Driving State with Router Params

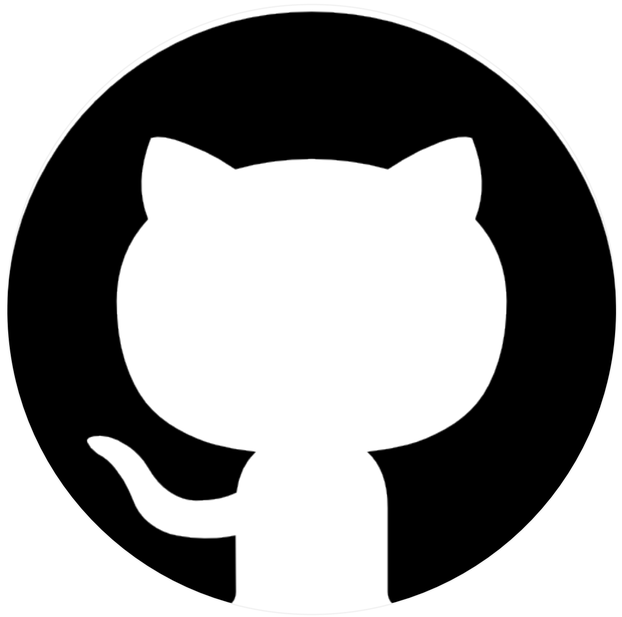
Lara Newsom

Software Engineer | Speaker | Angular GDE

@laranerdsom

What We Will Cover in This Module

- Three types of Angular router parameters
- Adding parameters to a route
- Consuming parameters from a route
- Defining router state flow



Repository for Module Six

github.com/lara-newsom/angular-routing-course

- module-six-start

Three Types of Angular Router Parameters

Required Path Parameters

Query Parameters

Optional Matrix Parameters

Route parameters are meant for small amounts of data like numbers, strings, or booleans

Required Path Parameters

- Dynamic path segments
- Declared in the path property of the route
- Prefixed with a colon
- Path parameters are named
- Only available in components activated by the route

Even though path parameters are visible in the URL, they are not accessible from everywhere

The Bus Analogy

Routes are like bus lines

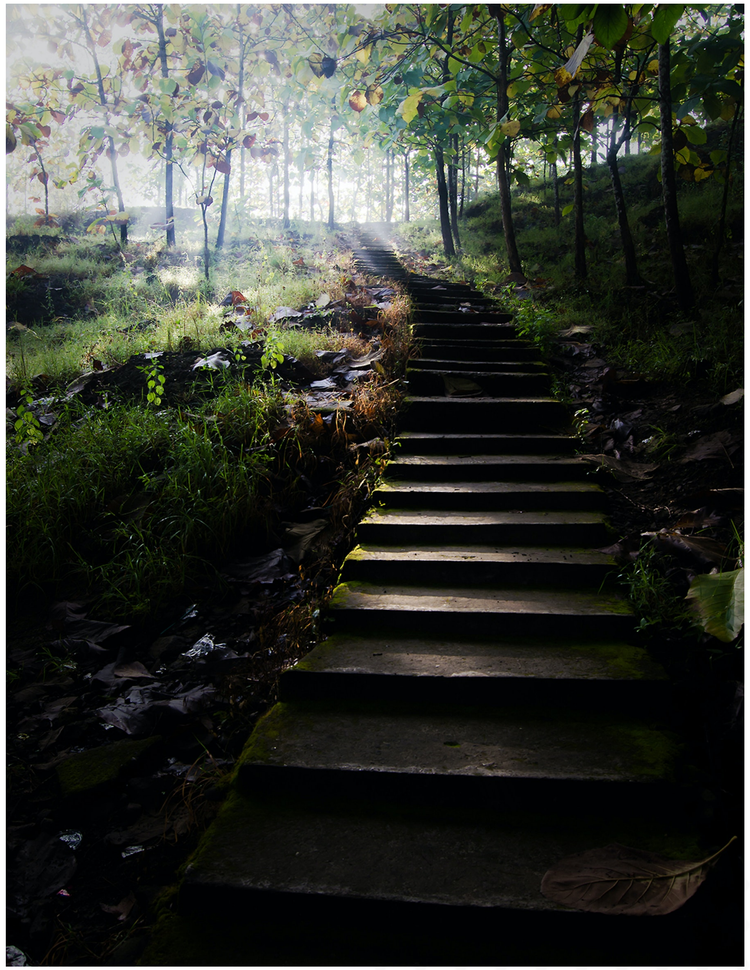
Parameters are bus passengers

Routed components are like bus stops

Passengers cannot get off at stops not on their route

Passengers cannot get off the bus before they have gotten on

Structuring URL Paths



Similar to API endpoint structure

Paths represent a hierarchy of information

Paths get more specific from left to right

Choose unique names for path parameters so there are not name collisions

Query Parameters

Appended to the end of the path

Separated from the path by a question mark

Separated from each other by ampersands

Globally available to any class

Not declared in the route

Query Parameter Handling Strategies

“ ”	“merge”	“preserve”
Replace existing parameters with new parameters ***This is the default behavior***	Merge new parameters with existing parameters during navigation	Preserve current parameters during navigation

Matrix URL Parameters

- Appended to path segments
- Separated from the segment by a semicolon
- Separated from each other by semicolons
- They can be on more than one path segment
- Only available on the activated route
- Not declared on the route

