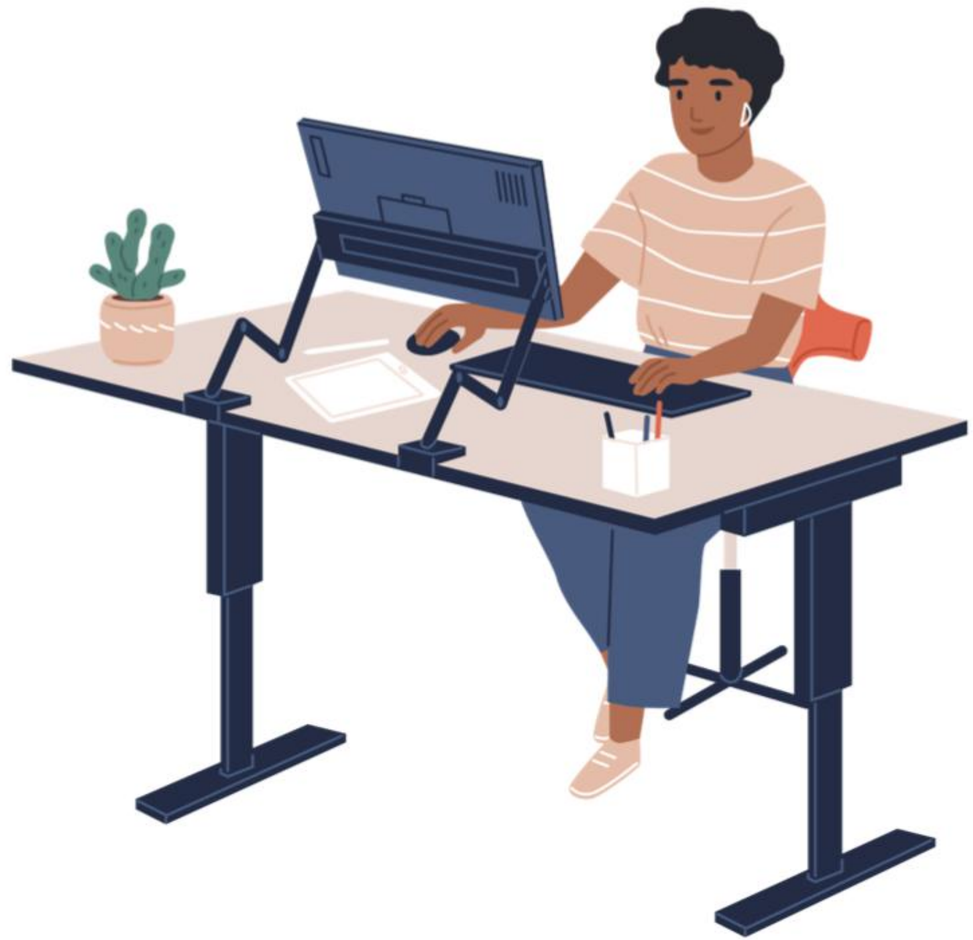
Learning Consolidation

# Implement Navigation Using Angular Routing



# In this Sprint you learned to…

- Explain navigation in a single-page application
- Enable routing with basic routes in an Angular Application
  - Define routes in an Angular application using the `Routes` array.
  - Add routes to the Angular application
- Access route information using the `ActivatedRoute` interface
- Set up wildcard routes and redirects in route configuration
- Explain the significance of Route order
- Consume the `Router` service to enable programmatic navigation between views.

Slide Note

# Web Navigation

## Multi-Page Application

- Navigation takes place between the web pages.
- Every navigation causes a complete web page load.

