Desenvolupament Web amb

# Angular – (MEAN stack 2)

## Mòduls

Raúl García Lepe raulgl@gmail.es

# Mòduls

- Ens permeten desacoblar les funcionalitats de la aplicació.
- S'utilitza la instrucció: **ng generate module *nom_module***
- Si ho volem amb sistema de enrutament propi o creen amb el modificador: **--routing**
- Una vegada creat s'ha de donar visibilitat en altres mòduls fent la importación del mòdul.

# Exercici 1

- Crear nova aplicació amb el nom **angular-modules**
- Modificar l'inici del servidor per a incorporar les opcions Ahead Of Time i obertura immediata del navegador.
- Incorporar les llibreries externes que necessitis (BootStrap, Semantics-UI)
- "Netejar" la plantilla app.component.html deixant només les etiquetes *router-outlet*

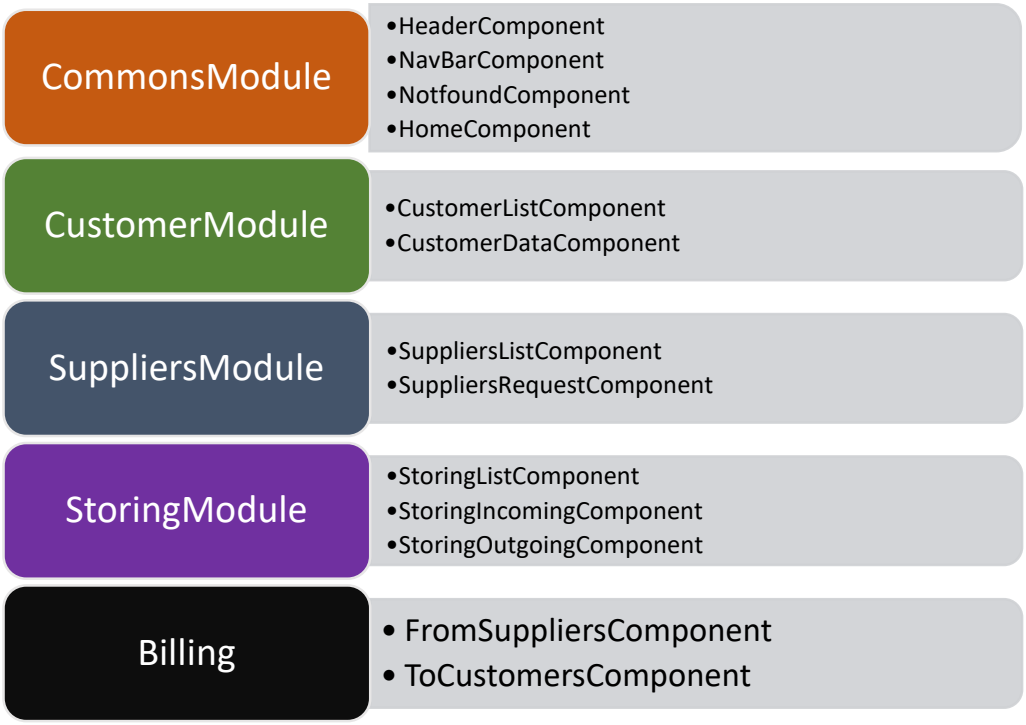
# Exercici 1(cont)

- Crear un nou mòdul amb el nom **commons** que ens servirà per agrupar tots els components comuns de la aplicació: headers, footers, barra de navegació,...
- Incorporar un nou component al mòdul commons amb el nom de **header** que sigui visible des de altres mòduls (modificador --export).
- Modificar el header.component.html afegint estils.
  - Amb *bootstrap* pots usar container-fluid i un element div amb la classe jumbotron. Dins portarà un element h1 amb el títol 'Cabecera de la aplicación'
- Importar el mòdul CommonsModule al AppModule per usar la classe header des dels components d'aquest mòdul
- Modificar el app.component.html incorporant-ne la etiqueta app-header.
- Mostrar el resultat

