



Module Overview



What Is a Component?

Creating the Component Class

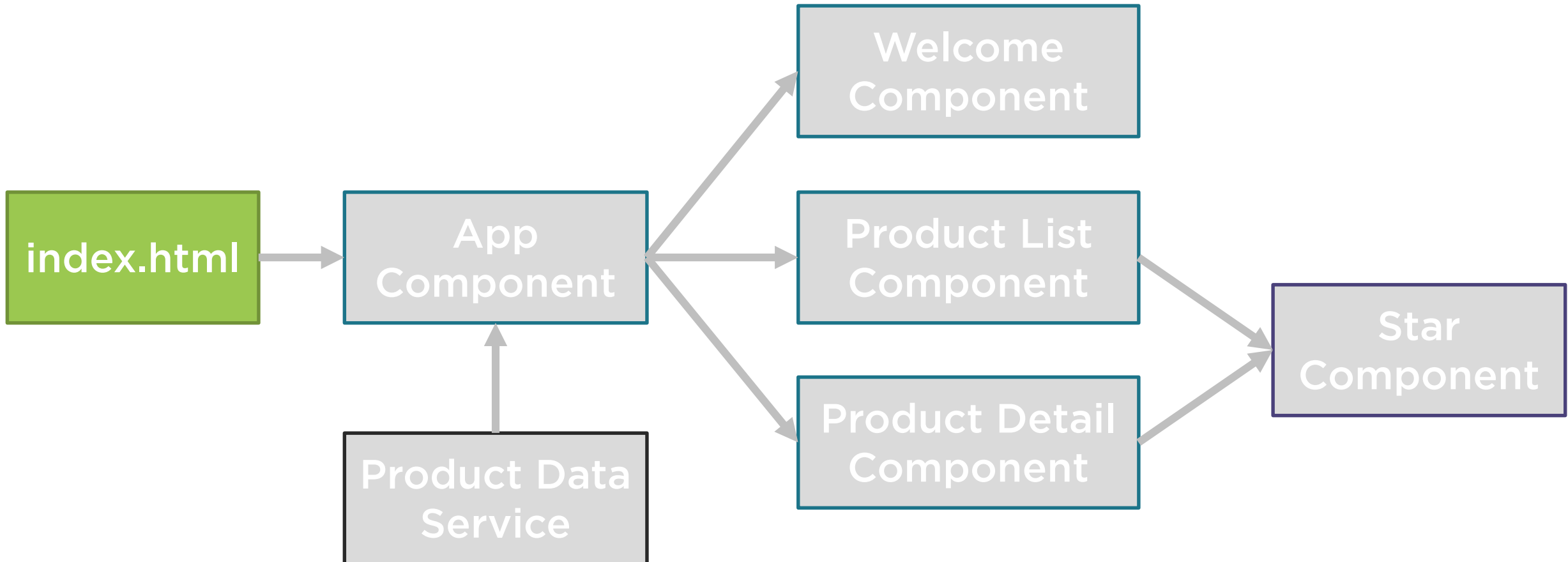
Defining the Metadata with a Decorator

Importing What We Need

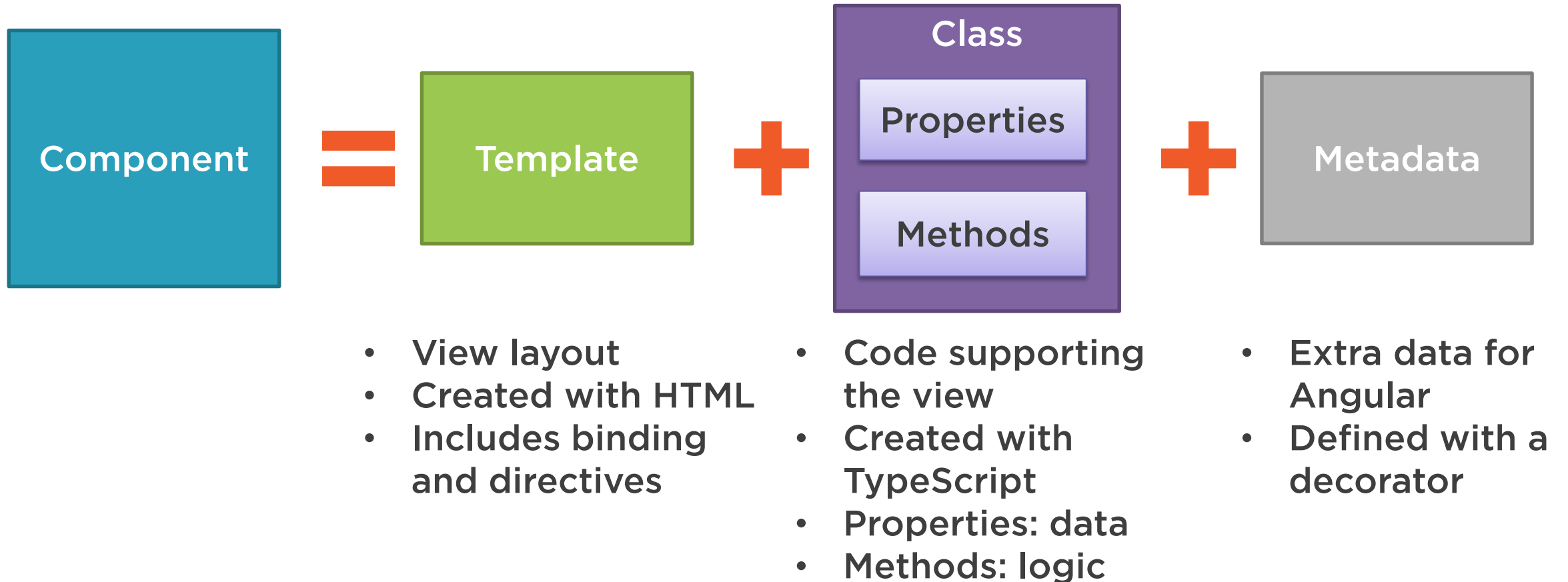
Bootstrapping Our App Component



Application Architecture



What Is a Component?



Component

app.component.ts

```
import { Component } from 'angular2/core';
```

Import

```
@Component({  
  selector: 'pm-app',  
  template: `  
    <div><h1>{{pageTitle}}</h1>  
      <div>My First Component</div>  
    </div>  
  `,  
})
```

Metadata &
Template

```
export class AppComponent {  
  pageTitle: string = 'Acme Product Management';  
}
```

Class



Creating the Component Class

app.component.ts

```
export class AppComponent {  
  pageTitle: string = 'Acme Product Management';  
}
```

class
keyword

Class Name

export
keyword

Component Name
when used in code



Creating the Component Class

app.component.ts

```
export class AppComponent {  
  pageTitle: string = 'Acme Product Management';  
}
```

Property
Name

Data Type

Default
Value

Methods



Defining the Metadata

app.component.ts

```
@Component({  
  selector: 'pm-app',  
  template: `  
    <div><h1>{{pageTitle}}</h1>  
      <div>My First Component</div>  
    </div>  
  `,  
})  
export class AppComponent {  
  pageTitle: string = 'Acme Product Management';  
}
```



Decorator

A function that adds **metadata** to a class, its members, or its method arguments.

Prefixed with an @.

Angular provides built-in decorators.

@Component()



Defining the Metadata

app.component.ts

```
@Component({
  selector: 'pm-app',
  template: `
    <div><h1>{{pageTitle}}</h1>
      <div>My First Component</div>
    </div>
  `
})
export class AppComponent {
  pageTitle: string = 'Acme Product Management';
}
```

Component
decorator

Directive Name
used in HTML

View Layout

Binding



Importing What We Need



Before we use an external function or class, we define where to find it

`import` statement

