Introduction to Web Development

Part 4: Demo Angular – SPA

arnaud.nauwynck@gmail.com

Course Esilv 2023

This document:
https://github.com/Arnaud-Nauwynck/presentations/web/angular-demos-4-angular-spa.pdf

Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

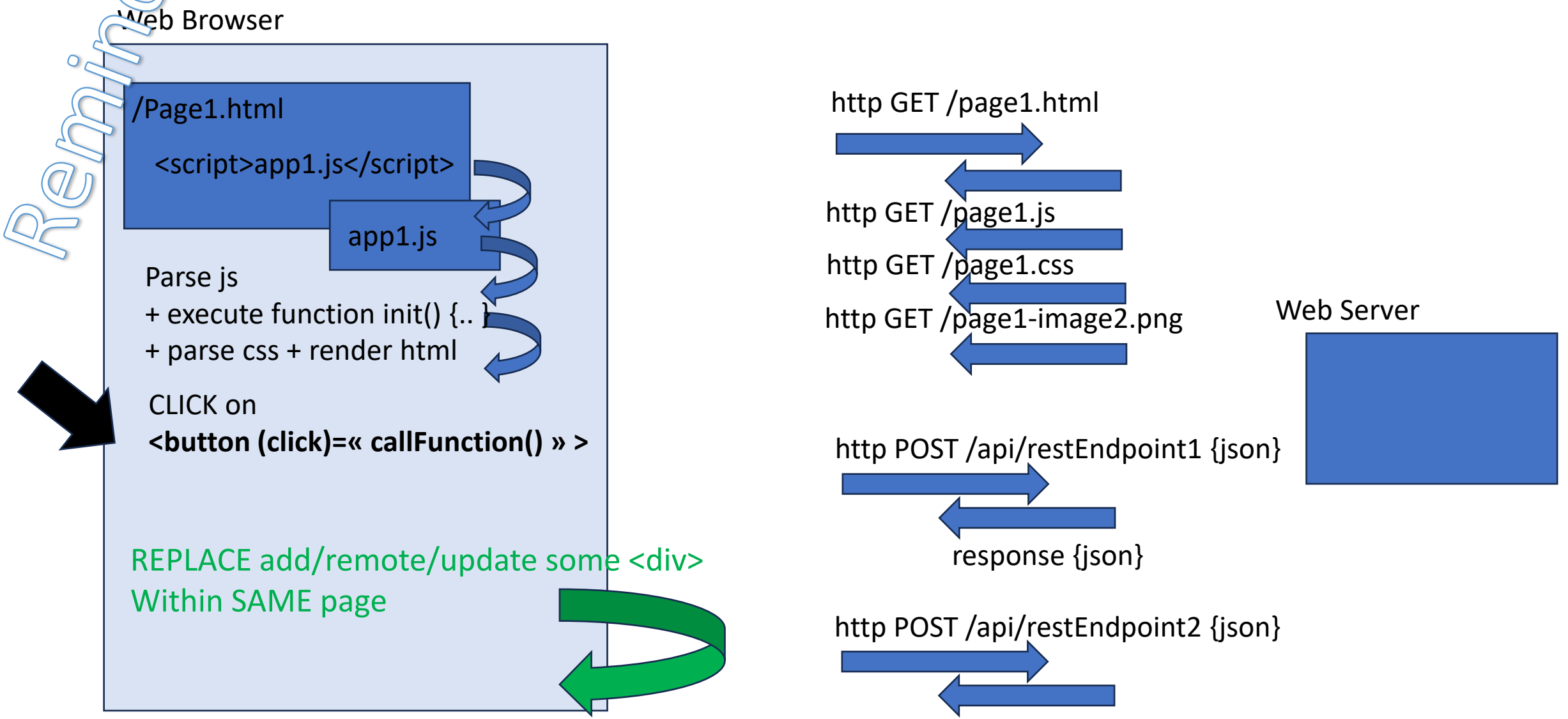
Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

SPA = Single Page Application
Single GET {html|js}, Multiple Rest Json requests

Reminder



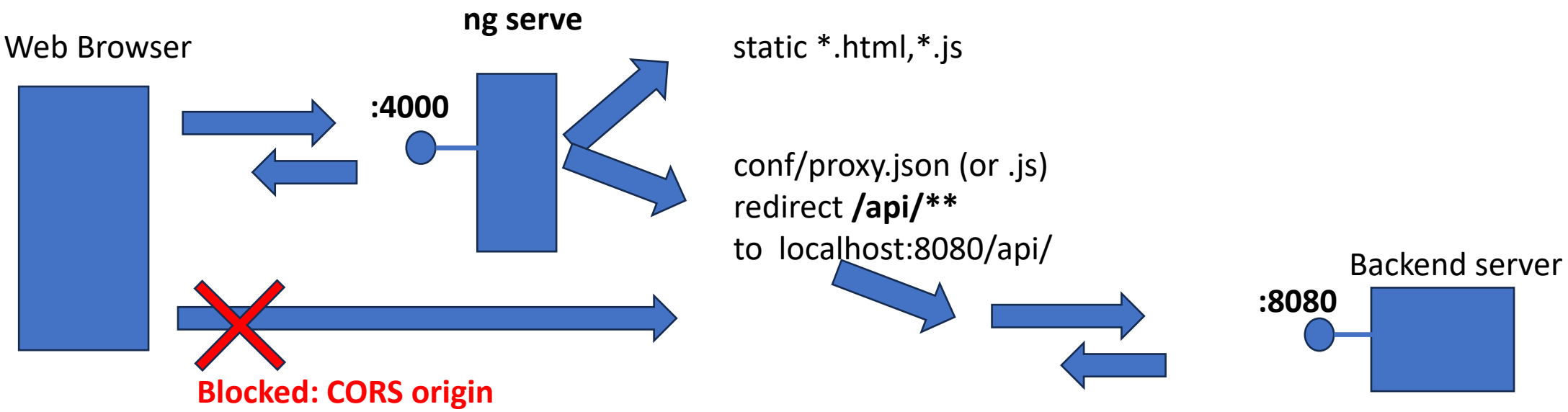
Reminder

Reminder FrontEnd / Rest API Backend CORS Origin problem !