Demo

Adding a required path parameter

Single Source of Truth

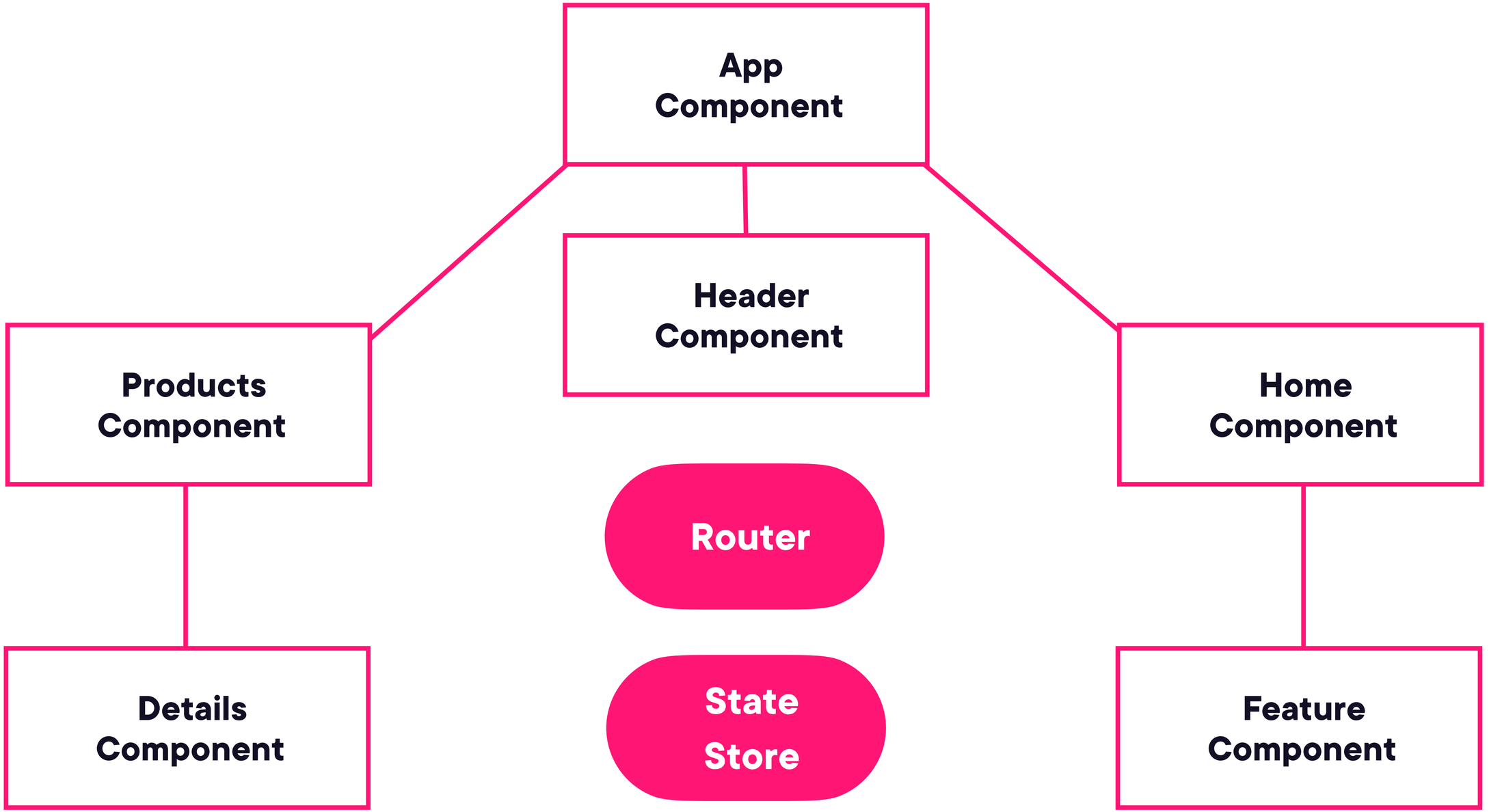
A central repository for collecting data so that all consumers receive the same data at the same time

