AngularJS

Building Nested Components

Using a Component

As a Directive

```
<body>
  <nat-app>Loading app...</nat-app>
</body>
```

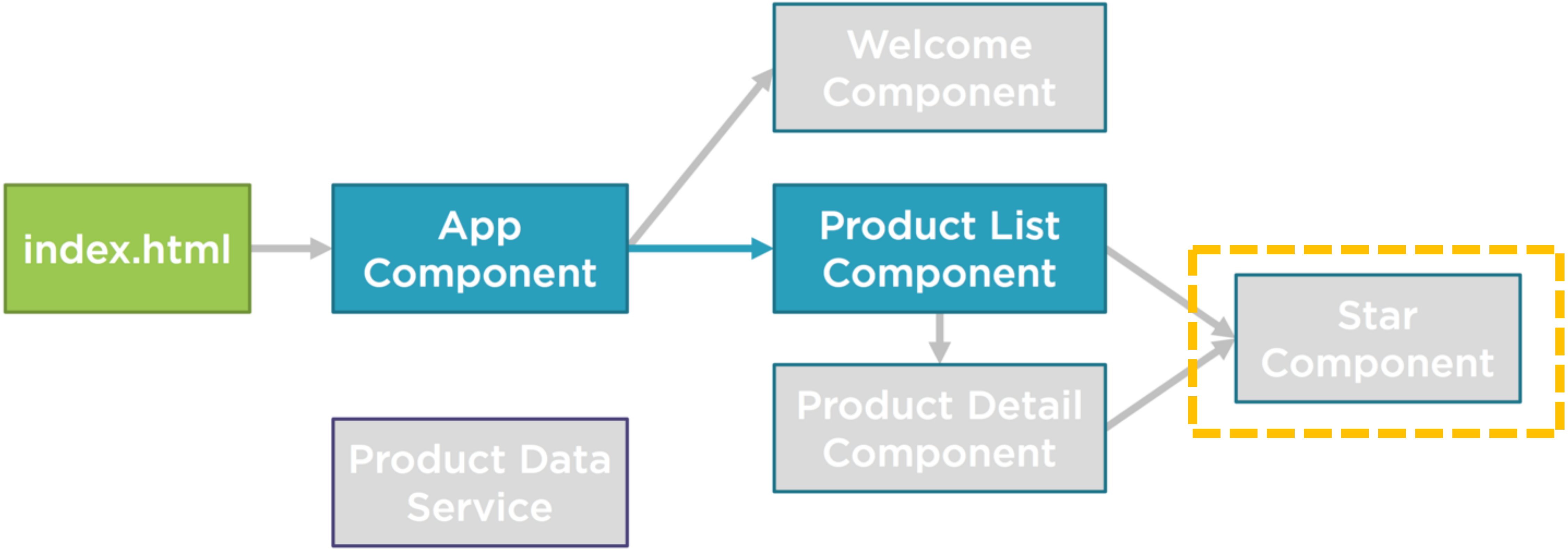
As a Routing Target

```
// Routes config
{ path: 'products', component:
ProductListComponent }
```

Module Overview

- Building a Nested Component
- Using a Nested Component
- Passing data to a Nested Component using @Input
- Raising an Event from a Nested Component using @Output