Because …

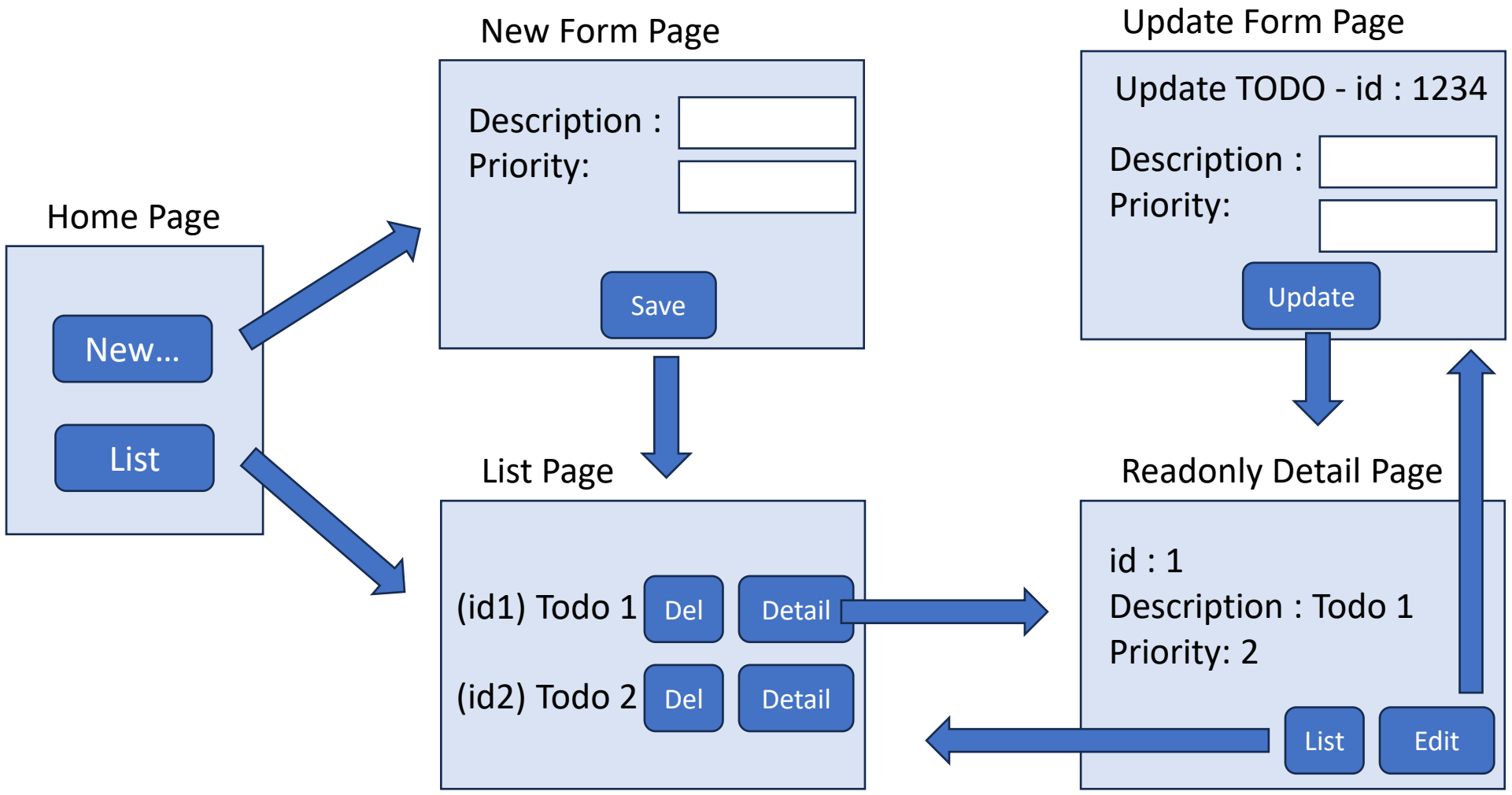
Web Browser blocks calls to « http://localhost:**8080**/api/** »

not comming from same « origin domain » as « http://localhost:**4000**/index.html » «app.js » !



Reminder

Reminder : Todo App Web Site Sketch



Reminder

Todo Web App Features Description … CRUD

C **Create** → Page 1: Simple Html Form with input fields
Click on « Save » button => save « Todo » object on server

R **Read All** → Page 2: Show list of « Todos »

Read By Id → page 3 for url « /todo/{id} »
Show detailed « Todo» page

U **Update** → Page 4: Editable page + « update » button

D **Delete** → « delete » button

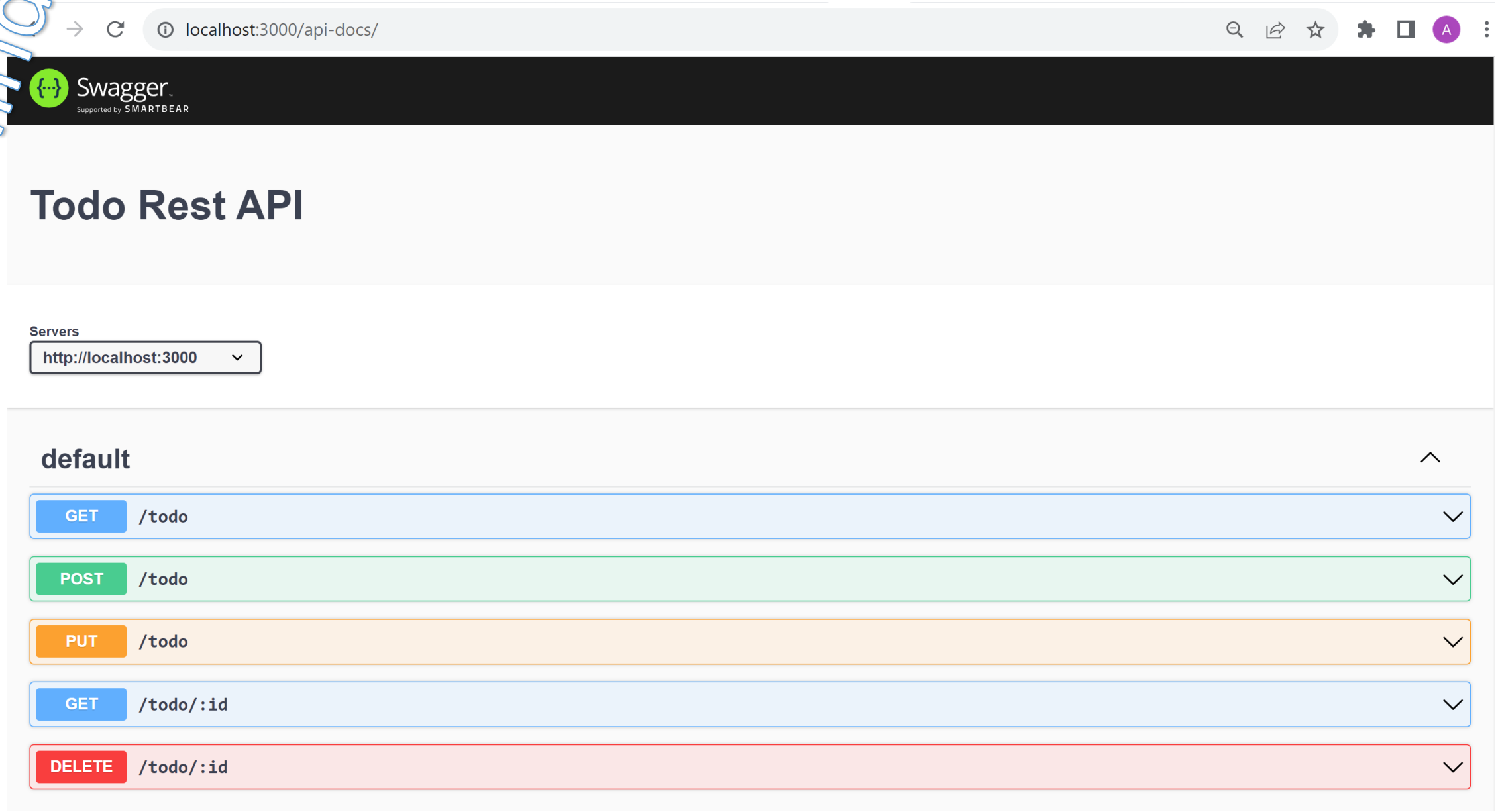
Reminder

Rest using NodeJs - Express

```
const express = require('express');
const app = express();

app.get('/', function (req, res) {
  res.send({ some: 'Hello Json Express' })
});

app.listen(3000);
```

