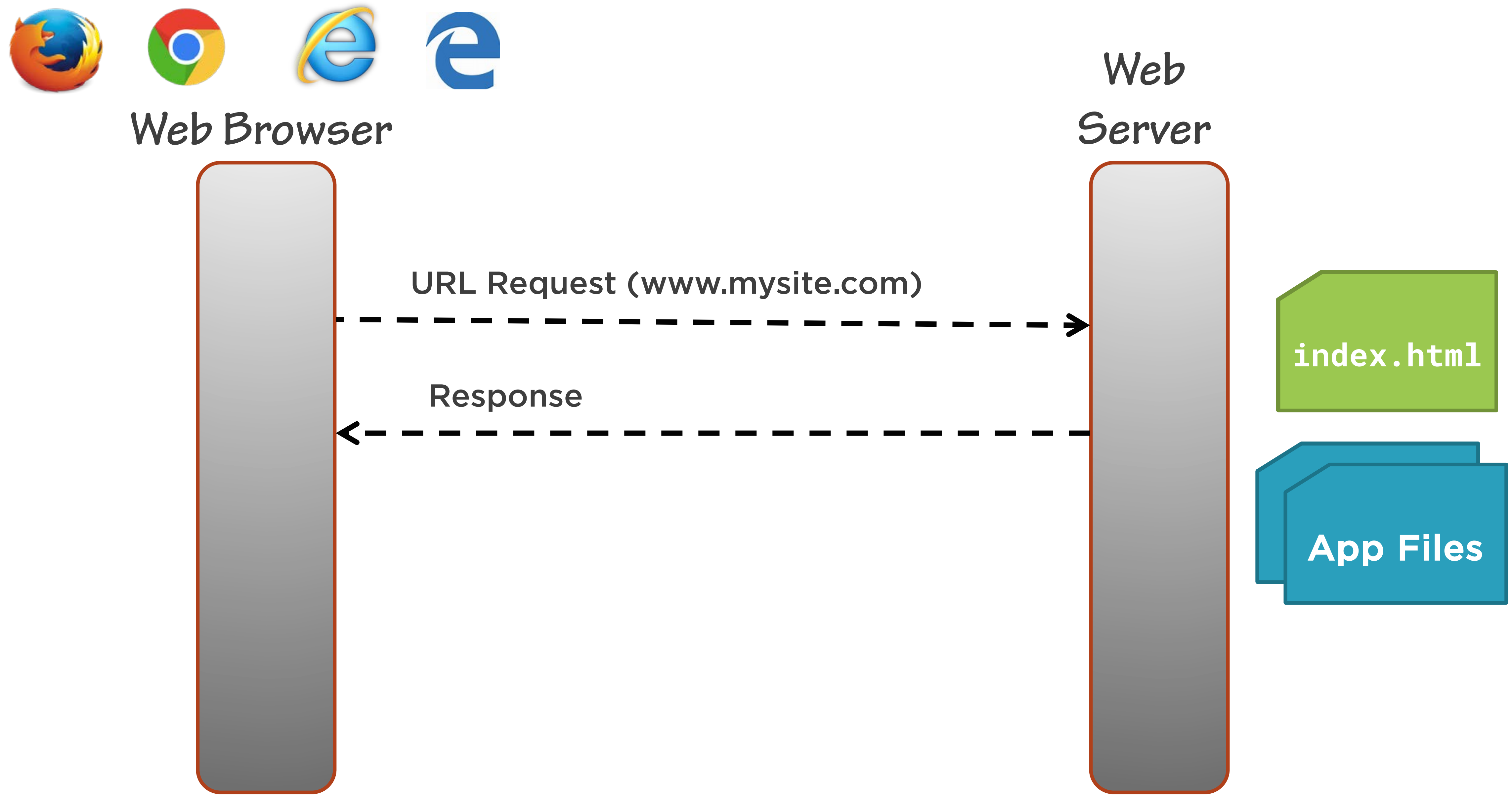