# Cabecera de la aplicación

# Una aplicació complerta

- Desenvoluparem una aplicació per la gestió d'una petita empresa.
- Consta d'una pàgina principal i quatre seccions: clients, proveïdors, magatzem i facturació. Cadascuna de les seccions seran mòduls amb els seus propis components que generaran vistes:



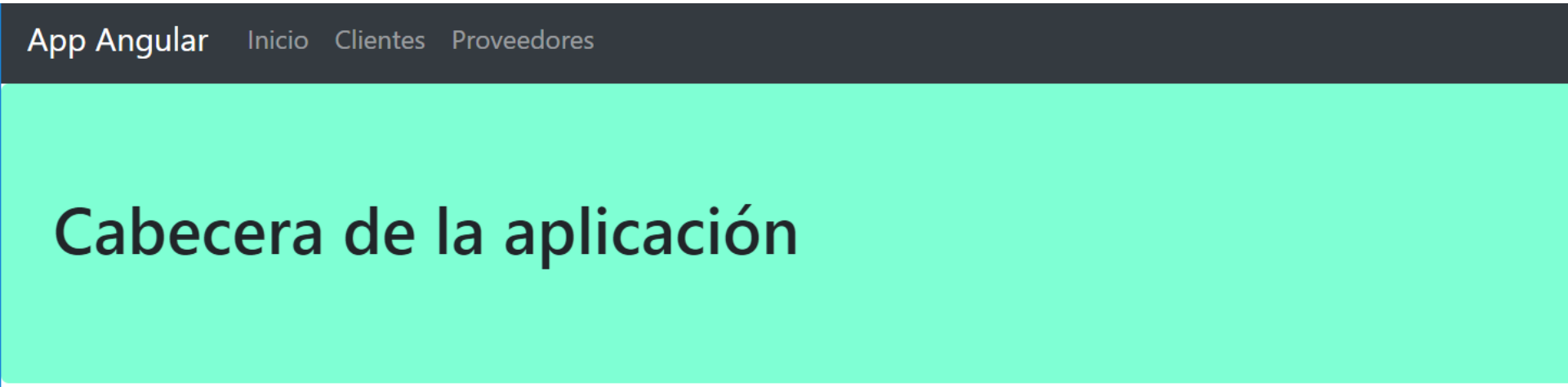
# Mòduls

- Donem d'alta els mòduls de les seccions a la nostra aplicació:
  - ng g m customers --routing true
  - ng g m suppliers --routing true
  - ng g m storing --routing true
  - ng g m billing --routing true
- Dins del CustomerModule crearem dos components:
  - Veure el llistast de clients: ng g c customers/customerList --export
  - Veure les dades d'un client: ng g c customers/customerData --export
- Dins del SuppliersModule:
  - Veure llistat de proveïdors: ng g c suppliers/suppliersList --export
  - Fer una petició a un proveïdor: ng g c suppliers/supplierRequest --export
- Storing:
  - Llistat d'articles: ng g c storing/storingList --export
  - Ficar article: ng g c storing/storingIncoming --export
  - Treure article: ng g c storing/storingOutgoing --export
- Billing:
  - Facturació proveïdors: ng g c billing/fromSuppliers --export
  - Facturació clients: ng g c billing/toCustomers --export

# Mòduls

- **CommonModule: 3 components**
  - Barra de navegació (navbar)
  - Pàgina 404 - no trobat - (notfound)
  - Pàgina d'inici (home)

- La Pàgina 404 porta els missatges de pàgina no encontrada
- La Barra de navegació la adaptarem de BootStrap
- La pàgina de inici ens indicarà que ens trobem a la primera pàgina