Reminder

Swagger-UI : http://localhost:3000/api-docs/



Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

$ ng new

```
$ ng new demo4-todo-app
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? CSS
CREATE demo4-todo-app/angular.json (2740 bytes)
CREATE demo4-todo-app/package.json (1045 bytes)
CREATE demo4-todo-app/README.md (1066 bytes)
CREATE demo4-todo-app/tsconfig.json (901 bytes)
CREATE demo4-todo-app/.editorconfig (274 bytes)
CREATE demo4-todo-app/.gitignore (548 bytes)
CREATE demo4-todo-app/tsconfig.app.json (263 bytes)
CREATE demo4-todo-app/tsconfig.spec.json (273 bytes)
CREATE demo4-todo-app/.vscode/extensions.json (130 bytes)
CREATE demo4-todo-app/.vscode/launch.json (470 bytes)
CREATE demo4-todo-app/.vscode/tasks.json (938 bytes)
CREATE demo4-todo-app/src/main.ts (214 bytes)
CREATE demo4-todo-app/src/favicon.ico (948 bytes)
CREATE demo4-todo-app/src/index.html (298 bytes)
CREATE demo4-todo-app/src/styles.css (80 bytes)
CREATE demo4-todo-app/src/app/app-routing.module.ts (245 bytes)
CREATE demo4-todo-app/src/app/app.module.ts (393 bytes)
CREATE demo4-todo-app/src/app/app.component.html (23115 bytes)
CREATE demo4-todo-app/src/app/app.component.spec.ts (1015 bytes)
CREATE demo4-todo-app/src/app/app.component.ts (218 bytes)
CREATE demo4-todo-app/src/app/app.component.css (0 bytes)
CREATE demo4-todo-app/src/assets/.gitkeep (0 bytes)
/ Installing packages (npm)...
```

$ git add . ; git ci –m « init »

```
$ git st
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .editorconfig
        new file:   .gitignore
        new file:   .vscode/extensions.json
        new file:   .vscode/launch.json
        new file:   .vscode/tasks.json
        new file:   README.md
        new file:   angular.json
        new file:   package-lock.json
        new file:   package.json
        new file:   src/app/app-routing.module.ts
        new file:   src/app/app.component.css
        new file:   src/app/app.component.html
        new file:   src/app/app.component.spec.ts
        new file:   src/app/app.component.ts
        new file:   src/app/app.module.ts
        new file:   src/assets/.gitkeep
        new file:   src/favicon.ico
        new file:   src/index.html
        new file:   src/main.ts
        new file:   src/styles.css
        new file:   tsconfig.app.json
        new file:   tsconfig.json
        new file:   tsconfig.spec.json


arnaud@DesktopArnaud /cygdrive/c/arn/perso/cours/esilv/web/angular-demos/demo4-todo-app
$ git ci -m "init"
[main 0feeb6c] init
 23 files changed, 20784 insertions(+)
 create mode 100755 demo4-todo-app/.editorconfig
```

$ ng add @ng-bootstrap/ng-bootstrap

```
$ ng add @ng-bootstrap/ng-bootstrap
i Using package manager: npm
√ Found compatible package version: @ng-bootstrap/ng-bootstrap@15.1.0.
√ Package information loaded.

The package @ng-bootstrap/ng-bootstrap@15.1.0 will be installed and executed.
Would you like to proceed? Yes
√ Packages successfully installed.
UPDATE package.json (1185 bytes)
√ Packages installed successfully.
UPDATE src/app/app.module.ts (464 bytes)
UPDATE angular.json (2807 bytes)
UPDATE src/main.ts (259 bytes)
UPDATE tsconfig.app.json (294 bytes)
UPDATE tsconfig.spec.json (300 bytes)
```

```
$ git add .; git ci -m "add @ng-bootstrap";
[main 3186a7f] add @ng-bootstrap
 7 files changed, 206 insertions(+), 269 deletions(-)
```

Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)

- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

$ ng g c todo-list-page

```
$ ng g c todo-list-page
CREATE src/app/todo-list-page/todo-list-page.component.html (29 bytes)
CREATE src/app/todo-list-page/todo-list-page.component.spec.ts (603 bytes)
CREATE src/app/todo-list-page/todo-list-page.component.ts (232 bytes)
CREATE src/app/todo-list-page/todo-list-page.component.css (0 bytes)
UPDATE src/app/app.module.ts (574 bytes)
```

Basic HTML : <li *ngFor=« let todo of todos »>

Hardcoded Todo objects in *.ts

Defining model with Interface or Class

```
2   /**
3    * structural interface for Todo
4    * (example: for exchanging via json)
5    */
6   export interface Todo {
7     id: number;
8     description: string;
9     priority: number;
10  }
```

```
12  /**
13   * model class for Todo
14   */
15  export class TodoModel {
16    id: number;
17    description: string;
18    priority: number;
19
20    constructor(id: number, description: string, priority: number) {
21      this.id = id;
22      this.description = description;
23      this.priority = priority;
24    }
25  }
```

Todo list page

$ ng g c TodoListPage

todo-list-page.component.html

```
<H1>Todo list</H1>

<ul>
  <li *ngFor="let todo of todos; index as idx">
    ({{todo.id}}) {{todo.description}} {{todo.priority}}
  </li>
</ul>
```

todo-list-page.component.ts

```
import { Component } from '@angular/core';
import { Todo, TodoModel } from '../model/todo';

@Component({
  selector: 'app-todo-list-page',
  templateUrl: './todo-list-page.component.html'
})
export class TodoListPageComponent {

  todos: TodoModel[] = [
    new TodoModel(1, "learn angular", 1),
    new TodoModel(2, "learn typescript", 1),
    new TodoModel(3, "learn http", 1),
  ];

  todoInterfaces: Todo[] = [
    {id: 1, description: "learn angular", priority: 1},
    {id: 2, description: "learn typescript", priority: 1},
    {id: 3, description: "learn http", priority: 1},
  ];

}
```

Todo router page « /todos »

Add to router url: « /todos »

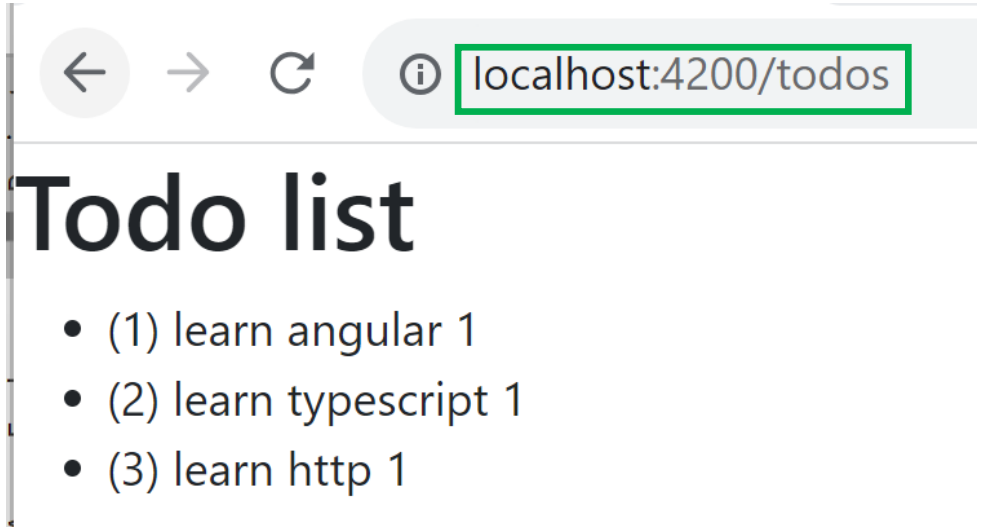
app-routing.module.ts

```ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { TodoListPageComponent } from './todo-list-page/todo-list-page.component';

const routes: Routes = [
  { path:'todos', component: TodoListPageComponent },
];
```