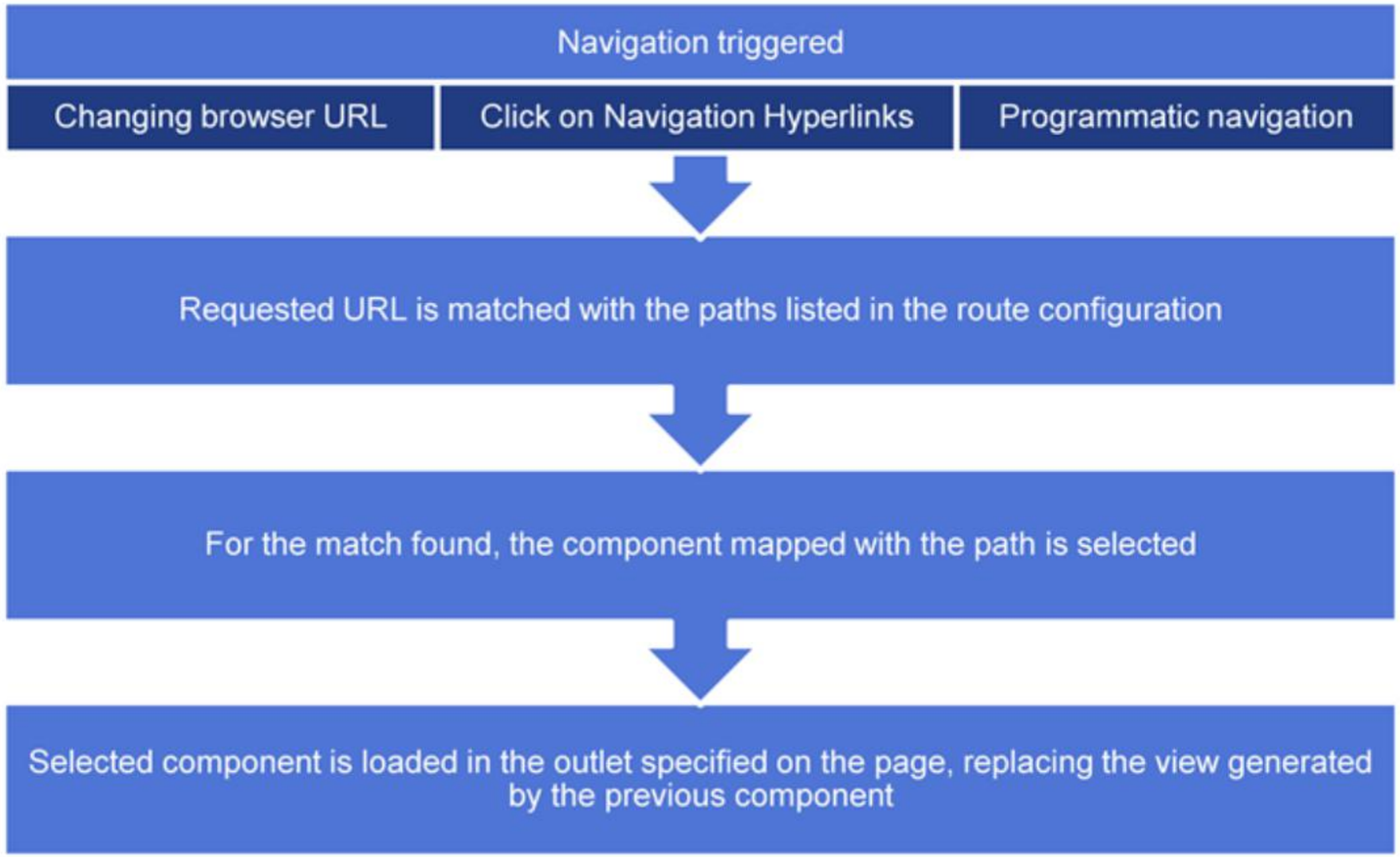
## Single-Page Application

- Navigation takes place between views.
- Every navigation changes the view with a new set of data without reloading the entire page.

Slide Note

During navigation, the previously loaded component is destroyed and the target component is dynamically loaded.

# How Does Navigation Happen in an Angular Application?



Slide Note

# <base href>

- This tag is automatically added to the **`index.html`** file, if the app is created using the Angular CLI command.
- If the app is manually created, ensure the tag is added to the `<head>` tag in the **`index.html`** file.

  `<base href = “/”>`

- This tag specifies a base path for resolving relative URLs to assets such as images, scripts, and style sheets.
- For the tag `<base href="/note/app/">`, the browser resolves a URL such as `images/profile.jpg` into a server request for `note/app/images/profile.jpg`.
  - During navigation, the Angular router uses the `base href` as the base path to component, template, and module files.

Slide Note

# Common Routing Tasks

Slide Note

# Using Angular Router

**Navigation in the Angular application is implemented with the help of following key components of the `@angular/router` package.**

Routing Module

- It is an `NgModule` that provides service and directives to the application.
- The `forRoot()` method of `RouterModule` is called to create `NgModule` with directives, route configurations, and `Router` service.

