# Welcome to the Apprenticeship Program – Session 28

# Agenda

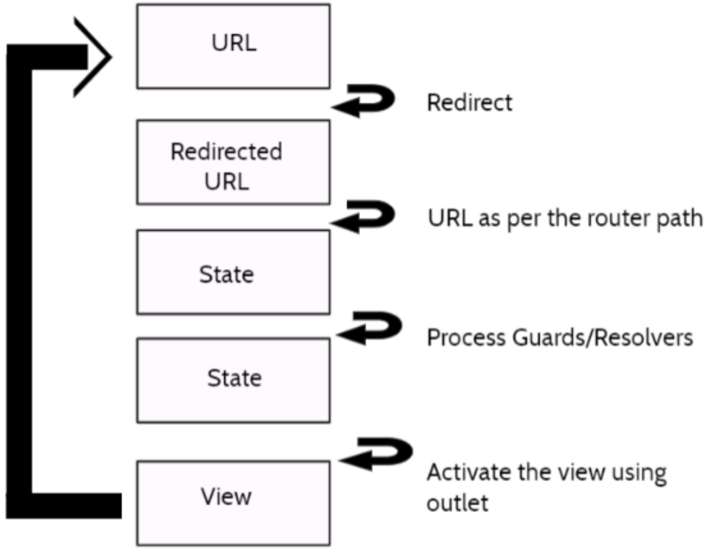
- What is Routing in Angular?
- Angular Router Basics
- Setting Up Routes
- Navigation using RouterLink
- Route Parameters
- Route Guards (Intro)

# What is Routing in Angular?

- Routing allows navigation between different views/components in a Single Page Application (SPA).
- Uses Angular Router Module to map URLs to components.
- Updates the view without reloading the page.

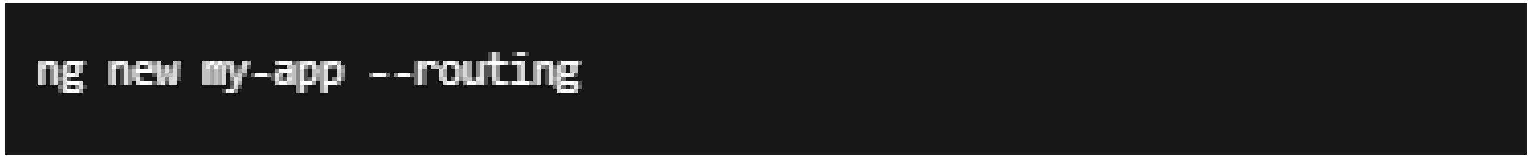
# How Routing Works in SPAs

- **A route is a URL path + a component.**
- **The router displays the component based on the URL.**
- **The page doesn’t reload — Angular replaces the content dynamically.**



# Enabling Routing in Angular

- **During Angular CLI project creation:**

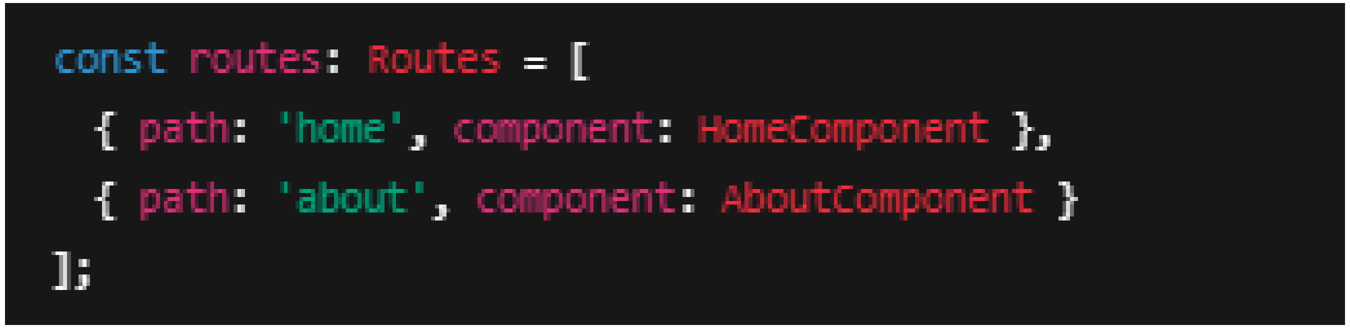


```
ng new my-app --routing
```

  - Adds app-routing.module.ts automatically to handle routes

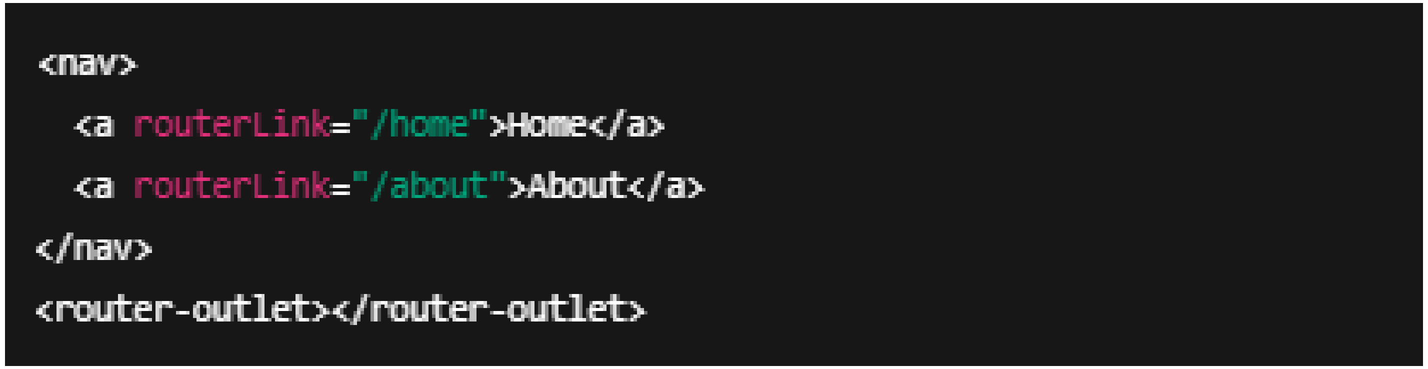
# Defining Routes

- Add this in app-routing.module.ts.
- Use RouterModule.forRoot(routes) to register



```
const routes: Routes = [
  { path: 'home', component: HomeComponent },
  { path: 'about', component: AboutComponent }
];
```