Use «router-outlet » in main page

app.component.html

```html
<router-outlet></router-outlet>
```



Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

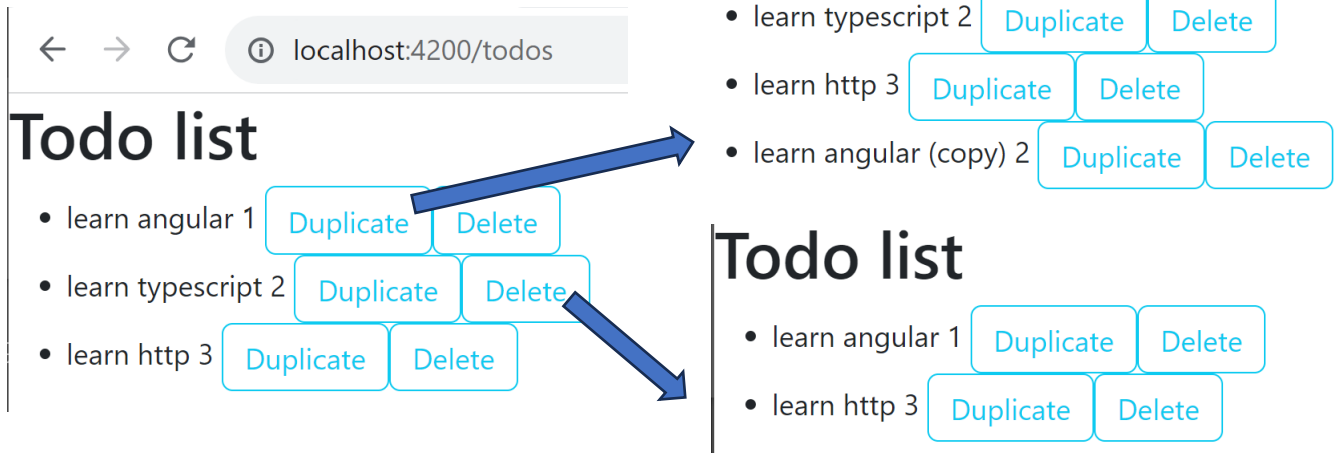
Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

Button to duplicate / delete Todo

```
7   <button type="button" class="btn btn-outline-info"
8           (click)="duplicateTodo(todo, idx)">Duplicate</button>
9
10  <button type="button" class="btn btn-outline-info"
11          (click)="deleteTodo(todo, idx)">Delete</button>
```

```
19  duplicateTodo(item: TodoModel, idx: number) {
20    let newId = this.todos.length + 1;
21    let dupItem = new TodoModel(newId, item.description + ' (copy)',
22                                item.priority+1);
23    this.todos.push(dupItem);
24  }
25
26  deleteTodo(item: TodoModel, idx: number) {
27    this.todos.splice(idx, 1);
28  }
```



Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

New Todo basic « Form »

```
$ ng g c TodoCreateSimple
CREATE src/app/todo-create-simple/todo-create-simple.component.html (33 bytes)
CREATE src/app/todo-create-simple/todo-create-simple.component.spec.ts (631 bytes)
CREATE src/app/todo-create-simple/todo-create-simple.component.ts (248 bytes)
CREATE src/app/todo-create-simple/todo-create-simple.component.css (0 bytes)
UPDATE src/app/app.module.ts (700 bytes)
```

todo-create-simple.component.html

```html
<H1>Create TODO</H1>

<div class="container">
  <div class="row">
    <label for="description" class="col-md-3">Description</label>
    <input id="description" class="col-md-5" type="text" [(ngModel)]="todo.description">
  </div>
  <div class="row">
    <label for="priority" class="col-md-3">Priority</label>
    <input id="priority" class="col-md-5" type="number" [(ngModel)]="todo.priority">
  </div>
  <div class="row">
    <div class="col-md-3"></div>
    <div class="col-md-3">
      <button type="button" class="btn btn-outline-info"
              (click)="onClickNew()">New</button>
    </div>
  </div>
</div>
```

todo-create-simple.component.ts

```ts
import { Component } from '@angular/core';

import { TodoModel } from '../model/todo';

@Component({
  selector: 'app-todo-create-simple',
  templateUrl: './todo-create-simple.component.html',
})
export class TodoCreateSimpleComponent {

  todo: TodoModel = new TodoModel(0, "", 1);

  onClickNew() {
    console.log('todo to add', this.todo);
  }
}
```

Add router « new-todo »

app-routing.module.ts

```
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes } from '@angular/router';
3  import { TodoListPageComponent } from './todo-list-page/todo-list-page.component';
4  import { TodoCreateSimpleComponent } from './todo-create-simple/todo-create-simple.component';
5
6  const routes: Routes = [
7    { path:'todos', component: TodoListPageComponent },
8    { path:'new-todo', component: TodoCreateSimpleComponent },
9  ];
```

Basic Form … [(ngModel)]=..
Need FormsModule
no validation yet

```
3  import { FormsModule } from '@angular/forms';

12 @NgModule({
13   declarations: [
14     AppComponent,
15     TodoListPageComponent,
16     TodoCreateSimpleComponent,
17     TodoUpdateSimpleComponent
18   ],
19   imports: [
20     BrowserModule,
21     AppRoutingModule,
22     NgbModule,
23     FormsModule
24   ],
```

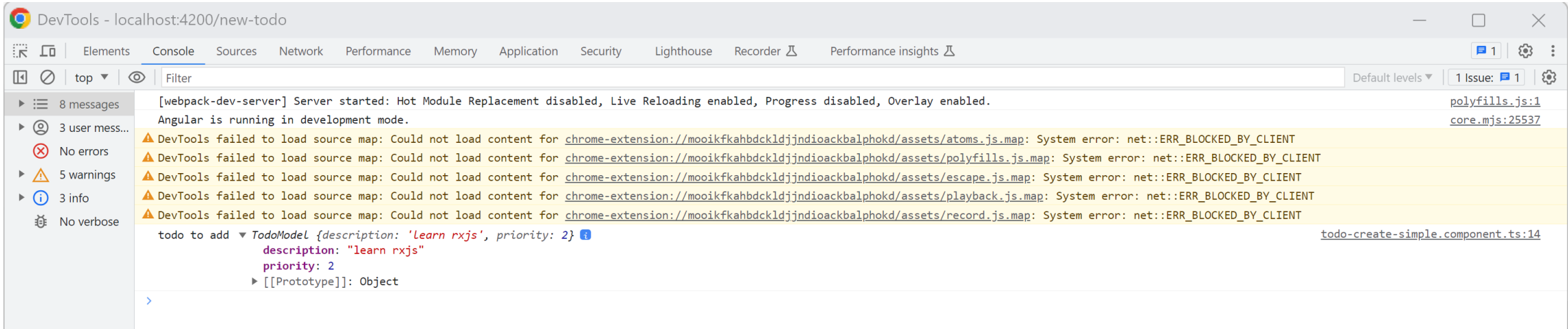
Create TODO

Description: learn rxjs

Priority: 2

New

Click + F12 Debugger Console



Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

add <A> or <button> « routerLink »

```
16  <div class="container">
17    <div class="row">
18      <label class="col-md-4">New using link or button</label>
19      <div class="col-md-2">
20        <A routerLink="/new-todo">New ...</A>
21      </div>
22      <div class="col-md-2">
23        <button type="button" class="btn btn-outline-info" routerLink="/new-todo">New ...</button>
24      </div>
25    </div>
26  </div>
```

url: /todos

New using link or button

New ...