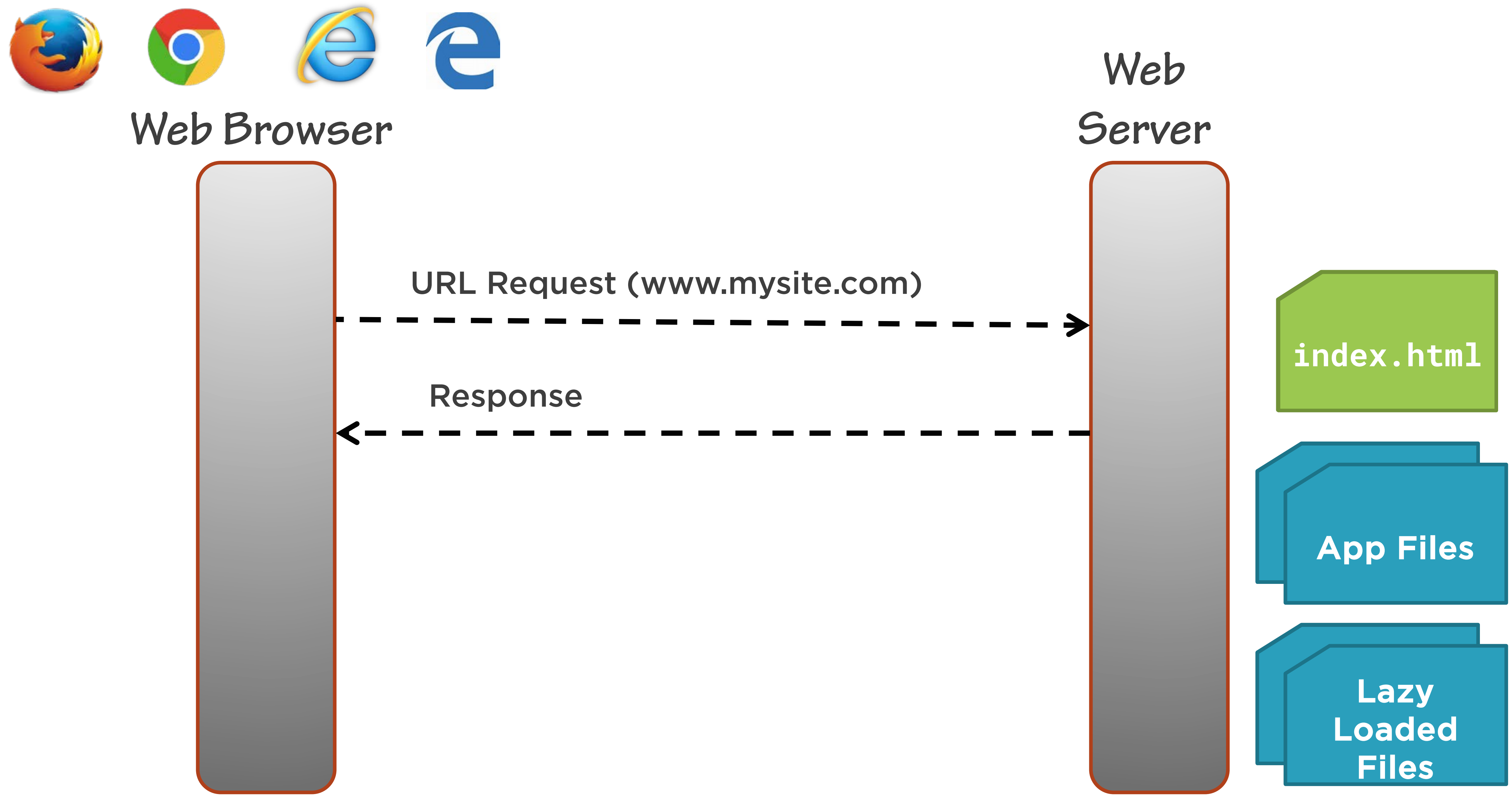
Lazy Loading