`import` allows us to use exported members from external modules

Import from a third-party library, our own modules, or from Angular

Angular Is Modular

angular2/
core

angular2/
animate

angular2/
http

angular2/
router



Importing What We Need

app.component.ts

```
@Component({  
  selector: 'pm-app',  
  template: `  
    <div><h1>{{pageTitle}}</h1>  
      <div>My First Component</div>  
    </div>  
  `,  
})  
export class AppComponent {  
  pageTitle: string = 'Acme Product Management';  
}
```



Importing What We Need

app.component.ts

```
import { Component } from 'angular2/core';
```

```
@Component({  
  selector: 'pm-app',  
  template: `  
    <div><h1>{{pageTitle}}</h1>  
      <div>My First Component</div>  
    </div>  
  `,  
})  
export class AppComponent {  
  pageTitle: string = 'Acme Product Management';  
}
```

import keyword

Angular library
module name

Member name



Completed Component

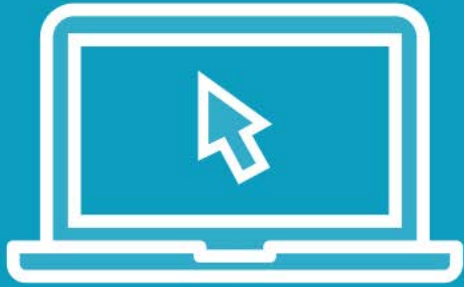
app.component.ts

```
import { Component } from 'angular2/core';

@Component({
  selector: 'pm-app',
  template: `
    <div><h1>{{pageTitle}}</h1>
      <div>My First Component</div>
    </div>
  `
})
export class AppComponent {
  pageTitle: string = 'Acme Product Management';
}
```



Demo



Creating the App Component



Bootstrapping Our App Component



Load the root component (bootstrapping)

Host the application



Single Page Application (SPA)



`index.html` contains the main page for the application

This is often the only Web page of the application

Hence an Angular application is often called a Single Page Application (SPA)

Hosting the Application

index.html

```
<body>
  <pm-app>Loading App...</pm-app>
</body>
```

app.component.ts

```
import {Component} from 'angular2/core';

@Component({
  selector: 'pm-app',
  template: `
    <div><h1>{{pageTitle}}</h1>
      <div>My First Component</div>
    </div>

  `
})
export class AppComponent {
  pageTitle: string = 'Acme Product Management';
}
```



