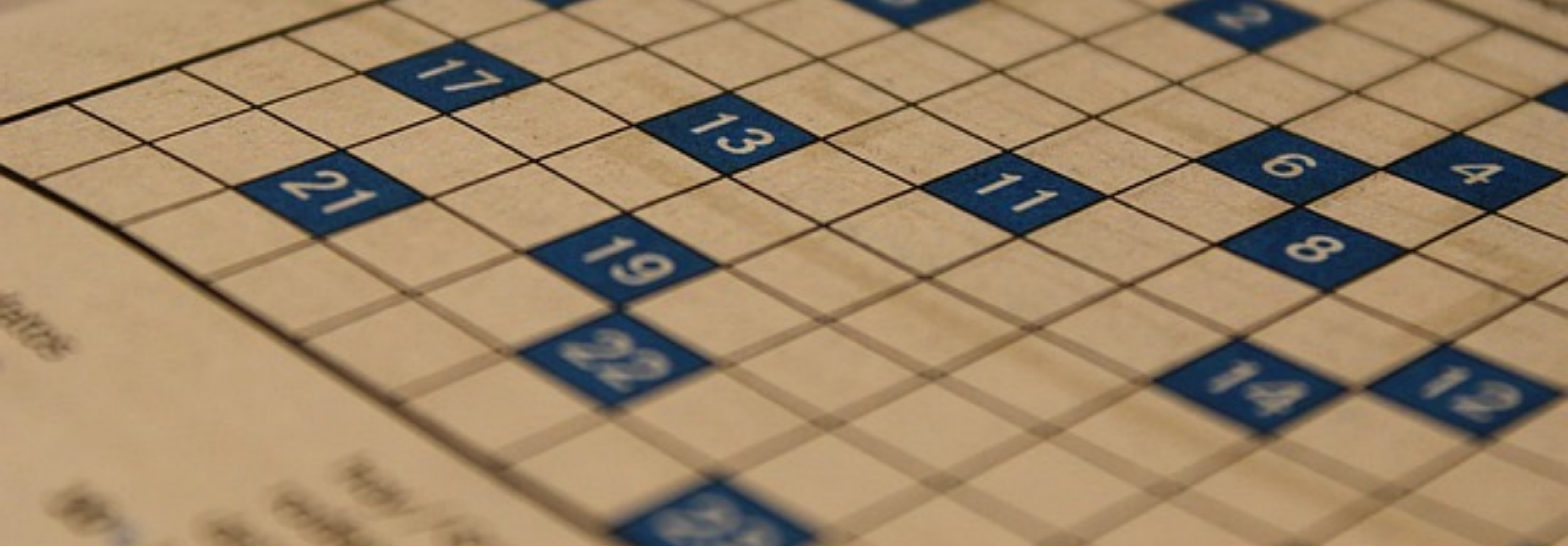
Return type: boolean | Observable<boolean> | Promise<boolean>

Guards in the Config

```
const APP_ROUTES: Routes = [
    {
        path: '/flug-buchen',
        component: FlugBuchenComponent,
        canActivate: [AuthGuard],
        children: [
            {
                path: 'flug-edit/:id',
                component: FlugEditComponent,
                canDeactivate: [FlugEditGuard]
            },
            […]
        ]
]
```

Provider für Guards

```
// app.module.ts
@NgModule({
    providers: [
      FlugEditGuard,
      AuthGuard
    ],
    […]
})
export class AppModule {
}
```



Resolver

What is a resolver?

- Service
- Get active when changing route
- Delays routing
- Useful for loading data (or sth. like micro frontends)
- In the meantime shows a loading indicator

Resolver

```
@Injectable()
export class FlightResolver implements Resolve<Flight>
{
    constructor(private flightService: FlightService) {
    }

    resolve(route, state): Observable<Flight> | Promise<Flight> | Flight {

        return […]
    }
}
```

Register resolver

```
const FLIGHT_BOOKING_ROUTES: Routes = [
    […]
    {
        path: 'flight-edit/:id',
        component: FlightEditComponent,
        resolve: {
            flight: FlightResolver
        }
    }

];
```

Receive data

```
@Component({ … })
export class FlightEditComponent {

    flight: Flight;

    constructor(private route: ActivatedRoute) { }

    ngOnInit() {
        this.route.data.subscribe(
            data => {
                this.flight = data['flight'];
            }
        );
    }
}
```

Lazy Loading

Why Lazy Loading?

- Load module once they are needed

- Improve initial load (size & time)

Root Module with Lazy Loading

```
const APP_ROUTE_CONFIG: Routes = [
    {
        path: '',
        redirectTo: 'home',
        pathMatch: 'full'
    },
    {
        path: 'home',
        component: HomeComponent
    },
    {
        path: 'flights',
        loadChildren:
            '[…]flight-booking.module#FlightBookingModule'
    }
];
```

Root Module with Lazy Loading

```
const APP_ROUTE_CONFIG: Routes = [
    {
        path: '',
        redirectTo: 'home',
        pathMatch: 'full'
    },
    {
        path: 'home',
        component: HomeComponent
    },
    {
        path: 'flights',
        loadChildren: () => import('[…]flight-booking.module')
                                 .then(m => m.FlightBookingModule);
    }
];
```

Routes for Feature Module

```
const FLUG_ROUTES =    [
    {
        path: '',
        component: FlugBuchenComponent,
        […]
    },
    […]
}
```

Routes for Feature Module

```
const FLUG_ROUTES =    [
    {
        path: 'subroute',
        component: FlugBuchenComponent,
        […]
    },
    […]
}
```

Preloading

Idea

- Lazy Loaded modules are loaded after initial app load
- On use module is already ready

Activate Preloading

```
export const AppRoutesModule =
                 RouterModule.forRoot(
                   ROUTE_CONFIG,
                   { preloadingStrategy: PreloadAllModules });
```

Conclusion

- Child Routes
- Aux Routes
- Guards
- Resolver
- Lazy Loading & Preloading