Deborah Kurata
CONSULTANT | SPEAKER | AUTHOR

@deborahkurata | blogs.msmvps.com/deborahk/


Web Browser
Web Server
URL Request (www.mysite.com)
Response
index.html
App Files


Web Browser
Web Server
URL Request (www.mysite.com)
Response
index.html
App Files
Lazy Loaded Files

Module Overview

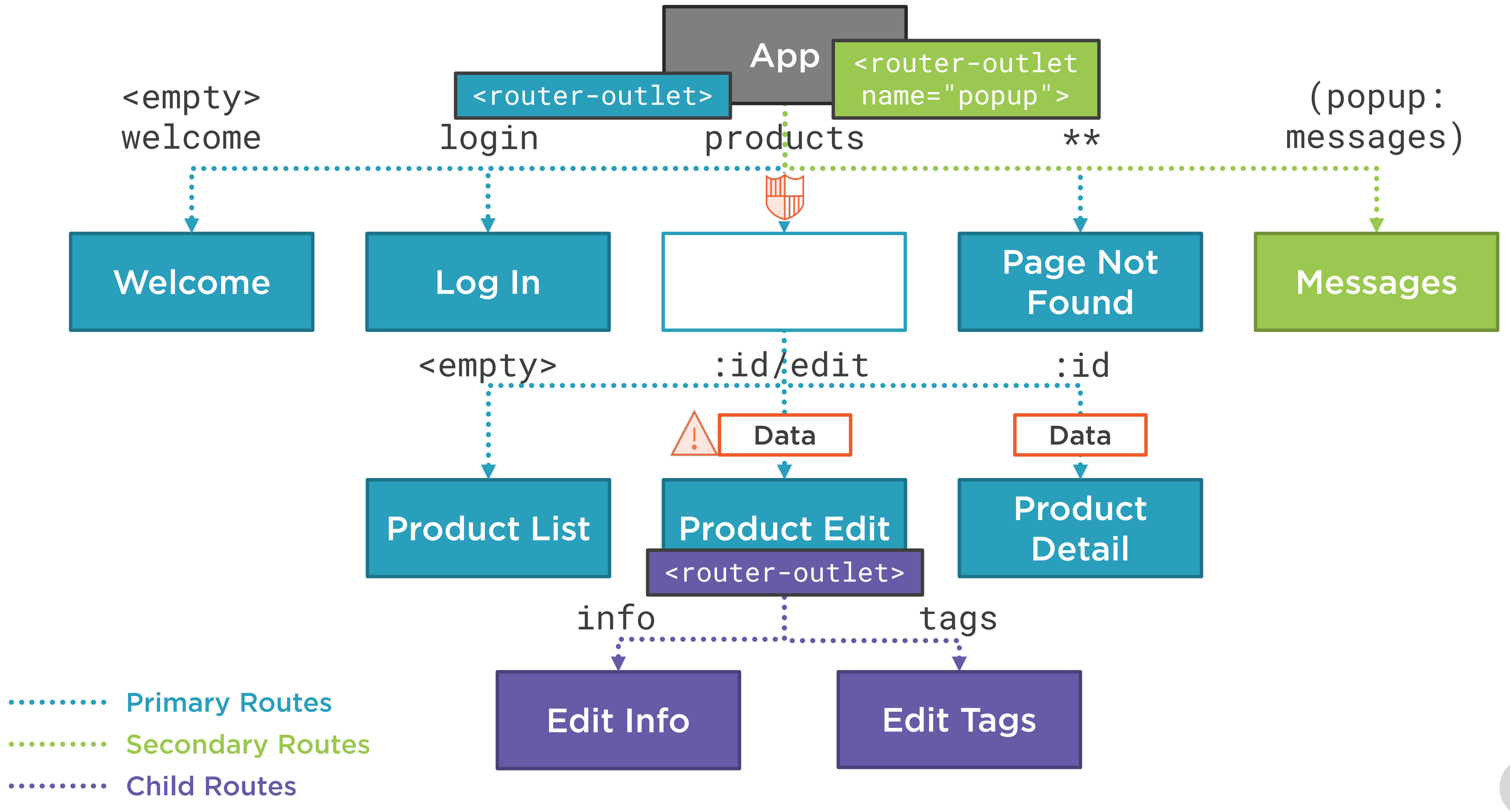
Preparing for Lazy Loading

Lazy Loading

canLoad Guard

Preloading Feature Modules

Custom Loading Strategies


App
<router-outlet>
<router-outlet name="popup">
<empty> welcome
login
products
**
(popup: messages)
Welcome
Log In
Page Not Found
Messages
<empty>
:id/edit
:id
Data
Data
Product List
Product Edit
Product Detail
<router-outlet>
info
tags
Edit Info
Edit Tags
Primary Routes
Secondary Routes
Child Routes

Preparing for Lazy Loading

- Use a feature module
- Routes grouped under a single parent
- Not imported in another module