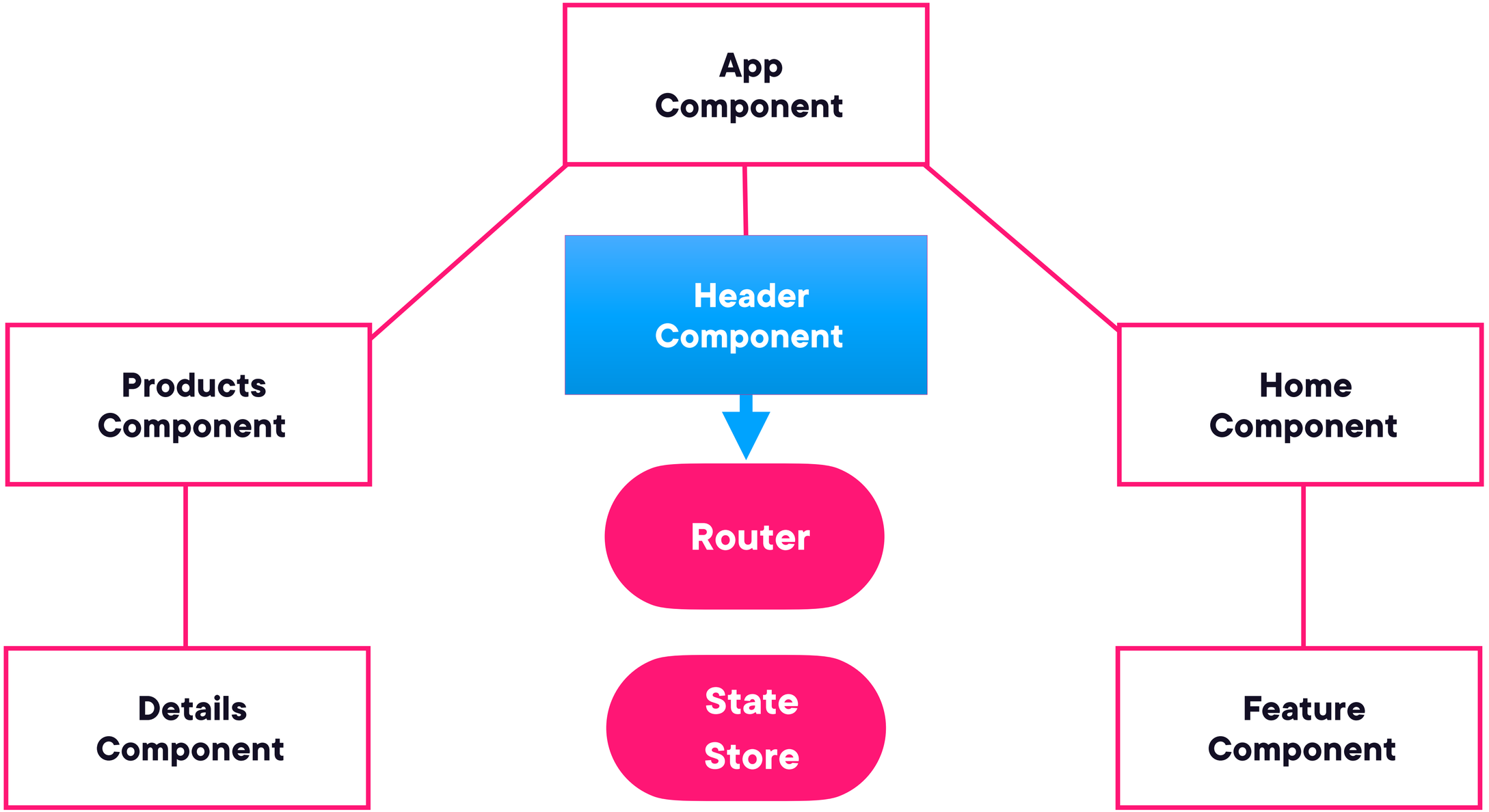
Using the Router as a Global Store

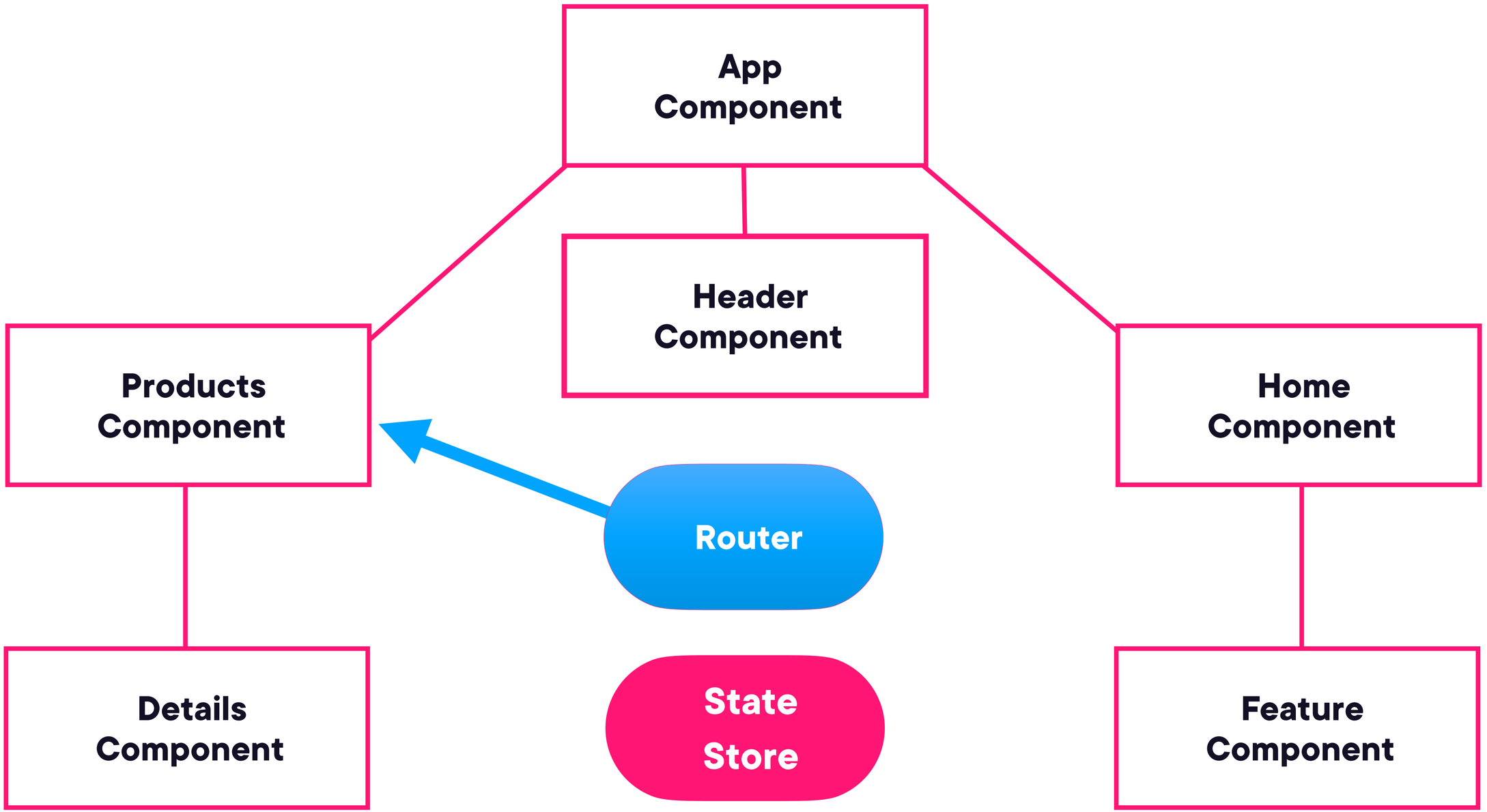
RouterService is globally available

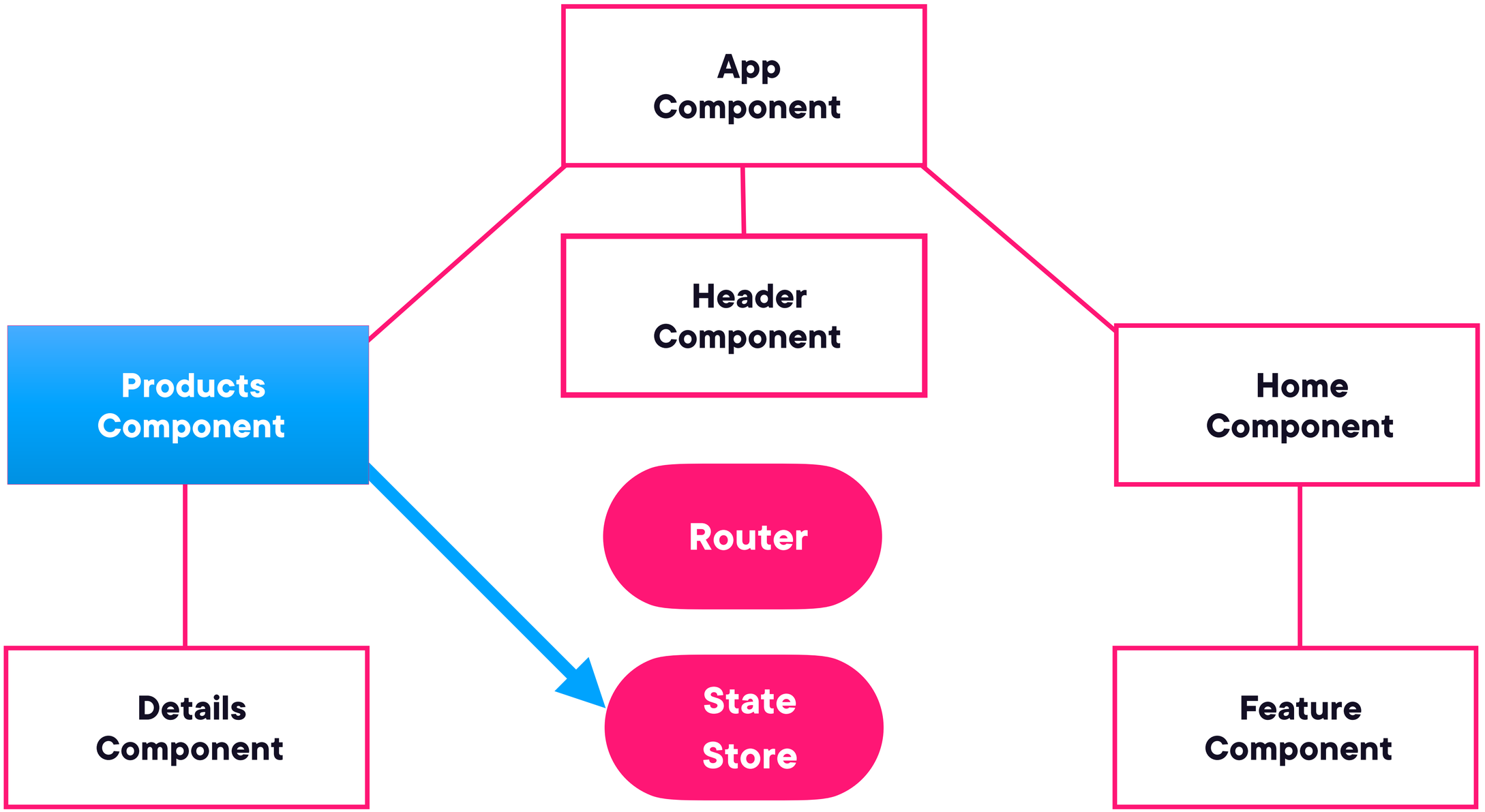
Can drive state changes

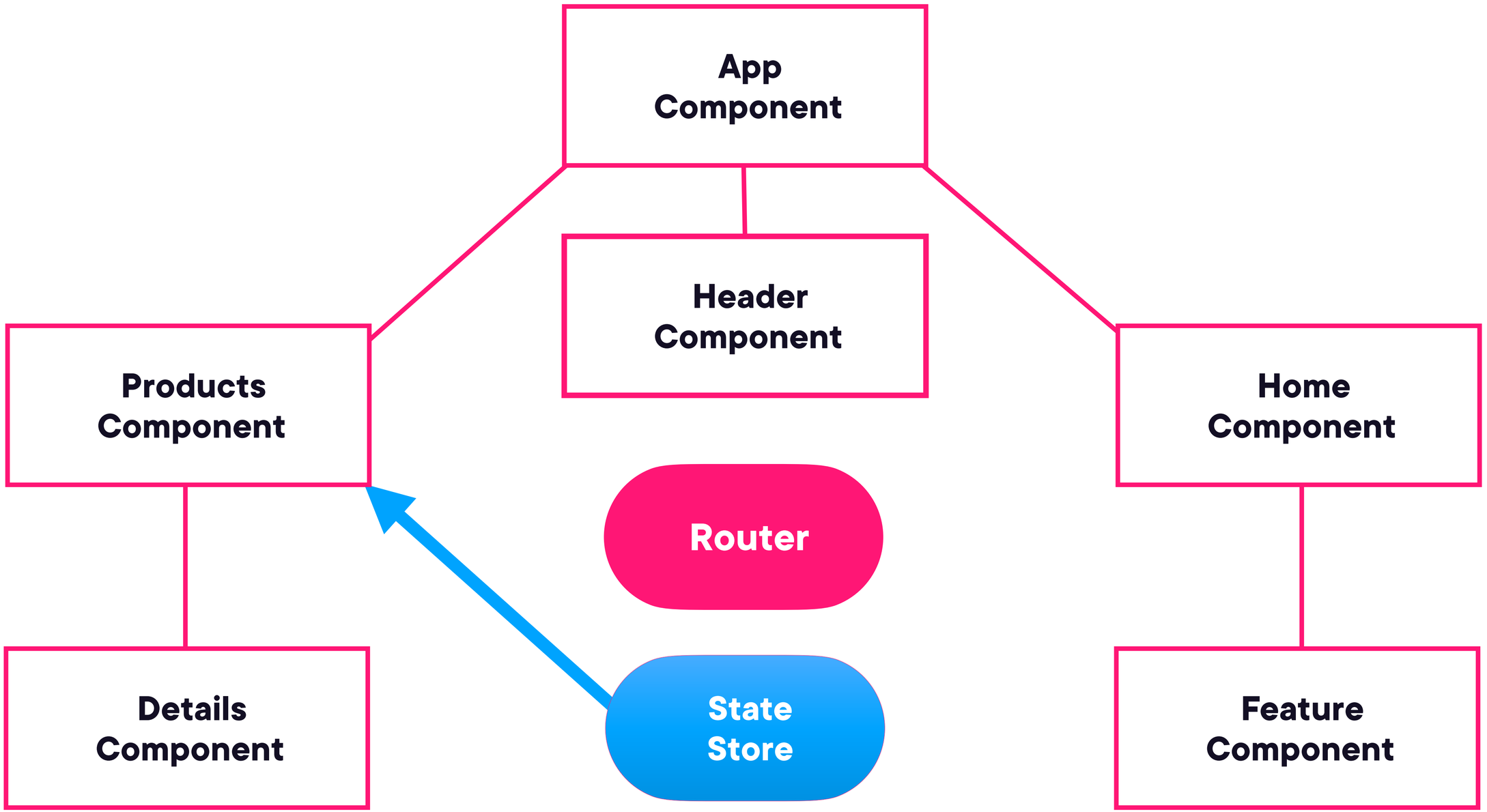