New ...

To url: /new-todo

Add Button with contextual id « /todo/id » router.navigate([« /todo », id]);

```
13  <button type="button" class="btn btn-outline-info"
14          (click)="onClickOpenDetail(todo)">Detail...</button>
```

```
2   import { Router } from '@angular/router';

13    constructor(private router: Router) {}

35    onClickOpenDetail(item: TodoModel) {
36      this.router.navigate(['/todo', item.id]);
37    }
```

Declare url « /todo/:id » with path parameter

app-routing.module.ts

```
1  import { NgModule } from '@angular/core';
2  import { RouterModule, Routes } from '@angular/router';
3  import { TodoListPageComponent } from './todo-list-page/todo-list-page.component';
4  import { TodoCreateSimpleComponent } from './todo-create-simple/todo-create-simple.component';
5  import { TodoDetailPageComponent } from './todo-detail-page/todo-detail-page.component';
6
7  const routes: Routes = [
8    { path:'todos', component: TodoListPageComponent },
9    { path:'new-todo', component: TodoCreateSimpleComponent },
10   { path:'todo/:id', component: TodoDetailPageComponent },
11 ];
```

Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

Todo Detail page for « /todo/:id »

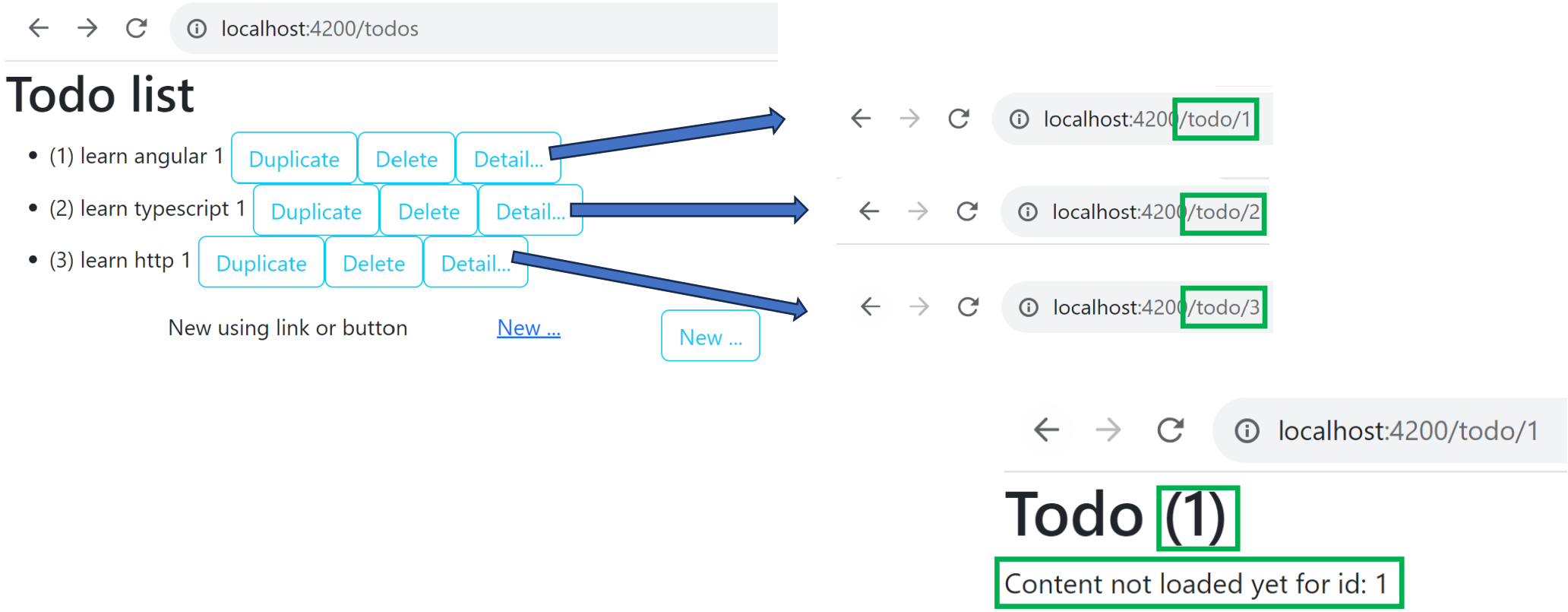
todo-detail-page.component.html

```
<H1>Todo ({{id}}) </H1>

<div *ngIf="todo" class="container">
  <div class="row">
    <label class="col-md-3">Description</label>
    <span class="col-md-5">{{todo.description}}</span>
  </div>
  <div class="row">
    <label class="col-md-3">Priority</label>
    <span class="col-md-5">{{todo.priority}}</span>
  </div>
</div>

<div *ngIf="!todo">
  Content not loaded yet for id: {{id}}
</div>
```

todo-detail-page.component.ts

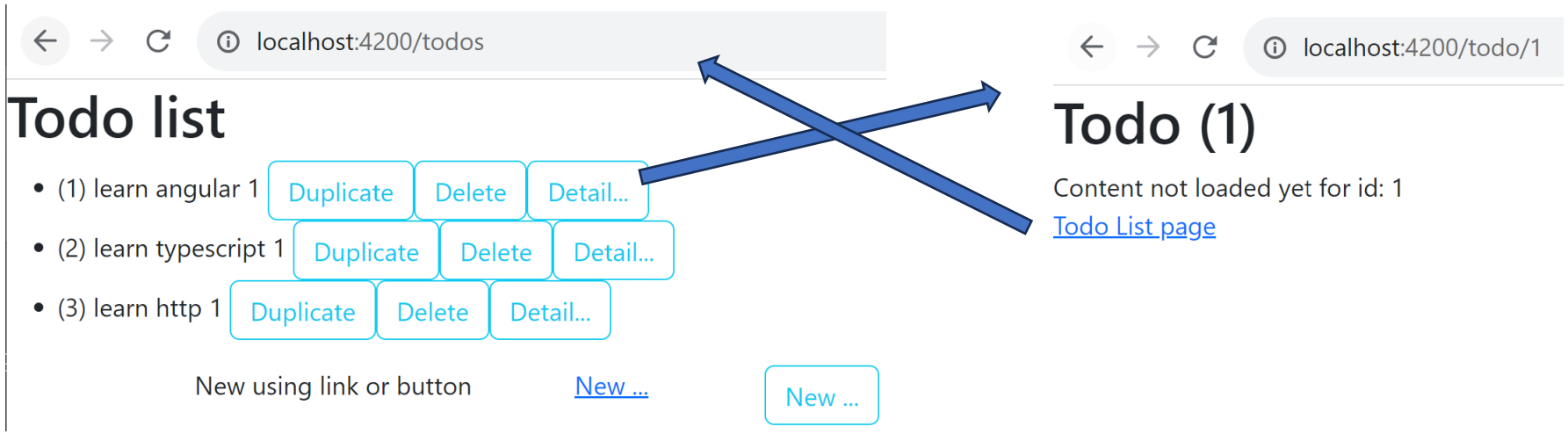
```
import { Component } from '@angular/core';
import { ActivatedRoute } from '@angular/router';

import { TodoModel } from '../model/todo';

@Component({
  selector: 'app-todo-detail-page',
  templateUrl: './todo-detail-page.component.html'
})
export class TodoDetailPageComponent {

  id: number = 0;
  todo: TodoModel|undefined = undefined;

  constructor(private activatedRoute: ActivatedRoute) {
    this.id = +activatedRoute.snapshot.params['id'];
    // cf next: load from id ...  this.todo = data;
  }
}
```