Lazy Loading

app-routing.module.ts

```
RouterModule.forRoot([
  ...
  {
    path: 'products',
    loadChildren: 'app/products/product.module#ProductModule'
  },
  ...
])
```

canLoad Guard

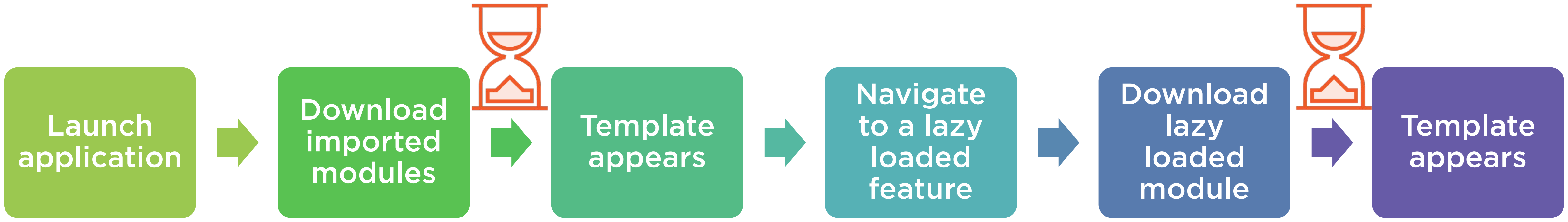
Checks criteria before loading an asynchronous route

Commonly used to:

- Prevent loading a route if a user cannot access it

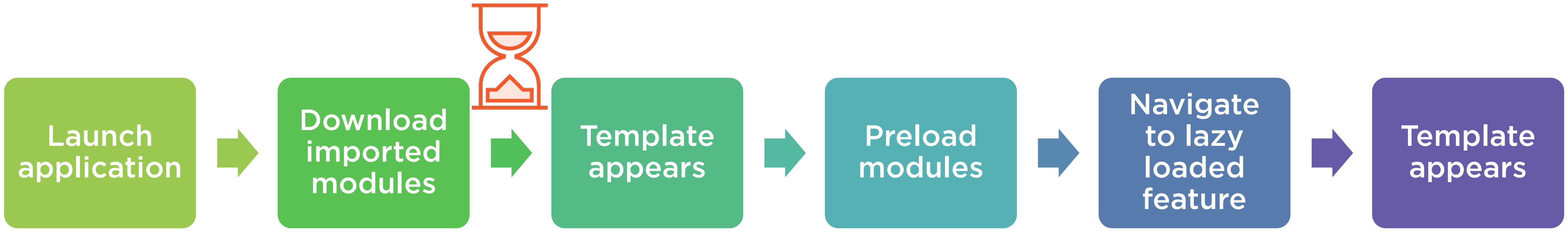
Preloading Feature Modules

Lazy Loading



Launch application
Download imported modules
Template appears
Navigate to a lazy loaded feature
Download lazy loaded module
Template appears

Preloading (Eager Lazy Loading)



Preload Strategies

No preloading

Preload all

Custom

Preload All Strategy

app-routing.module.ts

```
import { RouterModule, PreloadAllModules } from '@angular/router';
...

RouterModule.forRoot([
 { path: 'welcome', component: WelcomeComponent },
 {
   path: 'products',
   loadChildren: 'app/products/product.module#ProductModule'
 },
 ...
], { preloadingStrategy: PreloadAllModules })
```