Path parameters can be consumed from the route then set is a stateful store

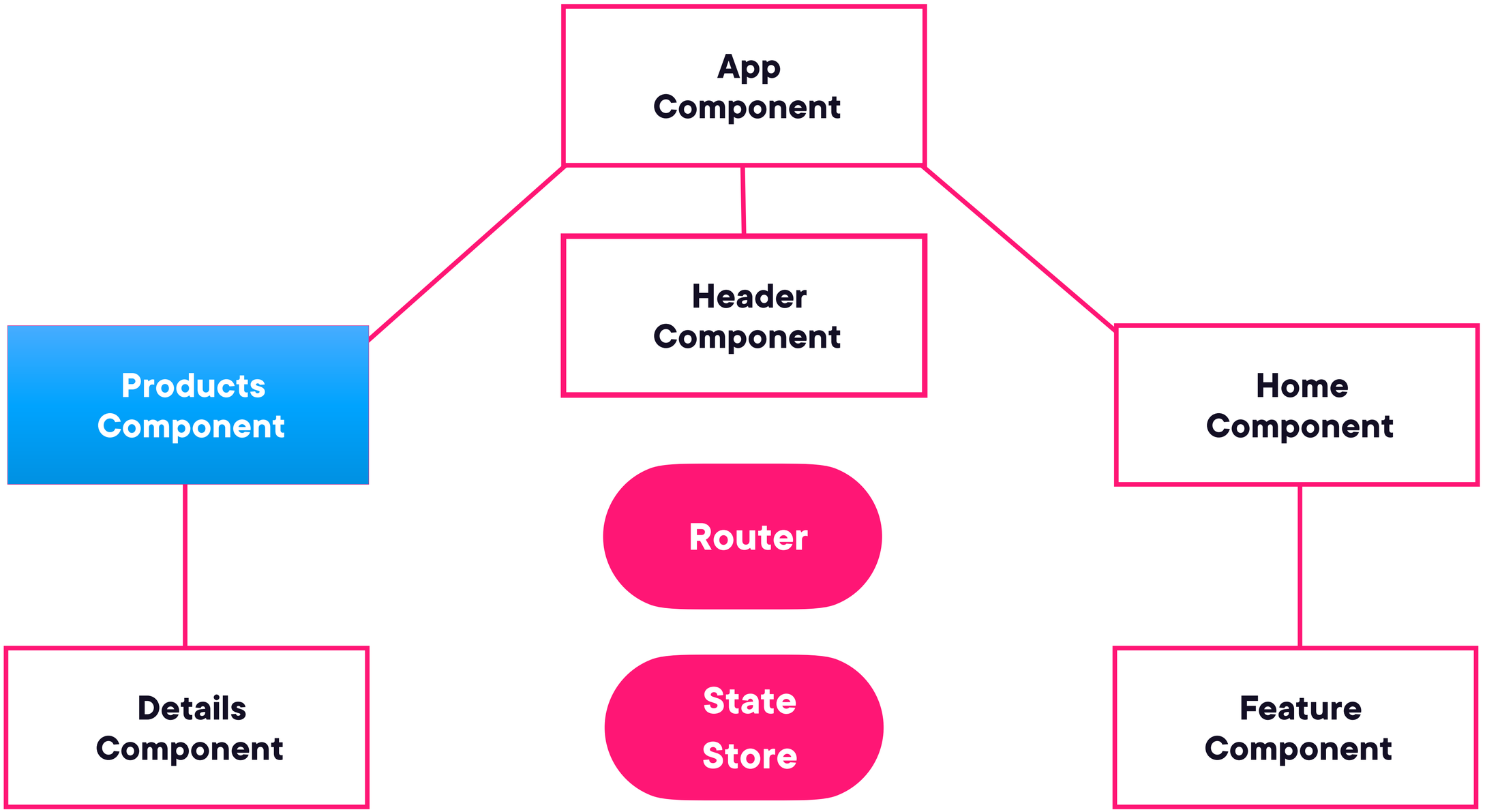

App
Component
Header
Component
Products
Component
Home
Component
Router
Details
Component
State
Store
Feature
Component