App Angular
Inicio
Clientes
Proveedores
Cabecera de la aplicación

# Routing

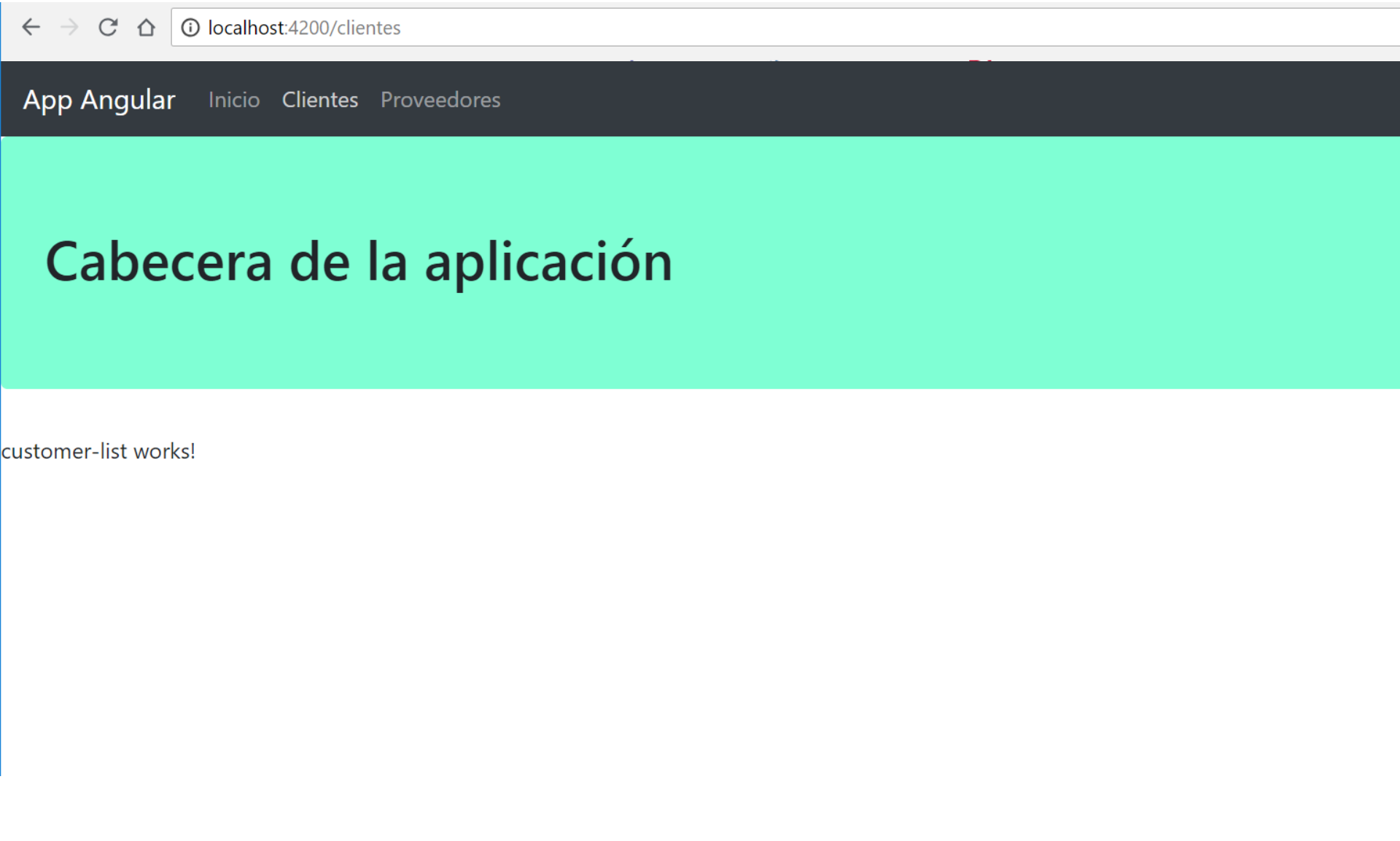
- El enrutament consisteix en relacionar les urls amb els components que es carreguen. Els enrutaments poden ser:
  - Globals: per arribar a totes o la majoria de les seccions (Root Routing)
  - De mòduls: mòduls específics per accedir les vistes de components d'un mòdul