`Router Service`

- This service helps to manage navigations among views and URL manipulations.

`RouterLink` Directive

- This is an attribute directive that transforms the host element into the link element.
- The value of this directive is the path to the component that needs to be navigated.

`RouterOutlet` Directive

- Specifies location on the page for loading the navigated component dynamically.

Slide Note

# Retrieving Route Information

- Information for application components can be passed using the route.
- `The ActivatedRoute` interface is used to retrieve this information.
- The route path and the parameters are available through an injected router service called the `ActivatedRoute`.
- The route parameter and query parameter data can be retrieved by subscribing to the observable properties: `paramMap` of `ActivatedRoute`.

Slide Note

# Setting Up Wild Card Routes

- An invalid route URL leads to error 404 (Not Found).
- Setting up wild card routes helps in handling this error.
- Angular recognizes the route in the router array as the wild card route if the value of the path is **.
- This route should be the last route in the router array.
- The router selects this route if none of the previous routes match the specified route URL.

Slide Note

# Setting Up Redirects

- When the application launches, the component rendering the landing view or the home view should by default be loaded.
- The route configuration would have a path to this home view component.
- Additionally, in the route configuration, a route configuration should be added that defaults to the home view.
- This is achieved with the help of the `redirectTo` property and `pathMatch` property.
- This redirect should precede the wild-card route.

Slide Note

# `pathMatch` Property

- `The pathMatch` property value determines the strategy used for matching the path with the URL elements.
- This property takes the following values:
  - `prefix`
    - This is the default value.
    - Angular will search for a prefix of the path (in the URL) in the routes array.
  - `full`
    - Angular will search for the exact path (in the URL) in the routes array.
    - This value is used when setting up route redirects with empty route paths.
      - Otherwise, the router would apply the redirect when navigating to any destination and hence would create an endless loop.

Slide Note

# Programmatic Navigation

- Programmatic navigation in Angular is executed using the Angular `Router` service.
- The Angular `Router` service class provides the navigation methods that accept the navigation paths and perform navigation.
- The Angular `Router` service provides two methods for navigation:
  - `navigateByUrl()`
    - ❖ This method accepts an absolute path and requests navigation to a view.

```
router.navigateByUrl('categories/10/product/1001');
```

  - `navigate()`
    - ❖ This method accepts an array of commands and requests navigation.

```
router.navigate(['categories', 10, 'product', 1001]);
```

Slide Note

The LoginComponent will never load since the path login will match wild card characters, so the app will navigate to NotFoundComponent

# Significance of Route Order

- The router uses the first-match-wins strategy when matching routes.
- More specific routes should be placed above the less specific ones.
- List routes with a static path first, followed by an empty path route, that matches the default route.
- The wild card route comes last because it matches every UR.

**Incorrect Route Order (logical error)**

```
const routes: Routes = [
 {
path: "home",
component: NavbarComponent,
},
{
 path: "**",
 component: NotFoundComponent
},

{
 path: "login",
 component: LoginComponent
}
```

**Correct Route Order**

```
const routes: Routes = [
 {
path: "home",
component: NavbarComponent,
},
{
 path: "login",
 component: LoginComponent
},

{
 path: "**",
 component: NotFoundComponent
}
```

Slide Note

# Self-Check

The _____directive substitutes the normal href property and makes it easier to work with route links in Angular.

1. routeLink
2. router-Link
3. routerOutlet
4. routerLink

Slide Note

# Self-Check: Solution

The _____directive substitutes the normal href property and makes it easier to work with route links in Angular.

1. routeLink
2. router-Link
3. routerOutlet
4. **routerLink**

Slide Note

# Self-Check

The route _____ allow/allows us to pass values in our URL to our component so that we can dynamically change our view content.

1. modules
2. template-reference
3. parameters
4. array

Slide Note

# Self-Check: Solution

The route _____ allow/allows us to pass values in our URL to our component so that we can dynamically change our view content.

1. modules
2. template-reference
3. **parameters**
4. array

Slide Note

# Self-Check

The _____ static method is the method that configures the root routing module for your app.

1. forRoot()
2. forChild()
3. forRoots()
4. forRoutes()

Slide Note

# Self-Check: Solution

The _____ static method is the method that configures the root routing module for your app.

1. **forRoot()**
2. forChild()
3. forRoots()
4. forRoutes()