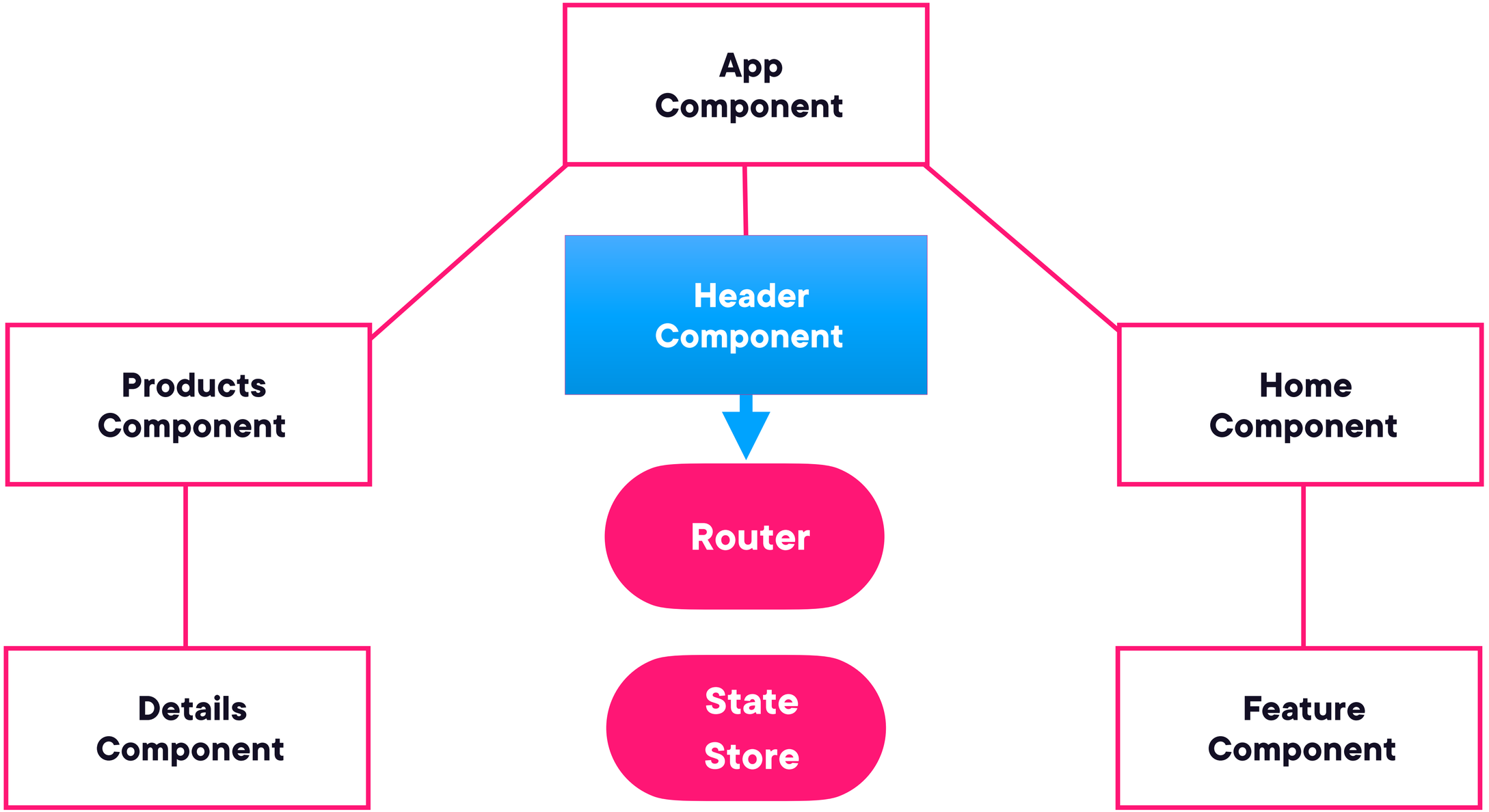

App
Component
Header
Component
Products
Component
Home
Component
Router
Details
Component
State
Store
Feature
Component