- Root Routing: Es troba al arxiu **app-routing.module.ts** relacionat amb el AppComponent. Aquest arxiu importa el NgModule per a crear el decorador, després s'importen:
  - RouterModule: mòdul creat per Angular per fer funcionar l'enrutament.
  - Routes: Matriu de tipus personalitzat que es defineix al propi Angular i es diu Route. Dins d'aquesta matriu es crearan les rutes a las vistes de la aplicació.
- El decorador no te propietat *declarations*. Només importa el mòdul RouterModule amb el mètode **forRoot(routes)** que indica l'enrutament al root. Després s'exporta.
- L'exportació es fa a la vista del AppComponent: a la **directiva router.outlet.** NO ÉS UNA ETIQUETA. Les directives amplien les funcionalitats de les etiquetes, aquesta rebre l'enrutament i ho posa en funcionament.

# Exercici 2

- Afegir les rutes a la pàgina principal de la aplicació:
  - Mitjançant la matriu Routes informem del path als diferents components:
    - path: ’’, component: HomeComponent
    - path: ’clientes’, component: CustomersListComponent
    - path: ’proveedores’, component: SuppliersListComponent
    - path: ‘**’, component: NotFoundComponent

- Modificar la barra de navegació per a enllaçar les rutes:
  - Substituim la propietat href por routerLink (no més funciona a Angular) i s’afegeix el valor de la propietat path de la matriu routes.


localhost:4200/clientes
App Angular
Inicio
Clientes
Proveedores
Cabecera de la aplicación
customer-list works!

# Lazy Load

- La càrrega mandrosa, diferida o *lazy load* consisteix que només es carregui la vista d'inici de l'aplicació, o les vistes que sapiguem, per experiència, que són més consultades

- La resta de les vistes no es carregaran en el navegador del client tret que aquest les sol·liciti expressament (prement un enllaç, per exemple). D'aquesta manera, les vistes que no són invocades pel client no es carreguen, i no consumeixen recursos de la connexió

- Molt utilitzades a SPA's i pensant en dispositius que puguin tenir connexions més lentes

# Lazy Load

- Requereix que els components que no es vagin a carregar immediatament s'hagin creat amb el flag --routing
- Des de lazy load s'accedeix a la carrega del mòdul però no a un component específic., per tant es requereix que tingui enrutador propi.
- Els enrutadors dels mòduls que no siguin root fan l'import amb el mètode **forChild:**

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

const routes: Routes = [];

@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class StoringRoutingModule { }
```

# Exercici 3

- Volem afegir una llista desplegable al menú d'Inici amb l'element 'Almacén' i els tres subelements d'aquest: Listado, Entradas, Salidas. Aquests elements es carregaran mode lazy load
- Per fer això requerim modificar la navbar afegint l'element i els subelements com a llista desplegable:

```
<li class='nav-item dropdown'>
  <a class= 'nav-link dropdown-toggle'href="#" id= "dropdown_almacen"
  data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">Almacén</a>
  <div class="dropdown-menu" aria-labelledby="dropdown_almacen">
    <a class="dropdown-item" routerLink="almacen">Listado</a>
    <a class="dropdown-item" routerLink="almacen/entradas">Entradas</a>
    <a class="dropdown-item" routerLink="almacen/salidas">Salidas</a>
  </div>
</li>
```

# Exercici 3(cont 1)

- Modifiquem l'element de routing del root afegint un nou element per a la càrrega lazy load:

```
const routes: Routes = [
  {path: '',
   component: HomeComponent
  },
  {path: 'clientes',
   component: CustomerListComponent
  },
  {path: 'proveedores',
  component: SuppliersListComponent
  },
  {path: 'almacen', loadChildren: './storing/storing.module#StoringModule'
  },
  {path: '**',
  component: NotfoundComponent
  },
```

# Exercici 3(cont 2)

- Modifiquem l'enrutador del mòdul lazy afegint el path als elements:

```
const routes: Routes = [
  { path: '', component: StoringListComponent},
  { path: 'entradas', component: StoringIncomingComponent},
  { path: 'salidas', component: StoringOutgoingComponent},