Application Architecture



index.html

App Component

Welcome Component

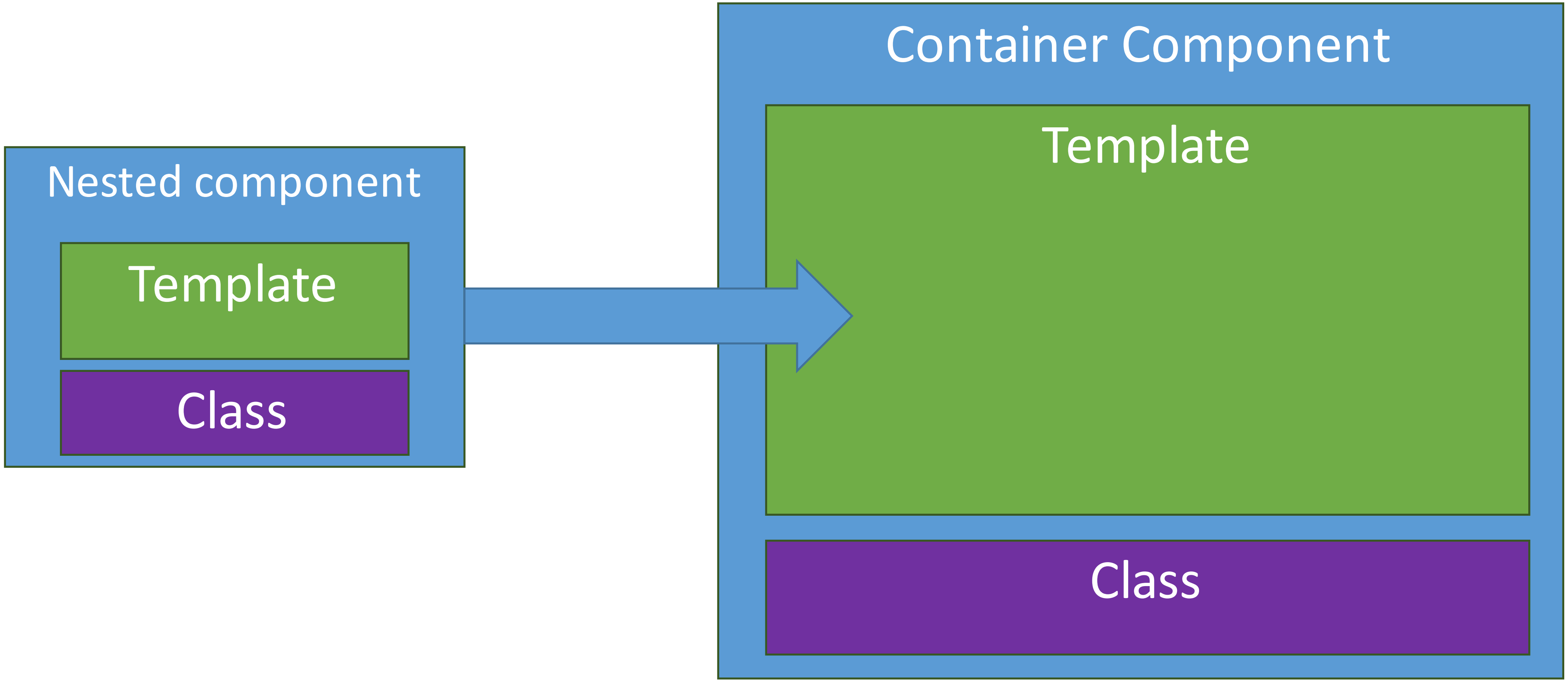
Product List Component

Product Detail Component

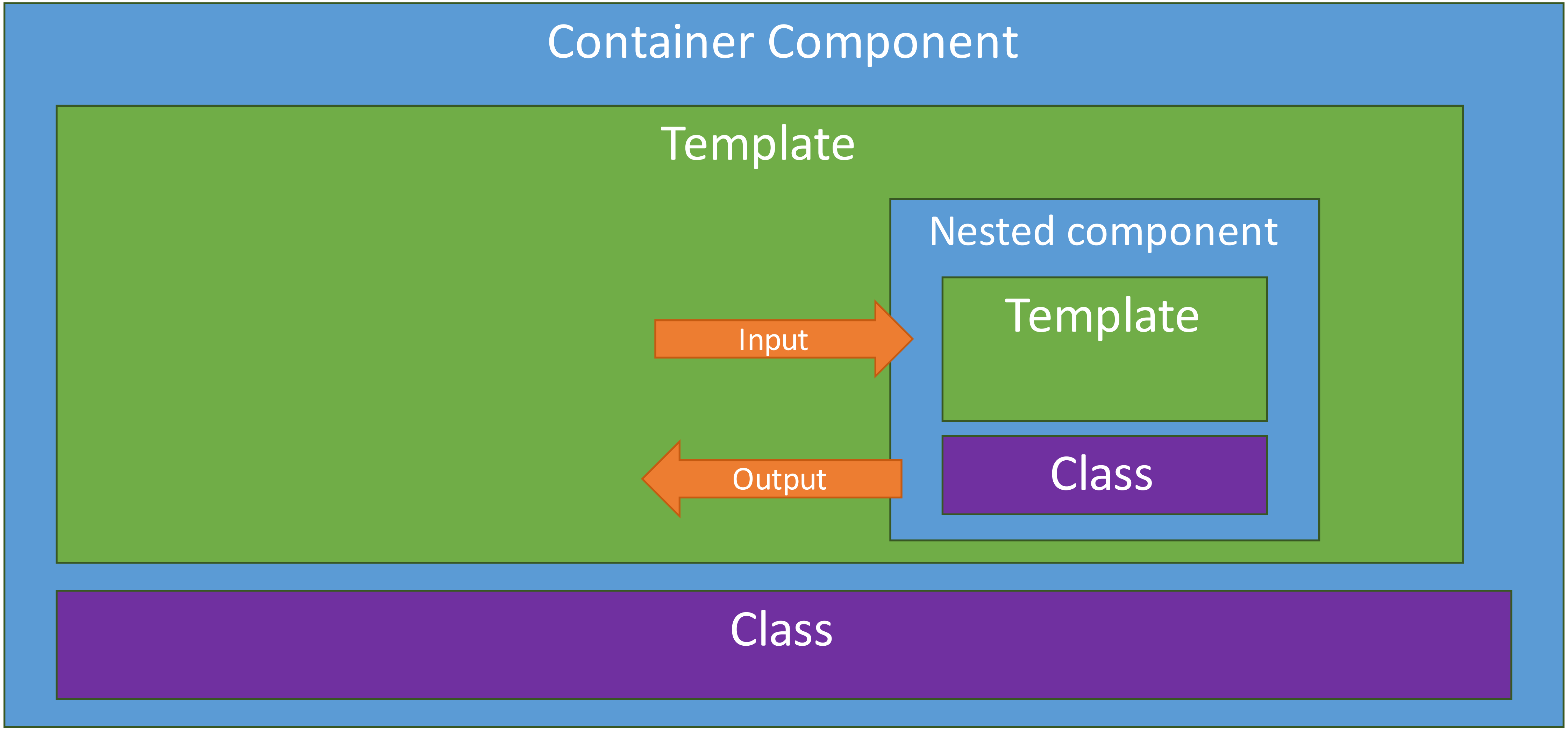
Star Component

Product Data Service

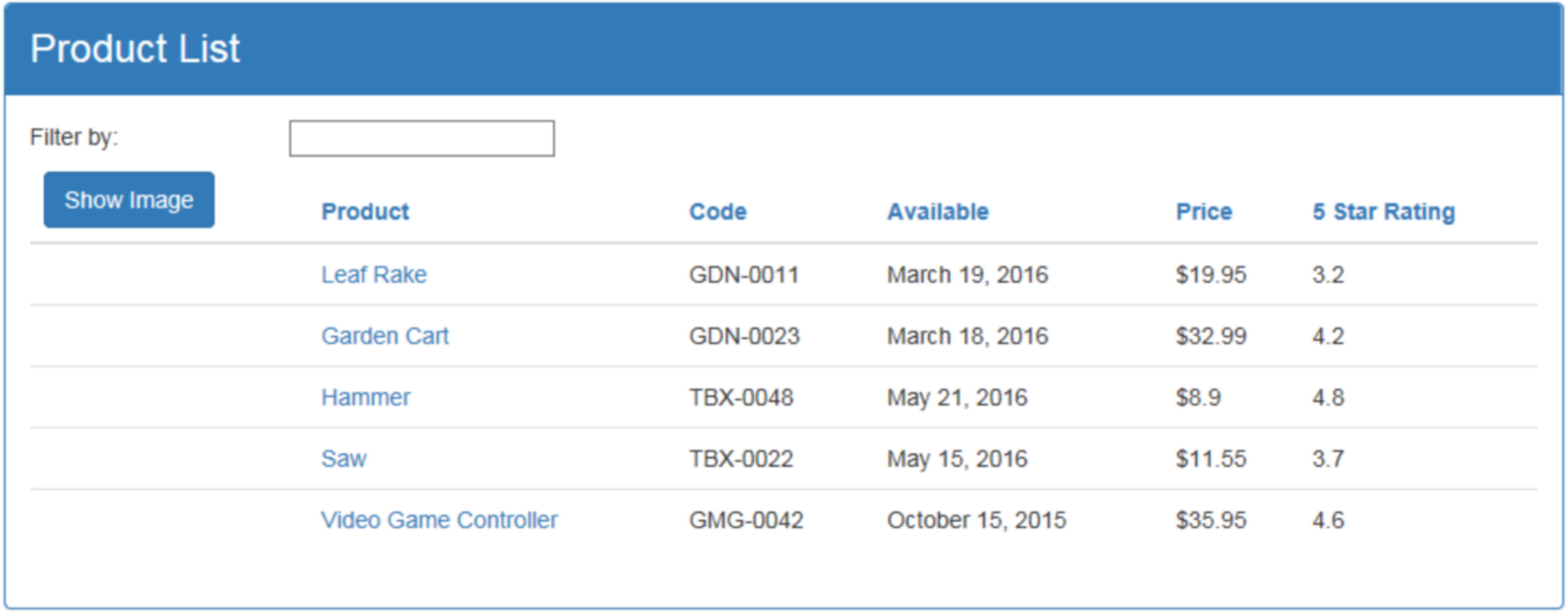
Building a Nested Component



Building a Nested Component



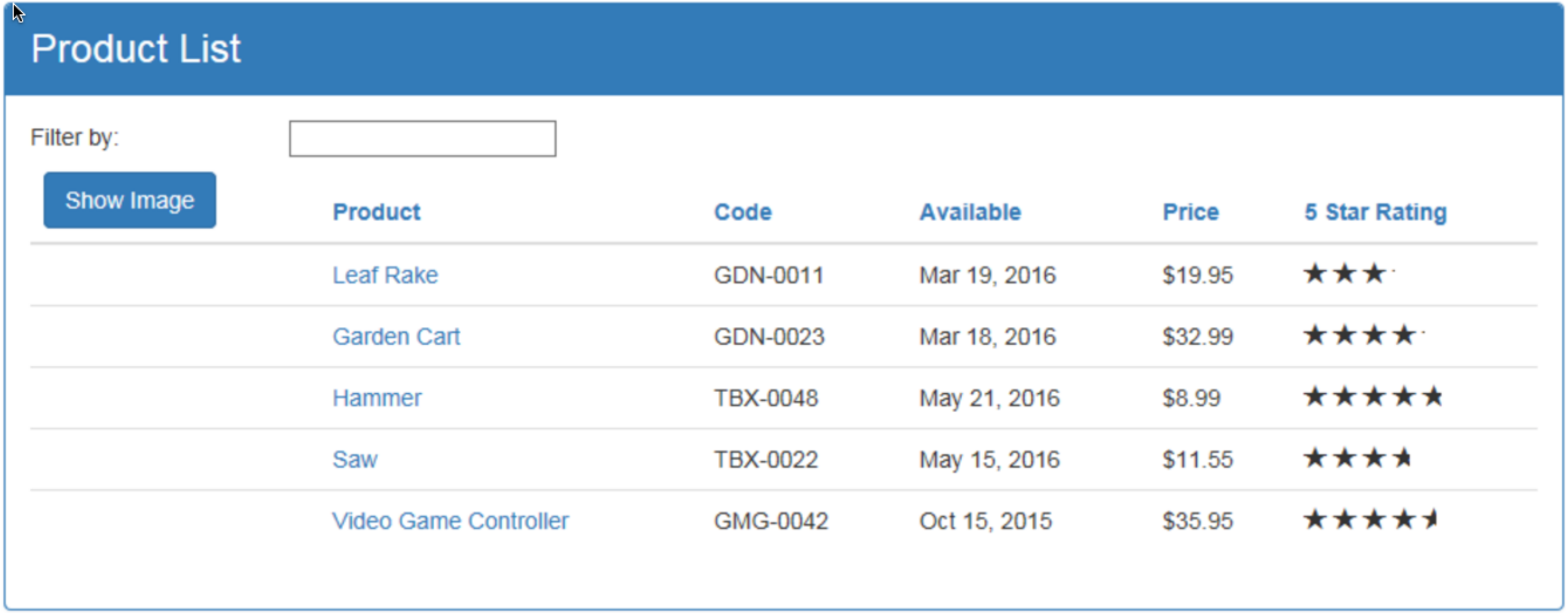
Product list view



Product List

Product	Code	Available	Price	5 Star Rating
Leaf Rake	GDN-0011	March 19, 2016	$19.95	3.2
Garden Cart	GDN-0023	March 18, 2016	$32.99	4.2
Hammer	TBX-0048	May 21, 2016	$8.9	4.8
Saw	TBX-0022	May 15, 2016	$11.55	3.7
Video Game Controller	GMG-0042	October 15, 2015	$35.95	4.6

Product list view



Product List

Filter by:

Show Image

Product	Code	Available	Price	5 Star Rating
Leaf Rake	GDN-0011	Mar 19, 2016	$19.95	★★★
Garden Cart	GDN-0023	Mar 18, 2016	$32.99	★★★★
Hammer	TBX-0048	May 21, 2016	$8.99	★★★★★
Saw	TBX-0022	May 15, 2016	$11.55	★★★★
Video Game Controller	GMG-0042	Oct 15, 2015	$35.95	★★★★★

Using a nested component as a directive

```
// product-list.components.ts

import { Component } from '@angular/core';
@Component({
  selector: 'nat-products',
  templateUrl: 'app/products/product-
list.component.html'
})
export class ProductListComponent {}
```

```
// product-list.component.html

<td>
  {{ product.starRating | number }}
</td>
```