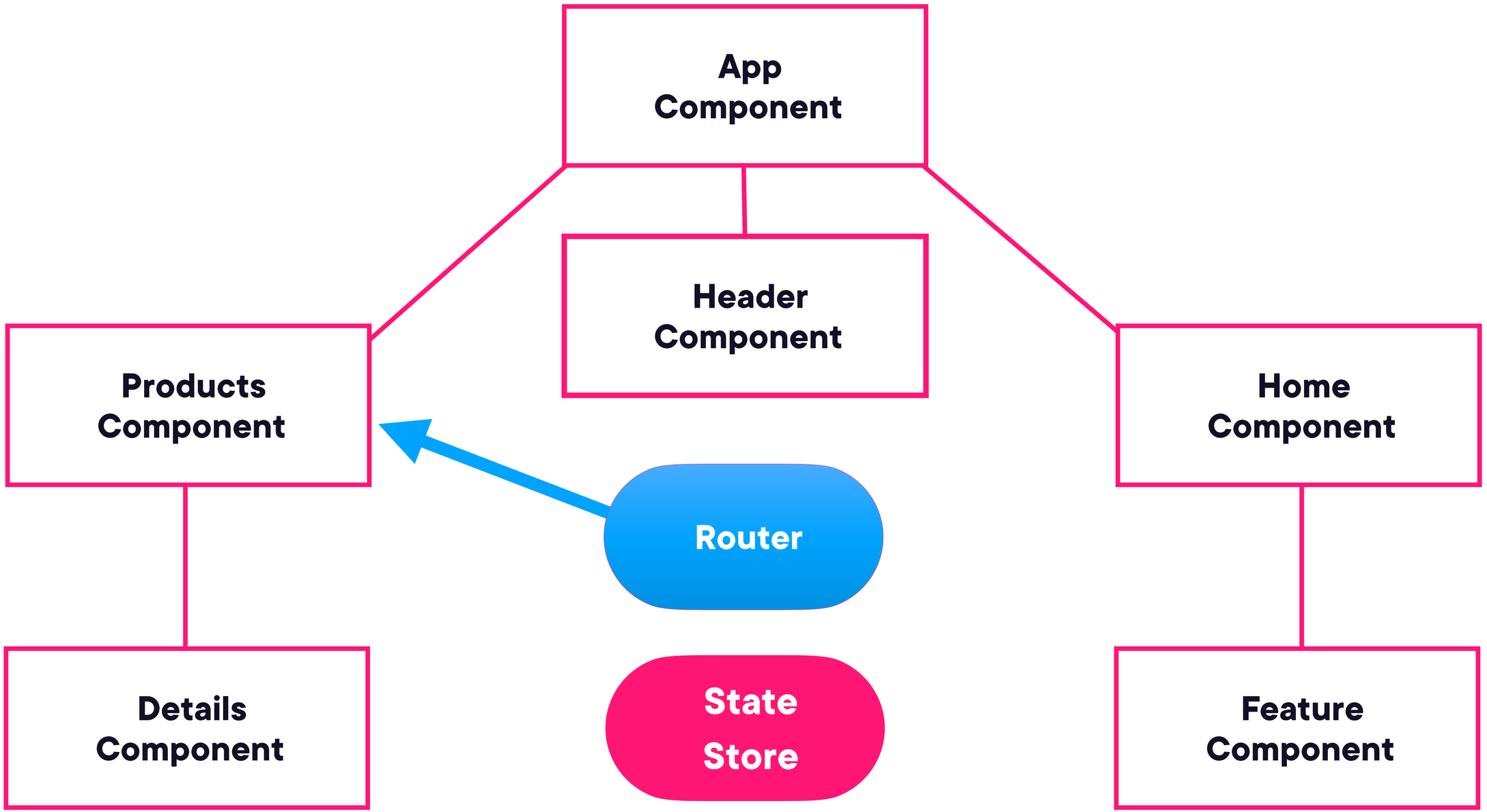

App
Component
Header
Component
Products
Component
Home
Component
Router
Details
Component
State
Store
Feature
Component

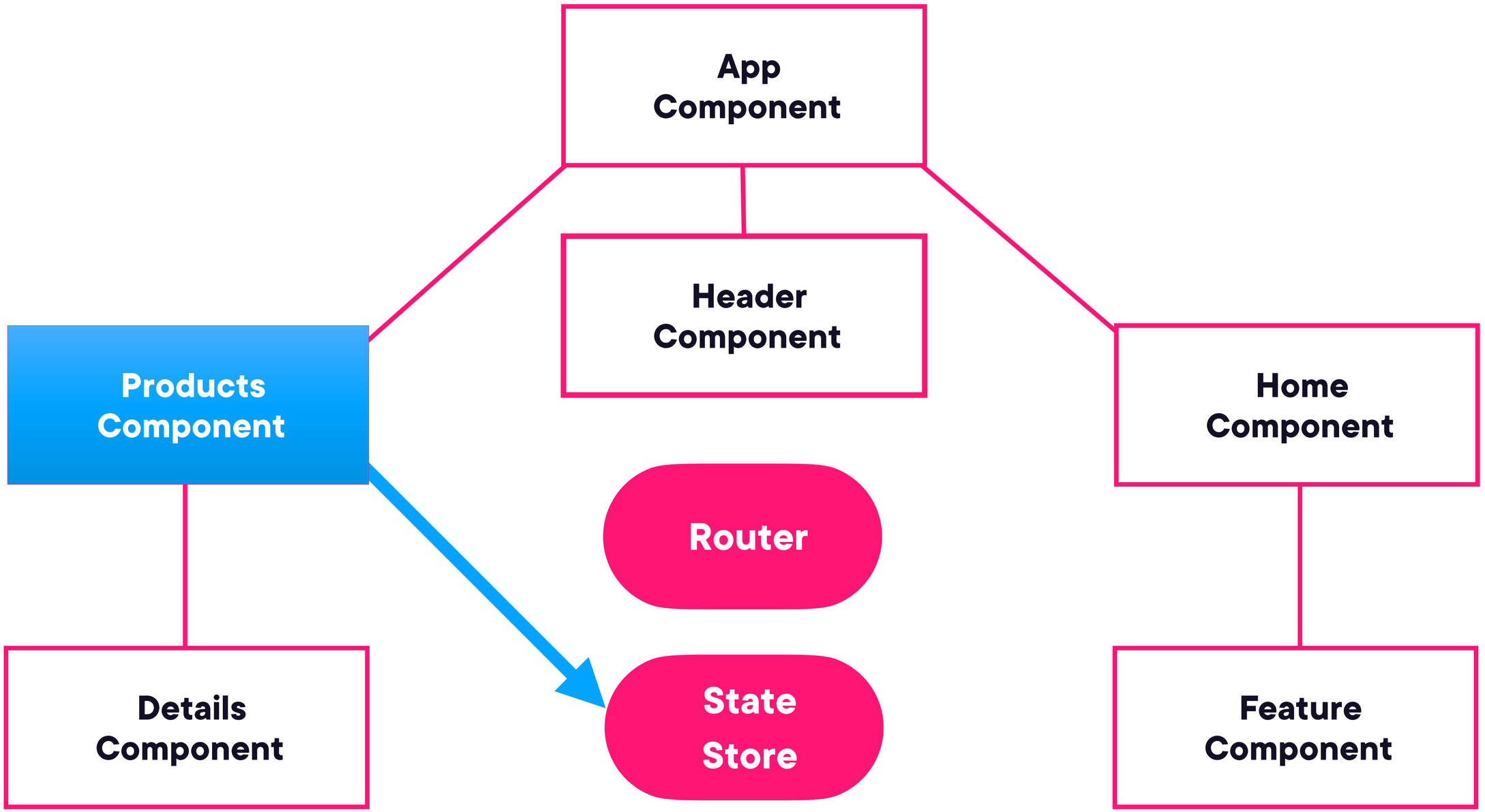

App
Component
Header
Component
Products
Component
Home
Component
Router
Details
Component
State
Store
Feature
Component