canLoad guard blocks preloading

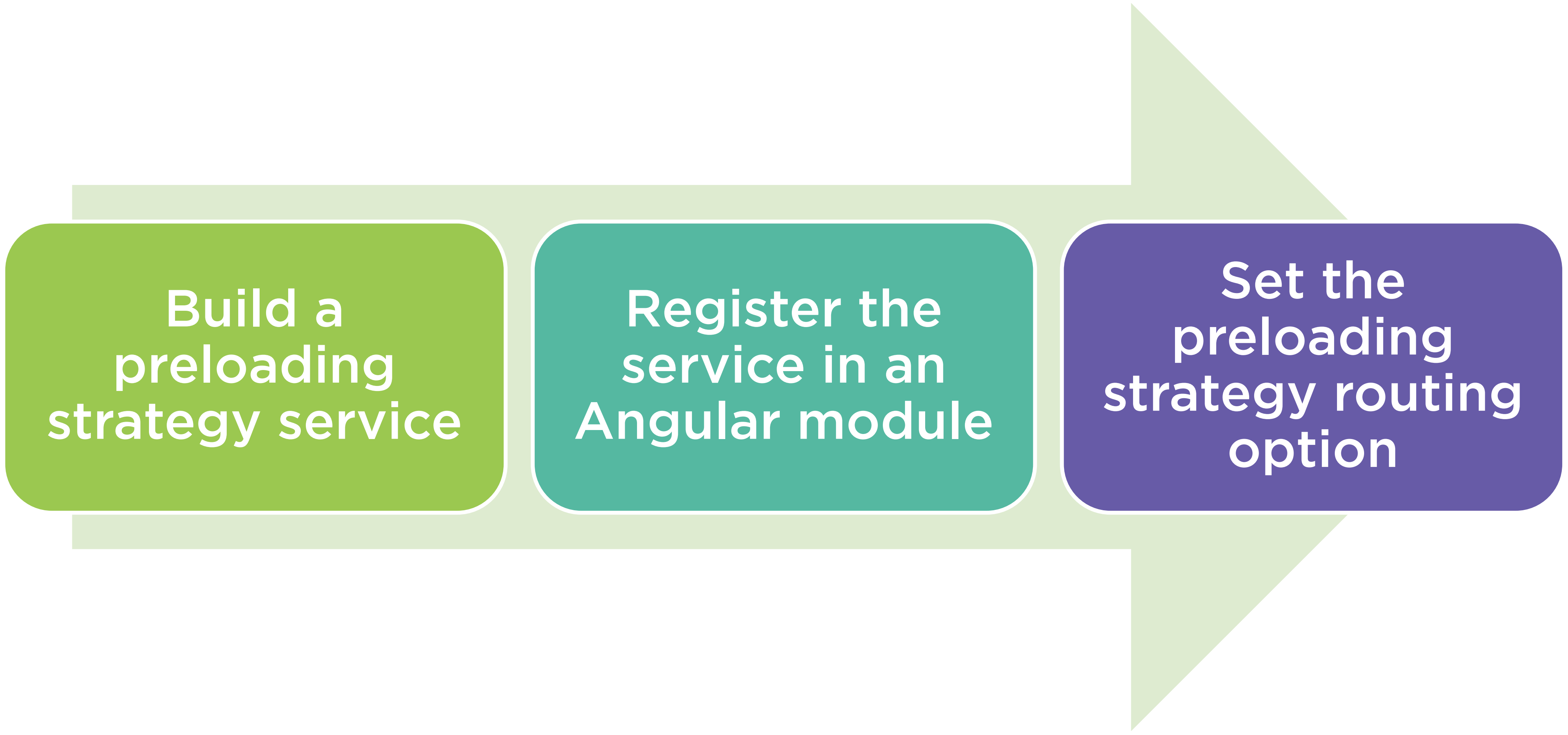
Preload Strategies

No preloading

Preload all

Custom

Custom Preloading Strategy


Build a preloading strategy service
Register the service in an Angular module
Set the preloading strategy routing option

Building a Preload Strategy Service

selective-strategy.service.ts

```
import { Injectable } from '@angular/core';
import { PreloadingStrategy } from '@angular/router';
import { Observable } from '@angular/router';

@Injectable()
export class SelectiveStrategy implements PreloadingStrategy {

    preload(route: Route, load: Function): Observable<any> {
      ...
    }
}
```

Registering and Enabling a Strategy

app-routing.module.ts

```
...
import { SelectiveStrategy } from './selective-strategy.service';

@NgModule({
  imports: [
      RouterModule.forRoot([
          { path: 'welcome', component: WelcomeComponent },
          {
             path: 'products',
             loadChildren: 'app/products/product.module#ProductModule'
           }, ...
      ], { preloadingStrategy: SelectiveStrategy }),
  ],
  providers: [ SelectiveStrategy, ... ]
})
export class AppModule { }
```

Lazy Loading Checklist: Preparing

Use a feature module

Group routes under a single parent

Don't import feature module

Lazy Loading Checklist: Configuring

Add the `loadChildren` property

Set it to:

- The full path to the module
- A hash
- And the module's class name

```
{
  path: 'products',
  loadChildren: 'app/products/product.module#ProductModule'
}
```