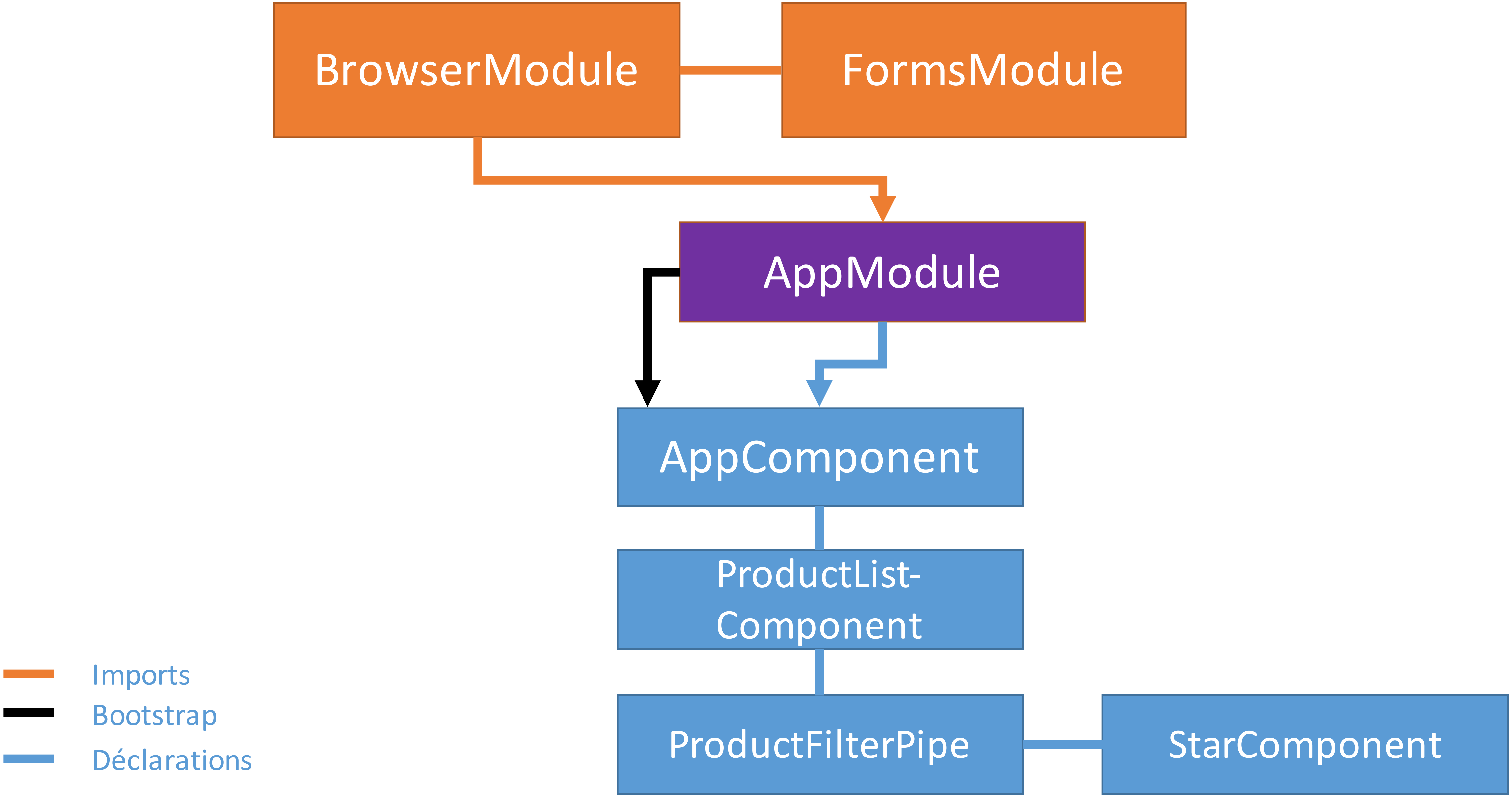
```
// star.component.ts

import { Component } from '@angular/core';
@Component({
  selector: 'nat-star',
  templateUrl: 'app/shared/star.component.html'
})
export class StarComponent {
  rating: number;
  starWidth: number;
}
```

Using a nested component as a directive

```
// product-list.components.ts

import { Component } from '@angular/core';
@Component({
  selector: 'nat-products',
  templateUrl: 'app/products/product-
list.component.html'
})
export class ProductListComponent {}
```

```
// product-list.component.html

<td>
  <nat-star></nat-star>
</td>
```

```
// star.component.ts

import { Component } from '@angular/core';
@Component({
  selector: 'nat-star',
  templateUrl: 'app/shared/star.component.html'
})
export class StarComponent {
  rating: number;
  starWidth: number;
}
```