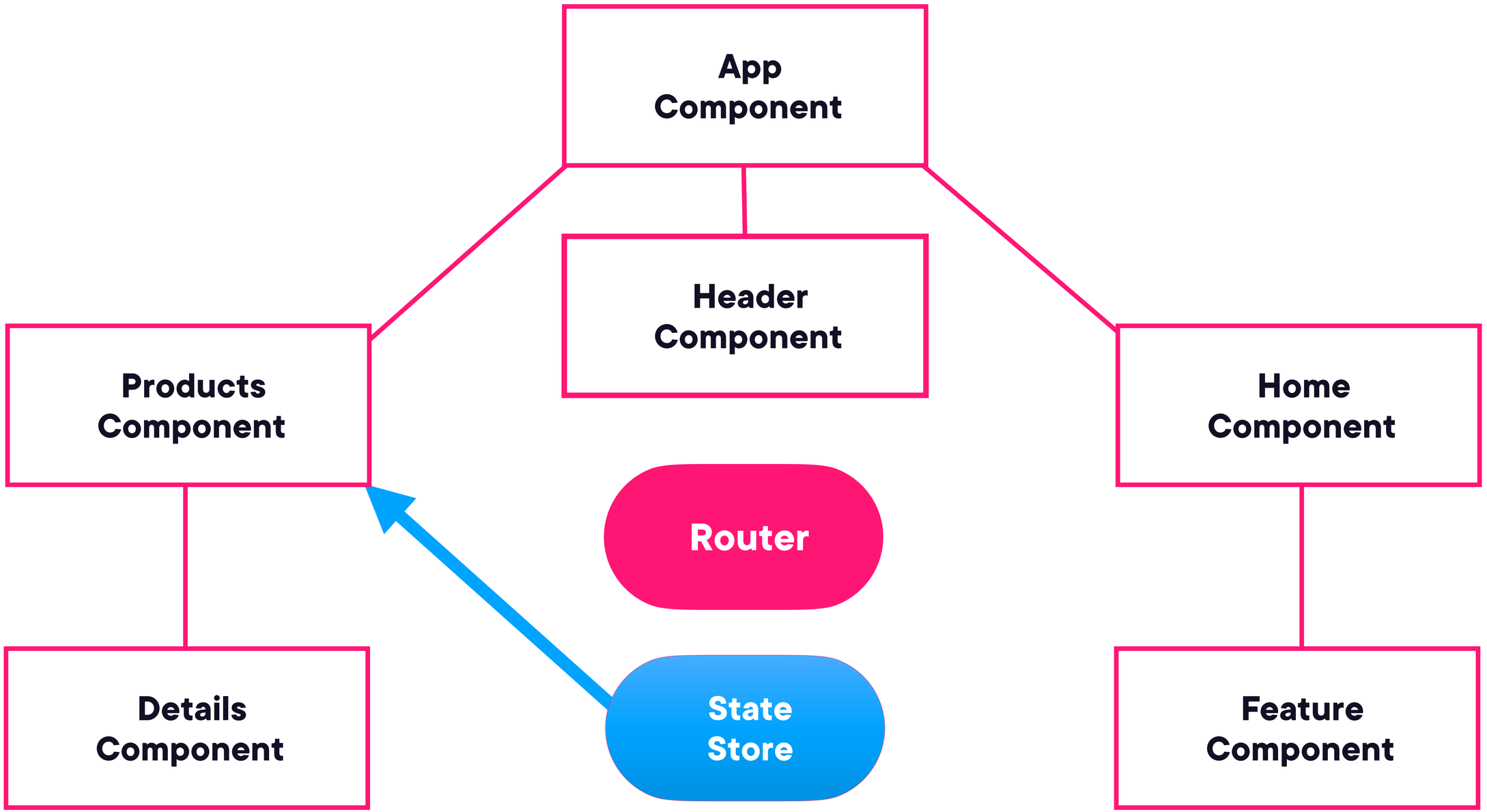

App Component
Header Component
Products Component
Home Component
Router
State Store
Details Component
Feature Component

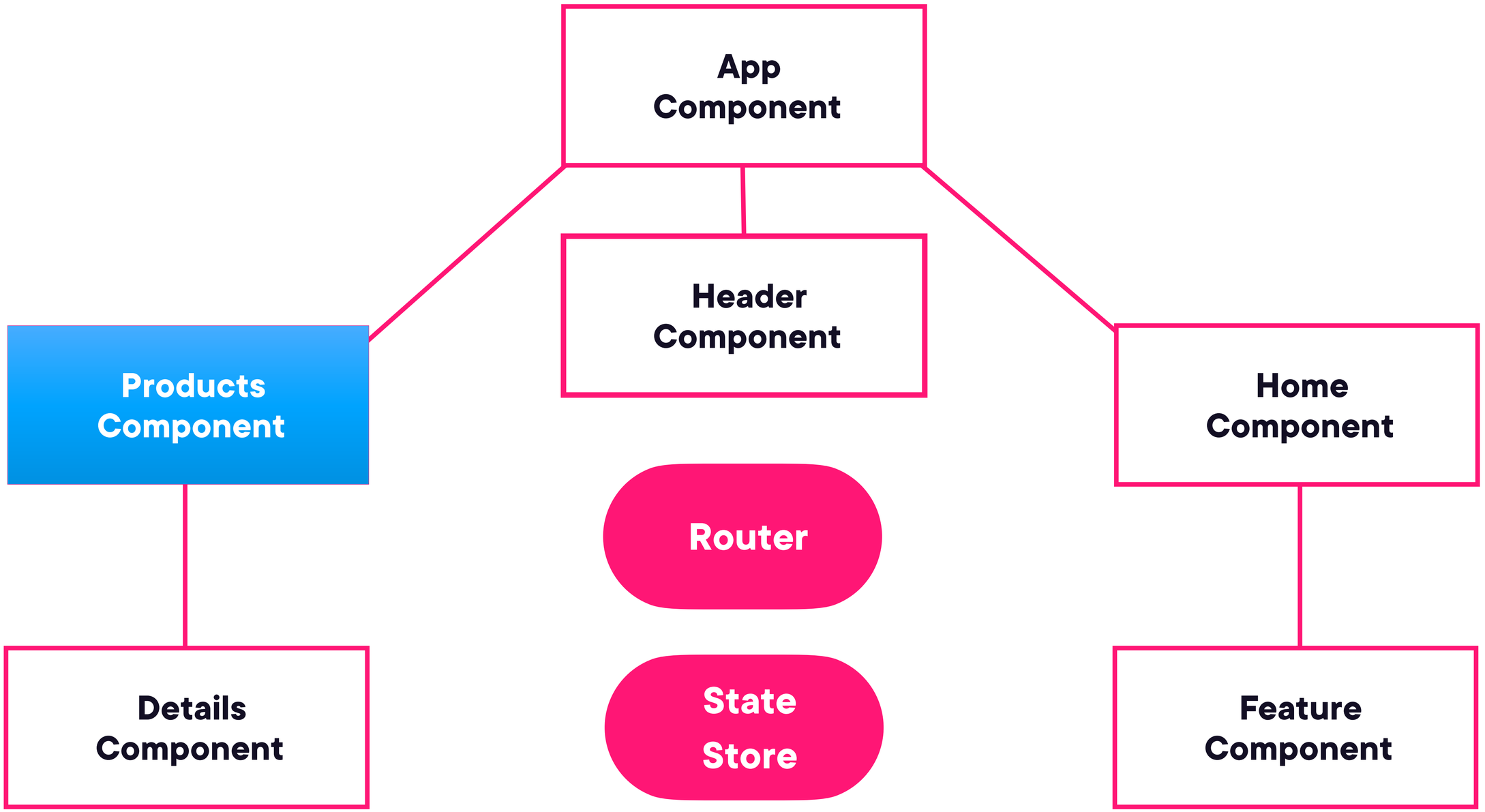

App Component
Header Component
Products Component
Home Component
Router
State Store
Details Component
Feature Component


App
Component
Header
Component
Products
Component
Home
Component
Router
Details
Component
State
Store
Feature
Component