Lazy Loading Checklist: Preloading

Preload all modules with `PreloadAllModules`

Or build a custom preloading strategy service

```
RouterModule.forRoot([
 {
   path: 'products',
   loadChildren: 'app/products/product.module#ProductModule'
 },
 ...
], { preloadingStrategy: PreloadAllModules })
```

Summary

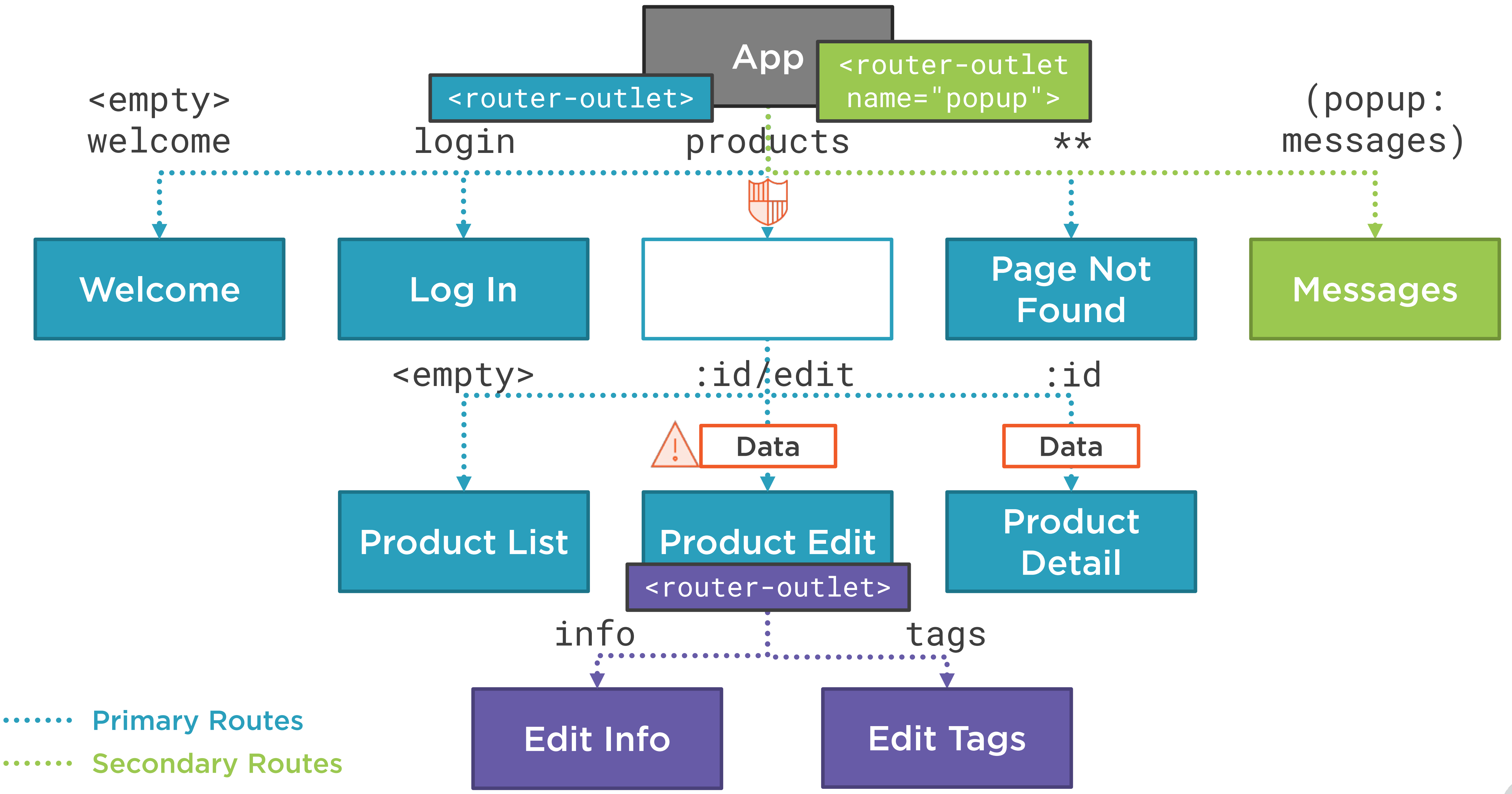
Preparing for Lazy Loading

Lazy Loading

canLoad Guard

Preloading Feature Modules

Custom Loading Strategies



········· Primary Routes

········· Secondary Routes

········· Child Routes