Todo Detail Page



Back to Todo List button

```
18    <A routerLink="/todos">Todo List page</A>
```



Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service
HttpClient (call to API backend)

Advanced (ag-grid, …)

Storing model in Service class (lifecycle != Component)

```
$ ng g s Todo
CREATE src/app/todo.service.spec.ts (347 bytes)
CREATE src/app/todo.service.ts (133 bytes)
```

todo.service.ts

```
1  import { Injectable } from '@angular/core';
2
3  @Injectable({
4    providedIn: 'root'
5  })
6  export class TodoService {
7
8    constructor() { }
9  }
```

inject Service in Component constructor

todo-list-page.component.ts

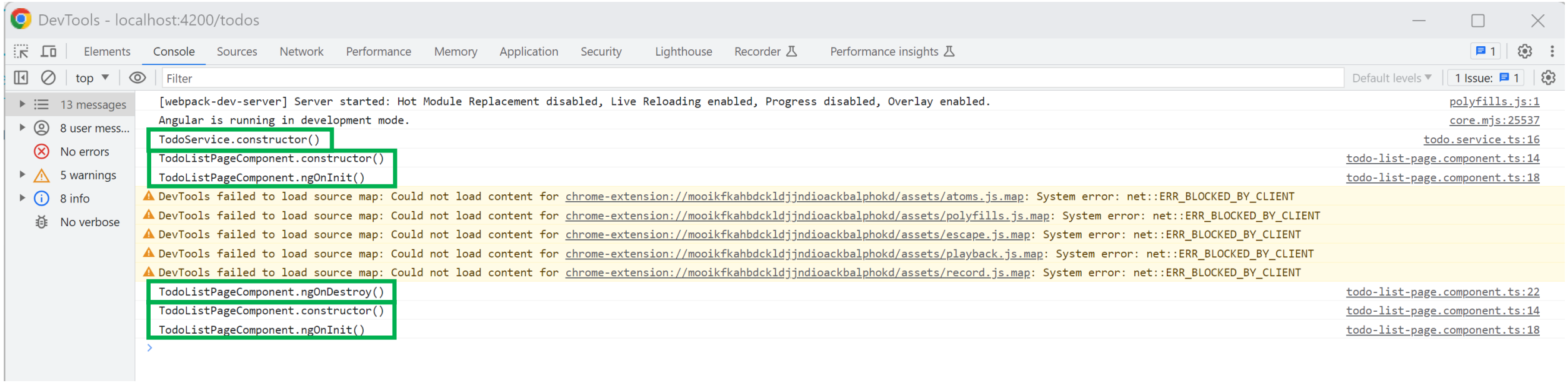
```
1  import { Component } from '@angular/core';
2  import { Router } from '@angular/router';
3  import { TodoModel } from '../model/todo';
4  import { TodoService } from '../todo.service';
5
6  @Component({
7    selector: 'app-todo-list-page',
8    templateUrl: './todo-list-page.component.html'
9  })
10 export class TodoListPageComponent {
11
12   constructor(private todoService: TodoService,
13               private router: Router) {
14   }
```

Showing LifeCycle: console.log() in constructor(), ngOnInit(), ngOnDestroy()

todo-list-page.component.ts

```
1  import { Component, OnInit, OnDestroy } from '@angular/core';
2  import { Router } from '@angular/router';
3  import { TodoModel } from '../model/todo';
4  import { TodoService } from '../todo.service';
5
6  @Component({
7    selector: 'app-todo-list-page',
8    templateUrl: './todo-list-page.component.html'
9  })
10 export class TodoListPageComponent implements OnInit, OnDestroy {
11
12   constructor(private todoService: TodoService, private router: Router) {
13     console.log('TodoListPageComponent.constructor()');
14   }
15
16   ngOnInit() {
17     console.log('TodoListPageComponent.ngOnInit()');
18   }
19
20   ngOnDestroy() {
21     console.log('TodoListPageComponent.ngOnDestroy()');
22   }
```