# Navigating with RouterLink

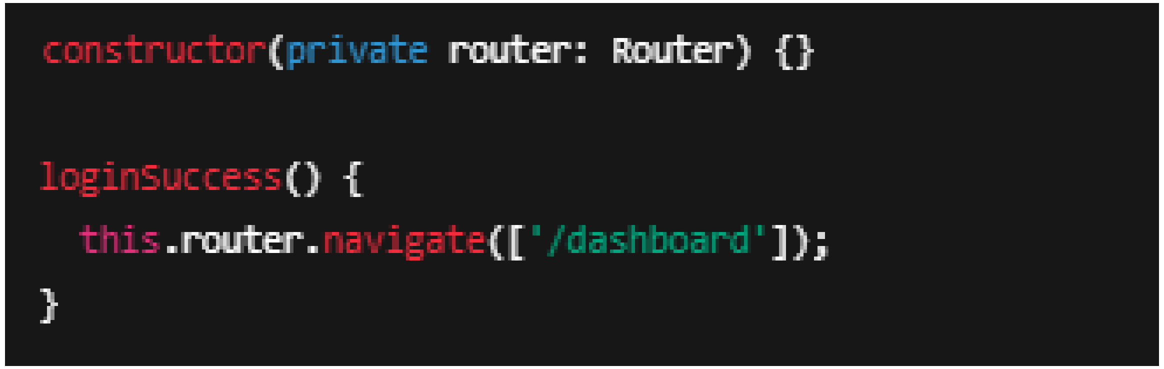


```
<nav>
  <a routerLink="/home">Home</a>
  <a routerLink="/about">About</a>
</nav>
<router-outlet></router-outlet>
```

- routerLink: Navigates without reloading.
- router-outlet: Placeholder for rendered view.

DEMO

# Using Programmatic Navigation



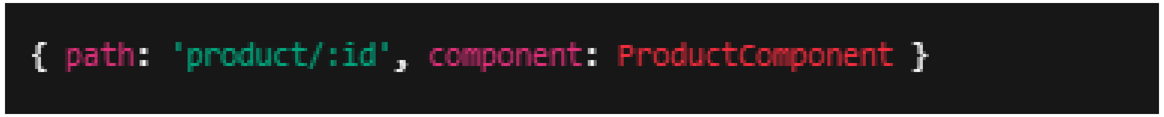
```
constructor(private router: Router) {}

loginSuccess() {
  this.router.navigate(['/dashboard']);
}
```

- Navigate from code (e.g., after login).
- Useful for navigation after user actions or validations.

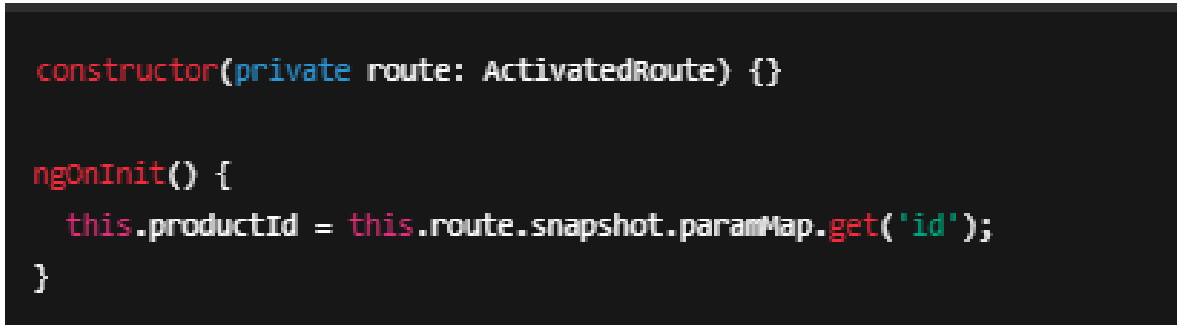
# Route Parameters

- Use dynamic paths:



```
{ path: 'product/:id', component: ProductComponent }
```

- Access the parameter:



```
constructor(private route: ActivatedRoute) {}

ngOnInit() {
  this.productId = this.route.snapshot.paramMap.get('id');
}
```

- Example URL: /product/42 -> id = 42

# Intro to Route Guards

- What are Route Guards?
  - Route guards are used to control navigation.
  - They prevent unauthorized access or warn before navigating away.
  - They restrict access based on conditions (e.g., authentication)

| Guard | Use |
|---|---|
| CanActivate | Controls if a route can be entered |
| CanDeactivate | Confirms before leaving a route |
| CanActivateChild | Controls access to child routes |
| CanLoad | Prevents lazy-loaded modules from loading if blocked |

# Conclusion and Q&A

- **Key Takeaways:**
  - Routing connects URLs to components.
  - RouterModule defines and registers routes.
  - routerLink handles navigation in templates.
  - router-outlet displays matched component.
  - Use route parameters for dynamic content.
  - Guards help manage protected routes.

**Questions?** 👀