BrowserModule

FormsModule

AppModule

AppComponent

ProductList-Component

ProductFilterPipe

StarComponent

Imports

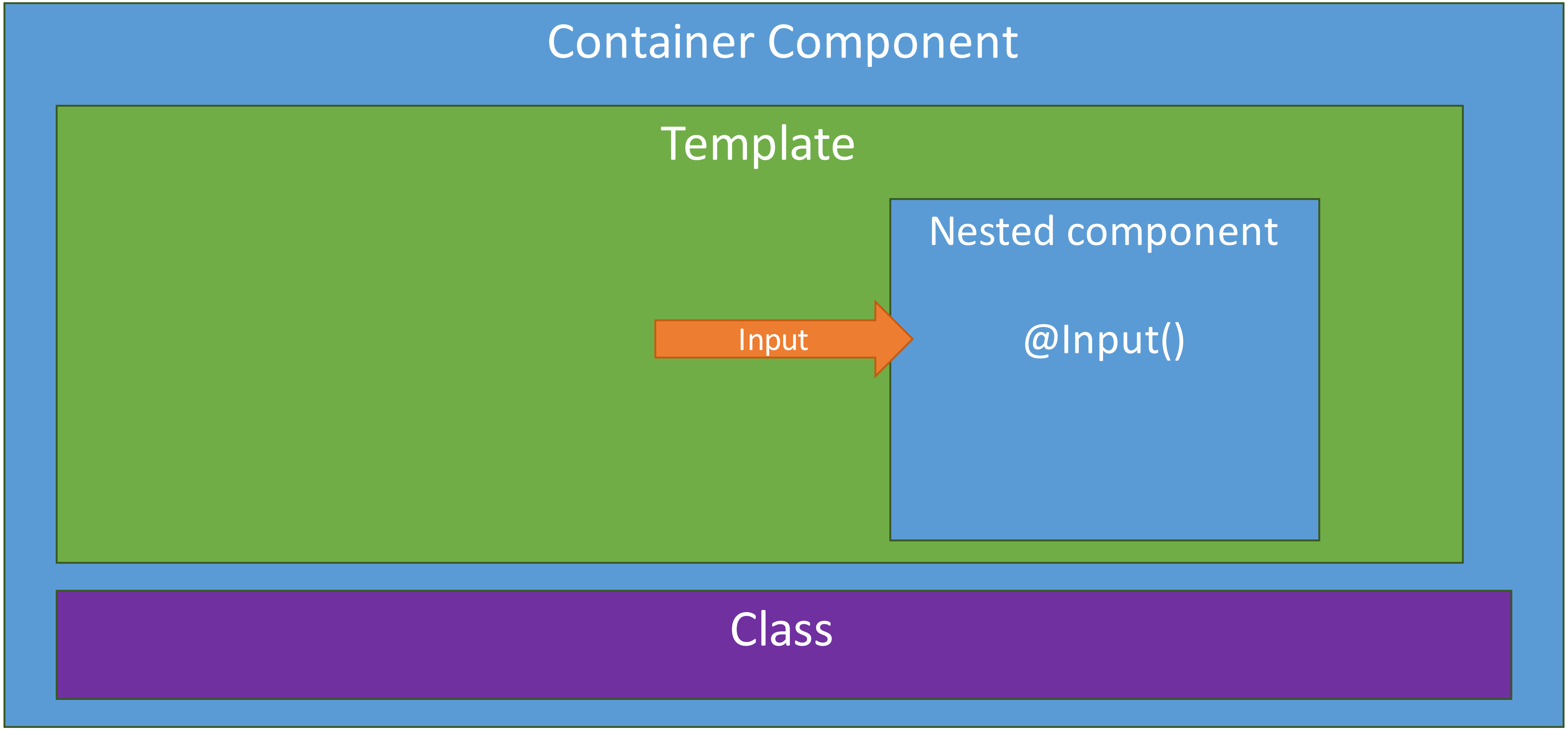
Bootstrap

Déclarations

Declare our component into AppModule

```
// app.module.ts
import { StarComponent} from 'shared/star.component';
@NgModule({
  imports: [
    BrowserModule, FormsModule
  ],
  declarations: [
    AppComponent, ProductListComponent,
    ProductFilterPipe, StarComponent
  ],
  bootstrap: [ AppComponent ]
})
export class AppModule {}
```

Passing data to a nested component (@Input)