Showing Life Cycle



Refactor: move « todos » code in service

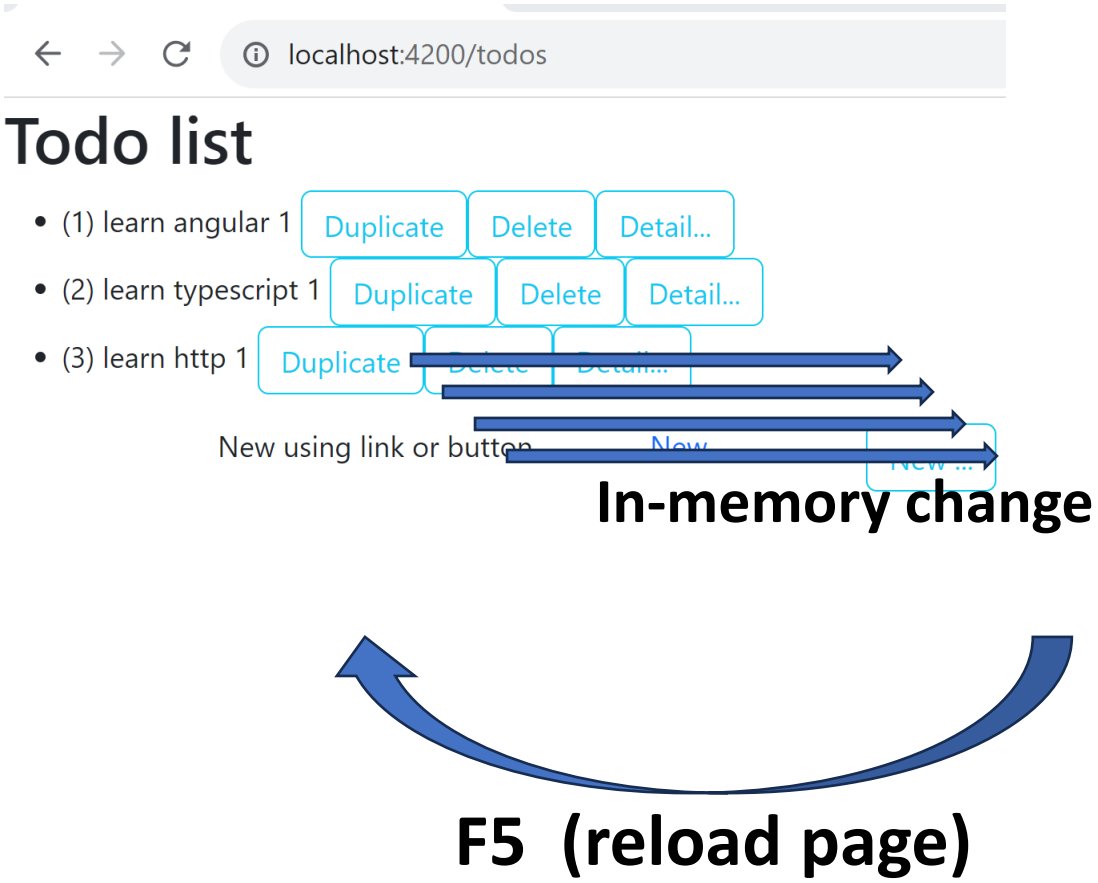
todo.service.ts

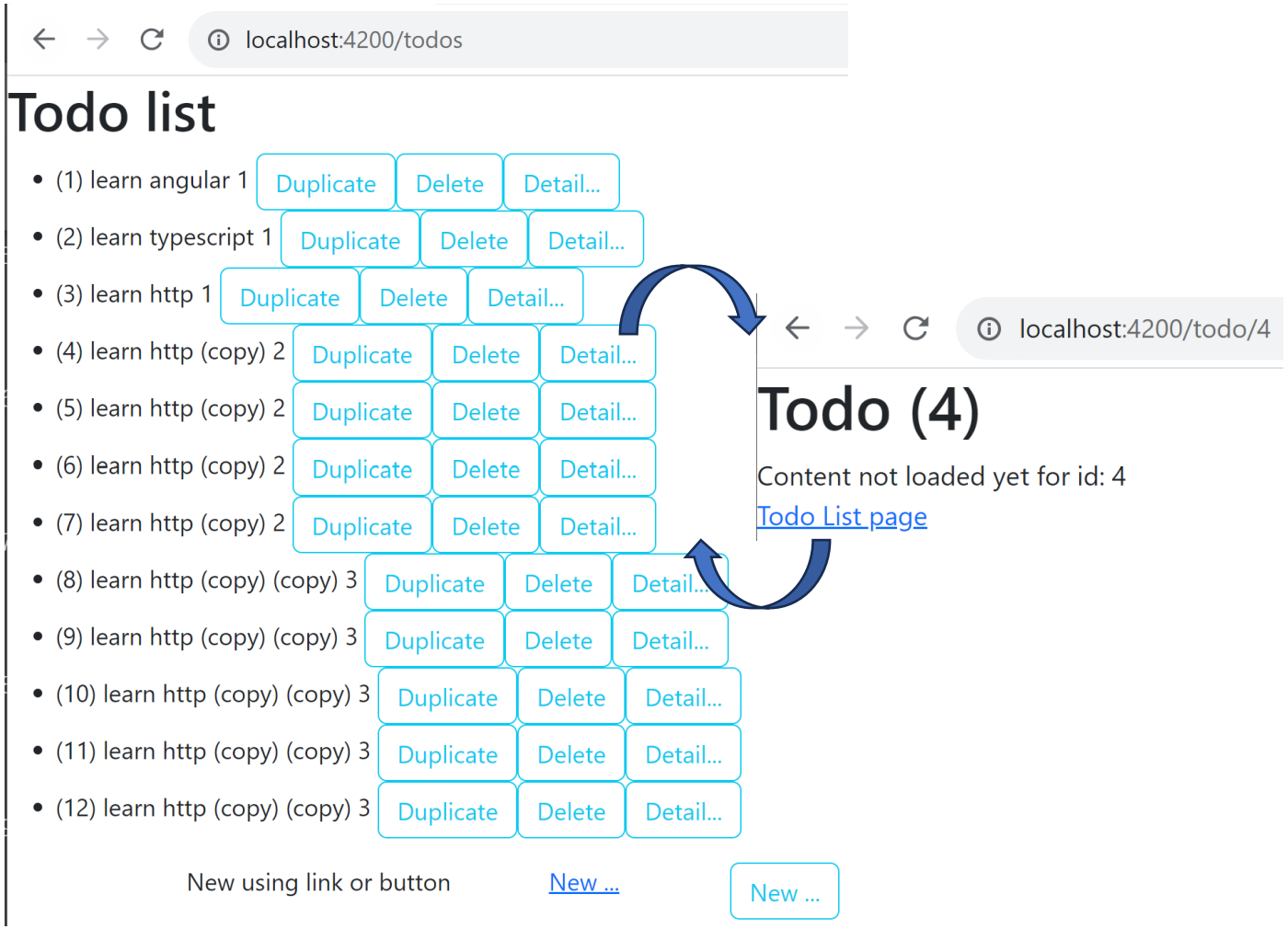
```
1  import { Injectable } from '@angular/core';
2  import { TodoModel } from './model/todo';
3
4  @Injectable({
5    providedIn: 'root'
6  })
7  export class TodoService {
8
9    todos: TodoModel[] = [
10     new TodoModel(1, "learn angular", 1),
11     new TodoModel(2, "learn typescript", 1),
12     new TodoModel(3, "learn http", 1),
13   ];
14
15   constructor() { }
16
17   duplicateTodo(item: TodoModel, idx: number) {
18     let newId = this.todos.length + 1;
19     let dupItem = new TodoModel(newId, item.description + ' (copy)',
20                                 item.priority+1);
21     this.todos.push(dupItem);
22   }
23
24   deleteTodo(item: TodoModel, idx: number) {
25     this.todos.splice(idx, 1);
26   }
27
28 }
```

Refactor Component: call service methods

```
24    get todos(): TodoModel[] { return this.todoService.todos; }
25
26    duplicateTodo(item: TodoModel, idx: number) {
27      this.todoService.duplicateTodo(item, idx);
28    }
29
30    deleteTodo(item: TodoModel, idx: number) {
31      this.todoService.deleteTodo(item, idx);
32    }
```

Navigate pages => Todos in-memory stored
Refresh page => Todos reset





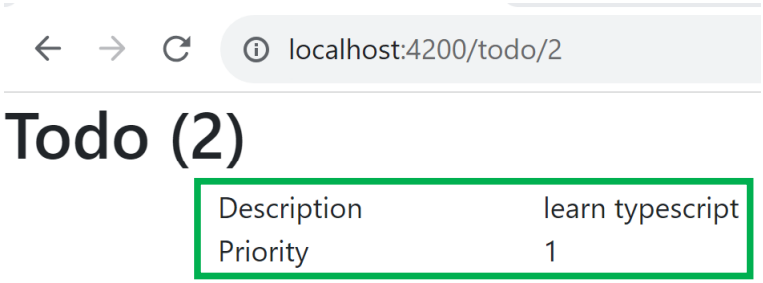
Using Service for detailed page « /todo/:id » « todoService.findById(id) »