];
```


localhost:4200/almacen/salidas
App Angular
Inicio
Clientes
Proveedores
Almacén
Listado
Entradas
Salidas
Cabecera de la apli
storing-outgoing works!

# Rutes amb paràmetres

- Són l'enllaç entre les dades persistides i la presentació a les vistes en funció de las dades demanades.
- No més carreguem una vista amb dades diferents en funció d'un paràmetre. EL COMPONENT I LA SEVA VISTA SEMPRE SERAN ELS MATEIXOS.

# Exercici 4

- Incorporar a la vista de clients un llistat amb els clients registrats. Cada client tindrà un enllaç als seus detalls:

1. Creem un nou path a la vista de dades del client amb l'identificador de cada client (customer-list.component.html):
2. Modifiquem el component customer-data per rebre el paràmetre i asignar-ho a una variable del component
3. Modifiquem la vista customer-data per a incorporar les dades que ens demanen per paràmetre
4. Modifiquem el enrutador de root per a incorporar la ruta al detall de cada client

# Exercici 4(cont)

1. Creem un nou path a la vista de dades del client amb l'identificador de cada client (customer-list.component.html):
   - Afegim l'enllaç al llistat de clients al AppRoutingModule:

```
{
  path: 'ver_cliente/:id',
  component: CustomerDataComponent
},
```

   - Incorporem el llistat d'enllaços a la vista de dades de client:

```
<div class='container'>
  <h1>Listado de clientes</h1>
</div>

<div class='container'><hr></div>

<div class='container'>
  <ul>
    <li><a routerLink='../ver_cliente/108'>Cliente 108</a></li>
    <li><a routerLink='../ver_cliente/227'>Cliente 227</a></li>
    <li><a routerLink='../ver_cliente/305'>Cliente 305</a></li>
    <li><a routerLink='../ver_cliente/543'>Cliente 543</a></li>
  </ul>
</div>
```

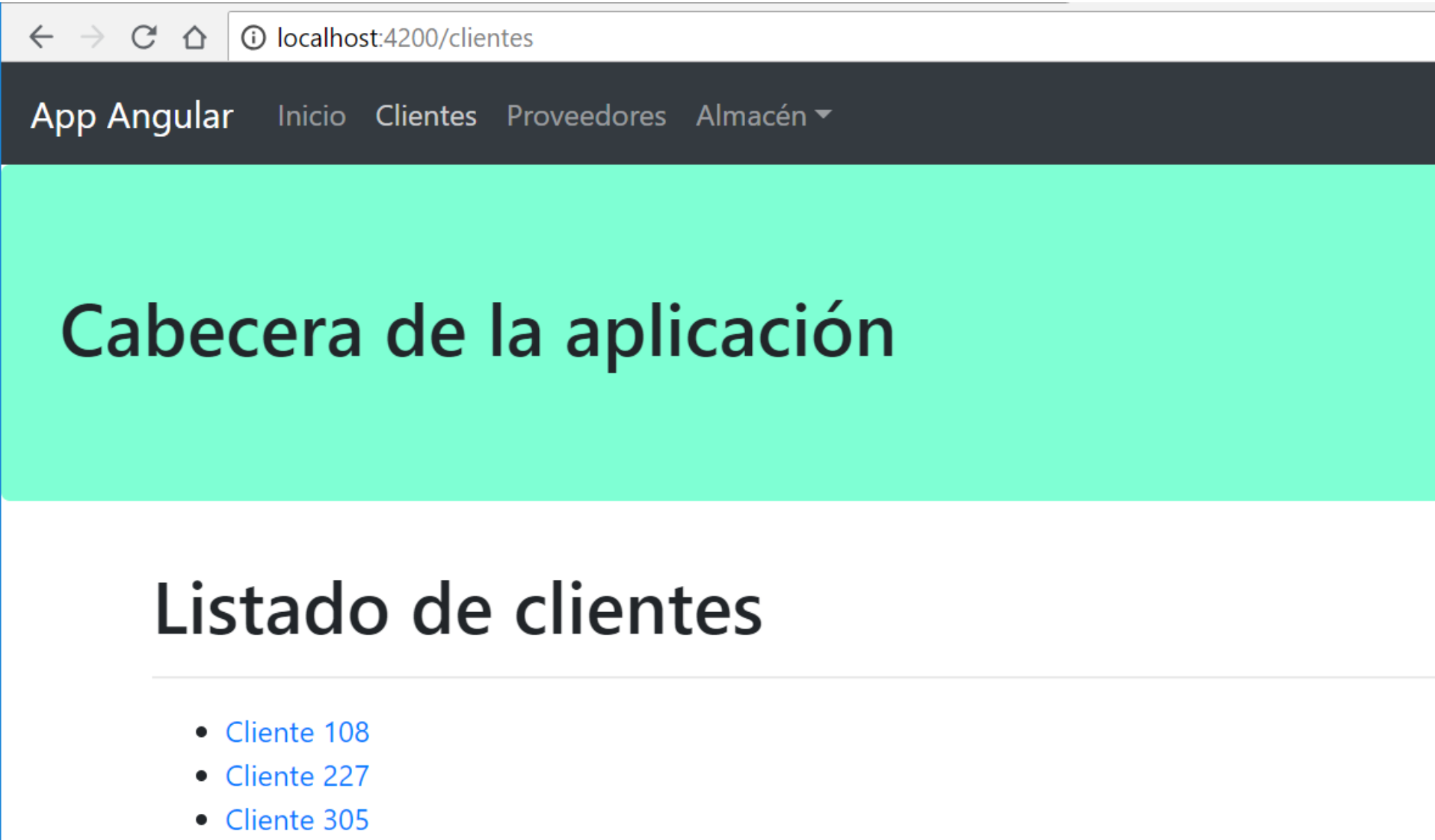
# Exercici 4(cont)

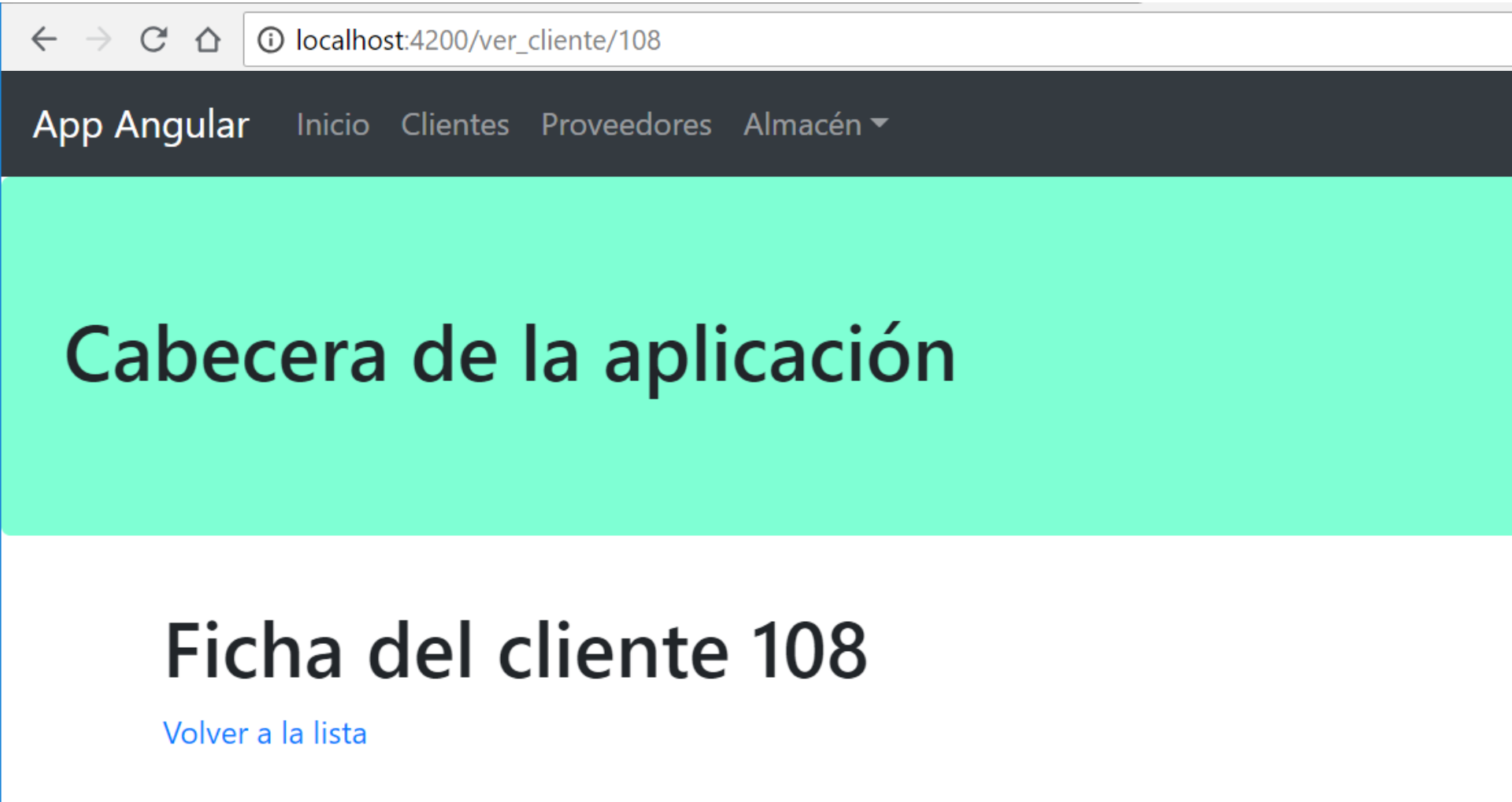
2. Modifiquem el component customer-data per rebre el paràmetre i asignar-ho a una variable del component