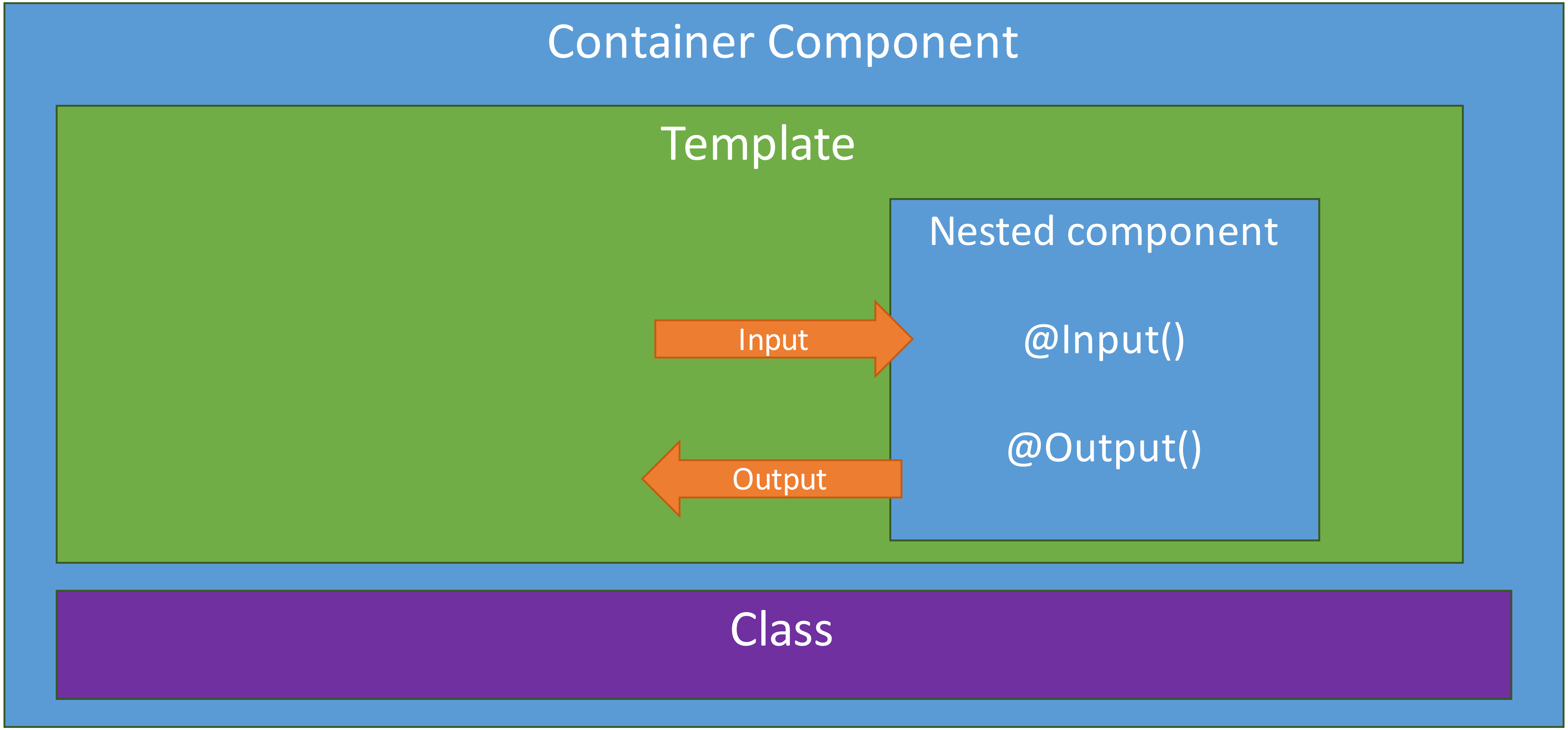
Passing data to a nested component (@Input)

```
// product-list.components.ts

import { Component } from '@angular/core';
@Component({
  selector: 'nat-products',
  templateUrl: 'app/products/product-
list.component.html'
})
export class ProductListComponent {}
```

```
// product-list.component.html

<td>
  <nat-star [rating]="product.starRating"></nat-star>
</td>
```

```
// star.component.ts

import { Component, Input } from
'@angular/core';
@Component({
  selector: 'nat-star',
  templateUrl: 'app/shared/star.component.html'
})
export class StarComponent {
  @Input() rating: number;
  starWidth: number;
}
```

Passing data to a nested component (@Input)



Container Component

Template

Nested component

Input

@Input()

@Output()