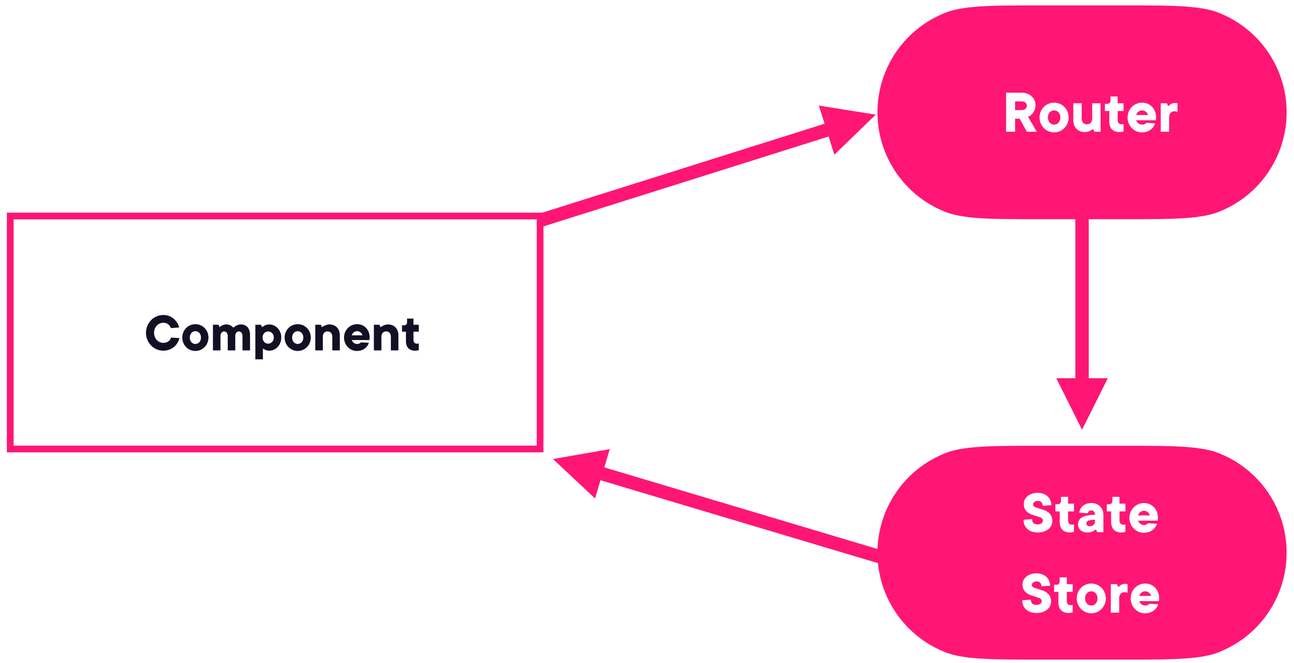

App
Component
Header
Component
Products
Component
Home
Component
Router
Details
Component
State
Store
Feature
Component


App
Component
Header
Component
Products
Component
Home
Component
Router
Details
Component
State
Store
Feature
Component


App Component
Header Component
Products Component
Home Component
Router
State Store
Details Component
Feature Component


App Component
Header Component
Products Component
Home Component
Router
State Store
Details Component
Feature Component

Driving State Changes Through the Router



State should flow through the application in a clear and predictable way

Consuming Path Parameters

Version 16+, bind directly to component input properties

Subscribe to the ParamsMap observable property from the ActivatedRoute

Use a setter on the input parameter to automatically invoke code when the value changes

Demo

Consuming path parameters in components

Understanding optional parameters

Optional Angular Parameters

Query Parameters

Matrix Parameters

Matrix URL Parameters

Have always been supported by the Angular Router

Only available on the activated route

Part of the URL path segment

Cannot be retained when the route changes

Adding Matrix Parameters

http://my-app.com/routePath;showModal=true;

```
<button
  [routerLink]="['/routePath', { showModal: true }]"
>Show Modal</button>
```

Adding Matrix Parameters

http://my-app.com/routePath;showModal=true;

```
showModal(){
  this.router.navigate(['routePath', { showModal: true }], {
    relativeTo: this.activateRoute
  });
}
```

Consuming Matrix Parameters

```
http://my-app.com/routePath;showModal=true;
```

```
export class DetailComponent {

  @Input() showModal = false;
```

Query Parameters

Separate from the path

Add from any class using the router or routerLink

Any class can consume them from the activated route

Can set handling strategy

Adding Query Parameters

http://my-app.com/routePath?showModal=true

```
<button
  routerLink="['detail']"
  [queryParams]="{ showModal: true }"
  queryParamsHandling="merge"
>Show Modal</button>
```

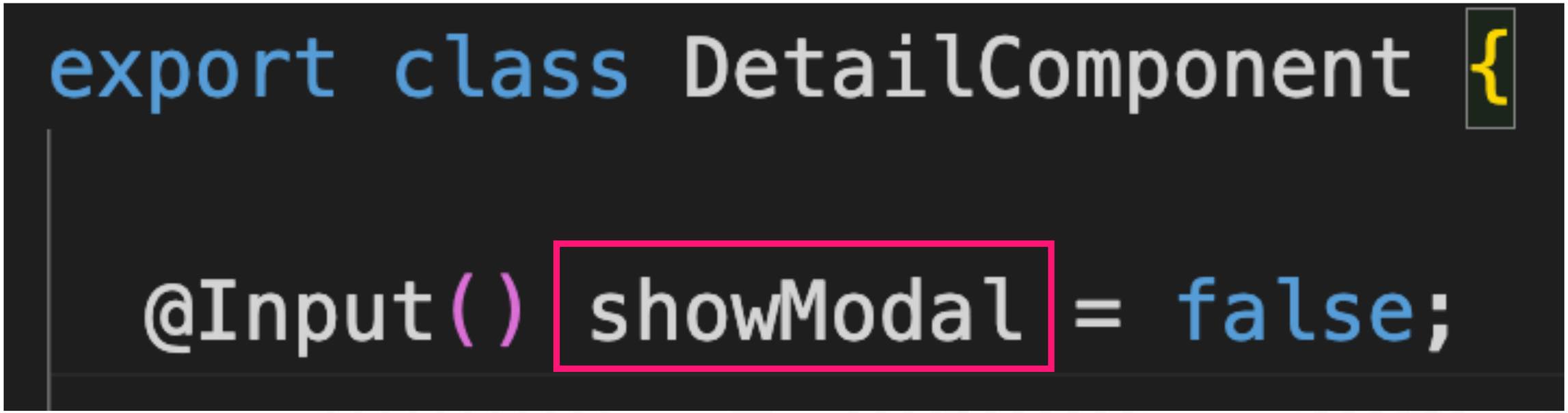
Adding Query Parameters

http://my-app.com/routePath?showModal=true

```
showModal(){
  this.router.navigate(['detail'], {
    queryParams: { showModal: true },
    queryParamsHandling: 'merge'
  })
}
```

Consuming Query Parameters

`http://my-app.com/routePath?showModal=true`



Since query parameters are globally available they are already a single source of truth for their values

ActivatedRoute

Gathers information about the route associated with the component loaded in the router outlet

Represents a router tree

Static properties are a snapshot from when the route is first activated

Observable properties emit new values as route information changes

Getting Query Params from the Activated Route

QueryParamMap	VS	QueryParams
Observable of ParamMap		**Observable of Params**
Built in get method to retrieve parameter values by key		**Values must be retrieved from the object directly by key**
Can retrieve multiple values for the same key using getAll		

Placeholder to show Angular docs on activated route

Demo

Adding and consuming query parameters

Summary

- Three types of router parameters
- Added and consumed a path parameter
- Added and consumed a query parameter
- Defined clear flow of state
- Created a reactive system that updates all consumers at the same time

Up Next:

Creating Nested Router Outlets and Auxiliary Routes