```
export class CustomerDataComponent implements OnInit {

  identificador: any;

  constructor(private customer: ActivatedRoute) {}

  ngOnInit() {
    this.identificador = this.customer.snapshot.params['id'];
  }
```

# Exercici 4(cont)

3. Modifiquem la vista customer-data per a incorporar les dades que ens demanen per paràmetre (en aquest cas no més tenim l'identificador)
4. Afegir un enllaç de retorn al llistat de clients des del seu detall.


localhost:4200/clientes
App Angular
Inicio
Clientes
Proveedores
Almacén
Cabecera de la aplicación
Listado de clientes
Cliente 108
Cliente 227
Cliente 305

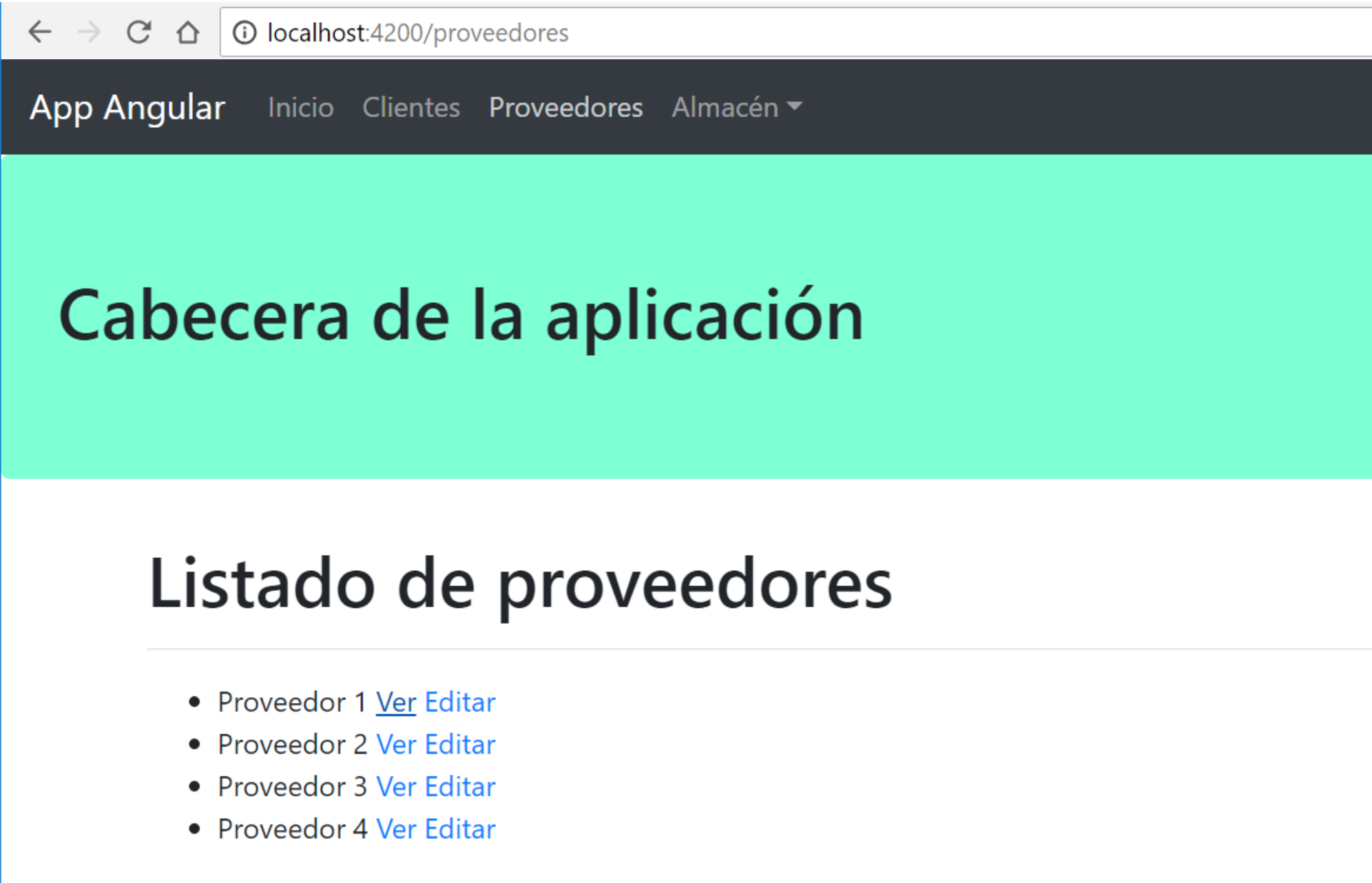

localhost:4200/ver_cliente/108
App Angular
Inicio
Clientes
Proveedores
Almacén
Cabecera de la aplicación
Ficha del cliente 108
Volver a la lista

# Exercici 5

- Modificar a lazy la càrrega dels proveïdors i mostrar un llistat de proveïdors amb un enllaç als detalls de cadascú:
  - Modificar la ruta de l'AppComponent a càrrega diferida (s'ha d'eliminar l'import del mòdul i es canvia el path component per loadchildren: *'ruta';*
  - Incorporar al html del suppliers-list el llistat de enllaços als proveïdors
  - Modificar el suppliers-router per a incorporar dos *path*: un **''** enllaçarà el component llistat i un ':id' enllaçarà al component request.
  - Modificar el RequestComponent incorporant la declaración de l'identificador i la captura del paràmetre.
  - Modificar el Request HTML per a incorporar un texte amb l'identificador i un enllaç per tornar al llistat.

# Exercici 6

- Crear una nova vista per l'edició de les dades del proveïdor:
  - Crear nou component amb el flag export
  - Modificar el suppliers-router per a incorporar dues rutes amb el paràmetre (ver_proveedor, editar_proveedor)
  - Mateixos passes que abans.


localhost:4200/proveedores
App Angular
Inicio
Clientes
Proveedores
Almacén
Cabecera de la aplicación
Listado de proveedores
Proveedor 1 Ver Editar
Proveedor 2 Ver Editar
Proveedor 3 Ver Editar
Proveedor 4 Ver Editar