Output

Class

Raising an Event (@Output)

```
// product-list.components.ts
import { Component } from '@angular/core';
@Component({
  selector: 'nat-products',
  templateUrl: 'app/products/product-
list.component.html'
})
export class ProductListComponent {
  onNotify(message: string): void {}
}
```

```
// product-list.component.html
<td>
  <nat-star [rating]="product.starRating »
    (notify)='onNotify($event)'>
  </nat-star>
</td>
```

```
// star.component.ts
import { Component, Input, Output, EventEmitter } from
'@angular/core';
@Component({
  selector: 'nat-star',
  templateUrl: 'app/shared/star.component.html'
})
export class StarComponent {
  @Input() rating: number;
  starWidth: number;
  @Output() notify: EventEmitter<string> = new
EventEmitter<string>();

  onClick() {
    this.notify.emit('clicked');
  }
}
```

```
// star.component.html
<div (click)='onClick()’>
  … stars ...
</div>
```

Checklist: Nested Component

- ✓ Input decorator
  - ✓ Attached to a property of any type
  - ✓ Prefix with @; suffic with ()
- ✓ Output decorator
  - ✓ Attached to a property declared as an EventEmitter
  - ✓ Use the generic argument to define the event payload type
  - ✓ Use the new keyword to create an instance of the EventEmitter
  - ✓ Prefix with @; Suffix with ()

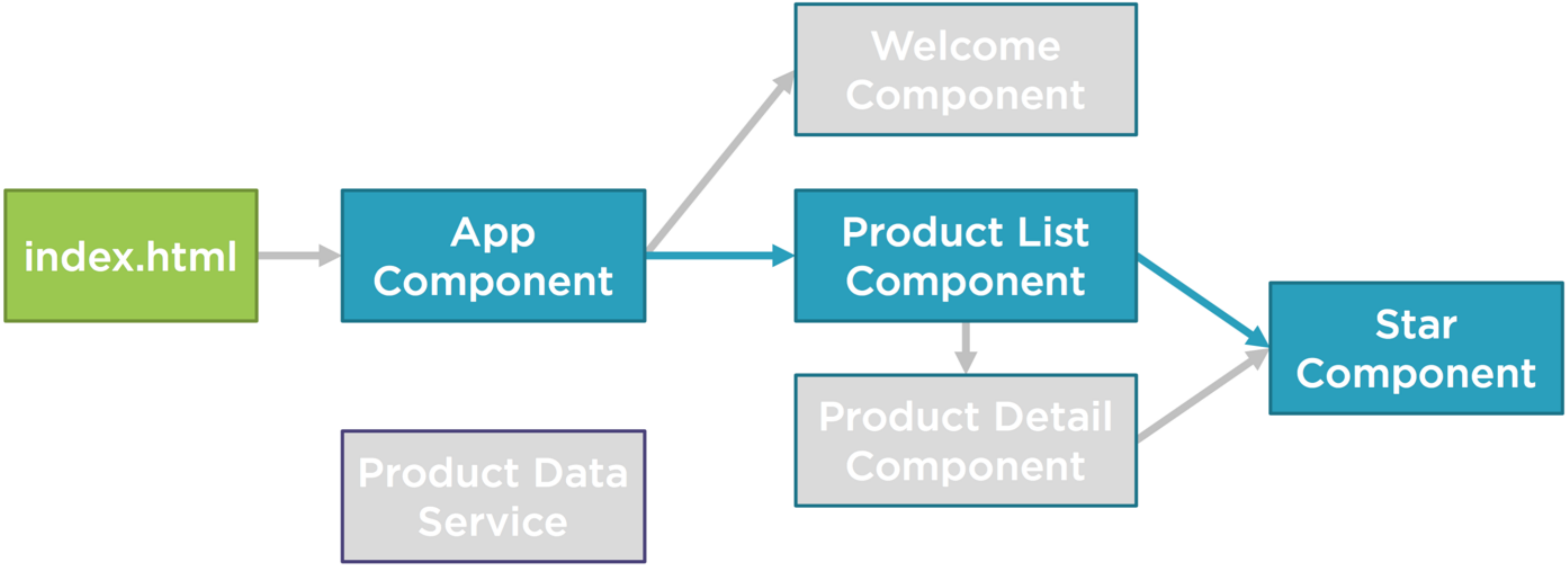
Checklist: Container Component

- ✓ Use the directive
    - ✓ Directive name -> nested component's selector
- ✓ Use property binding to pass data to the nested component
- ✓ Use event binding to respond to events from the nested component
    - ✓ Use $event to access the event payload passed from the nested component

Module Overview

- Building a Nested Component
- Using a Nested Component
- Passing data to a Nested Component using @Input
- Raising an Event from a Nested Component using @Output

Application Architecture


index.html
App Component
Welcome Component
Product List Component
Product Detail Component
Star Component
Product Data Service