Angular 2 Application Startup

index.html

```
System.import('app/main');
```

```
<body>
  <pm-app>Loading App ...
</pm-app>
</body>
```

main.ts (bootstrapper)

```
import { bootstrap }
  from 'angular2/
    platform/browser';

import { AppComponent }
  from './app.component';

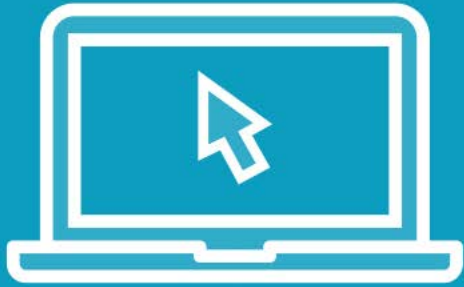
bootstrap(AppComponent);
```

app.component.ts

```
@Component({
  selector: 'pm-app',
  template: `
    <div>{{pageTitle}}</div>
  `
})
export class AppComponent
{ }
```



Demo



Bootstrapping the App Component



Component Checklist







Class -> Code

Decorator -> Metadata

Import what we need



Component Checklist: Class



Clear name

- Use PascalCasing
- Append "Component" to the name

export keyword

Data in properties

- Appropriate data type
- Appropriate default value
- camelCase with first letter lowercase

Logic in methods

- camelCase with first letter lowercase

Component Checklist: Metadata



Component decorator

- Prefix with @; Suffix with ()

selector: Component name in HTML

- Prefix for clarity

template: View's HTML

- Correct HTML syntax

Component Checklist: Import



Defines where to find the members that this component needs

`import keyword`

Member name

- Correct spelling/casing

Module path

- Enclose in quotes
- Correct spelling/casing

Summary



What is a Component?

Creating the Component Class

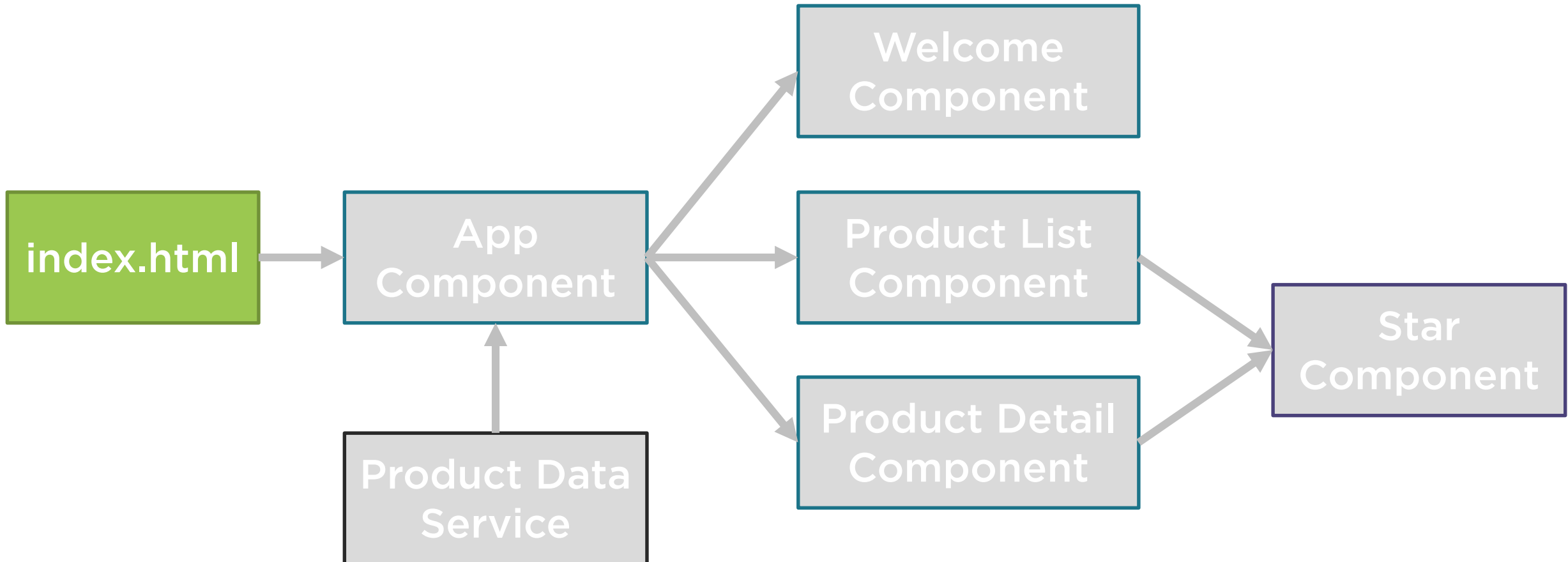
Defining the Metadata with a Decorator

Importing What We Need

Bootstrapping Our App Component



Application Architecture



Application Architecture