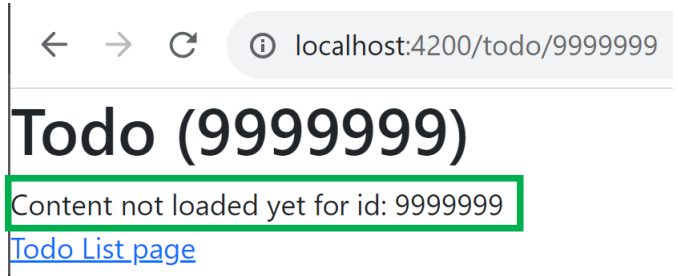
todo-detail-page.component.ts

```ts
import { Component } from '@angular/core';
import { ActivatedRoute } from '@angular/router';

import { TodoModel } from '../model/todo';
import { TodoService } from '../todo.service';

@Component({
  selector: 'app-todo-detail-page',
  templateUrl: './todo-detail-page.component.html'
})
export class TodoDetailPageComponent {

  id: number = 0;
  todo: TodoModel|undefined = undefined;

  constructor(private todoService: TodoService,
              private activatedRoute: ActivatedRoute) {
    this.id = +activatedRoute.snapshot.params['id'];
    this.todo = this.todoService.findById(this.id);
  }
}
```

todo.service.ts

```ts
findById(id: number): TodoModel|undefined {
  return this.todos.find(x => x.id === id);
}
```





Using Service for « new » Form page

todo-create-simple.component.ts

```ts
import { Component } from '@angular/core';
import { Router } from '@angular/router';

import { TodoModel } from '../model/todo';
import { TodoService } from '../todo.service';

@Component({
  selector: 'app-todo-create-simple',
  templateUrl: './todo-create-simple.component.html',
})
export class TodoCreateSimpleComponent {

  todo: TodoModel = new TodoModel(0, "", 1);

  constructor(private todoService: TodoService,
              private router: Router) {}

  onClickNew() {
    let newObj = this.todoService.saveNew(this.todo);
    this.router.navigate([ 'todo', newObj.id ]);
  }

}
```

todo.service.ts

```ts
  saveNew(src: TodoModel): TodoModel {
    let newId = this.generateNewId();
    let obj = new TodoModel(newId, src.description, src.priority)
    this.todos.push(obj);
    return obj;
  }

  generateNewId(): number {
    return 1 + this.todos.reduce(
        (acc,val) => [ Math.max(acc[0], val.id) ],
        [0]
        )[0];
  }
```

Demo Outline

Reminder : http client/server, SPA: Single Page Application
Classical Web app tutorial: « Todo web application »
Rest Api Backend Server – NodeJS,Express,Swagger

Angular + ng-bootstrap setup
Component Pages (offline / mock data)
- List Page
- Duplicate/Delete button
- Detail Page
- New Page
- links between pages

Service

HttpClient (call to API backend)

Advanced (ag-grid, …)

Using HttpClient Module

app.module.ts

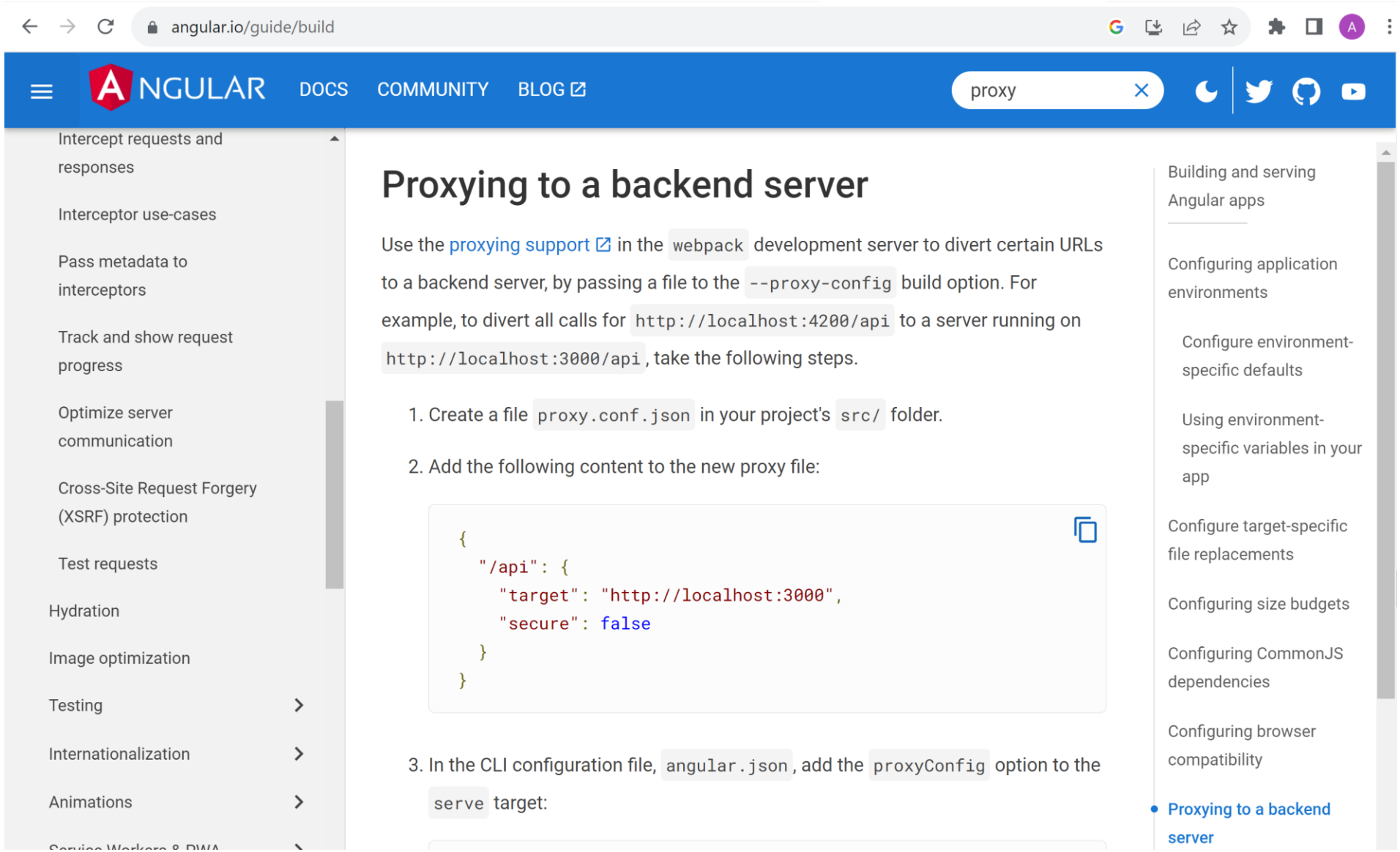
```
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3  import { FormsModule } from '@angular/forms';
4  import { HttpClientModule } from '@angular/common/http';
5
14 @NgModule({
15   declarations: [
16     AppComponent, TodoListPageComponent, TodoCreateSimpleCompor
17   ],
18   imports: [
19     BrowserModule, AppRoutingModule, NgbModule, FormsModule,
20     HttpClientModule
21   ],
22   providers: [],
23   bootstrap: [AppComponent]
24 })
25 export class AppModule { }
```

Calling http from Service

todo.service.ts

```
1   import { Injectable } from '@angular/core';
2   import { HttpClient } from '@angular/common/http';


17    constructor(private httpClient: HttpClient) {
18      console.log('TodoService.constructor()');
19
20      console.log('Calling http GET /api/todo ... subscribe to async response');
21      this.httpClient.get('/api/todo').subscribe(resp => {
22        console.log('Finished http GET /api/todo', resp);
23      }, err => {
24        console.log('Failed http GET /api/todo', err);
25      });
26    }
```

Configuring angular proxy.json

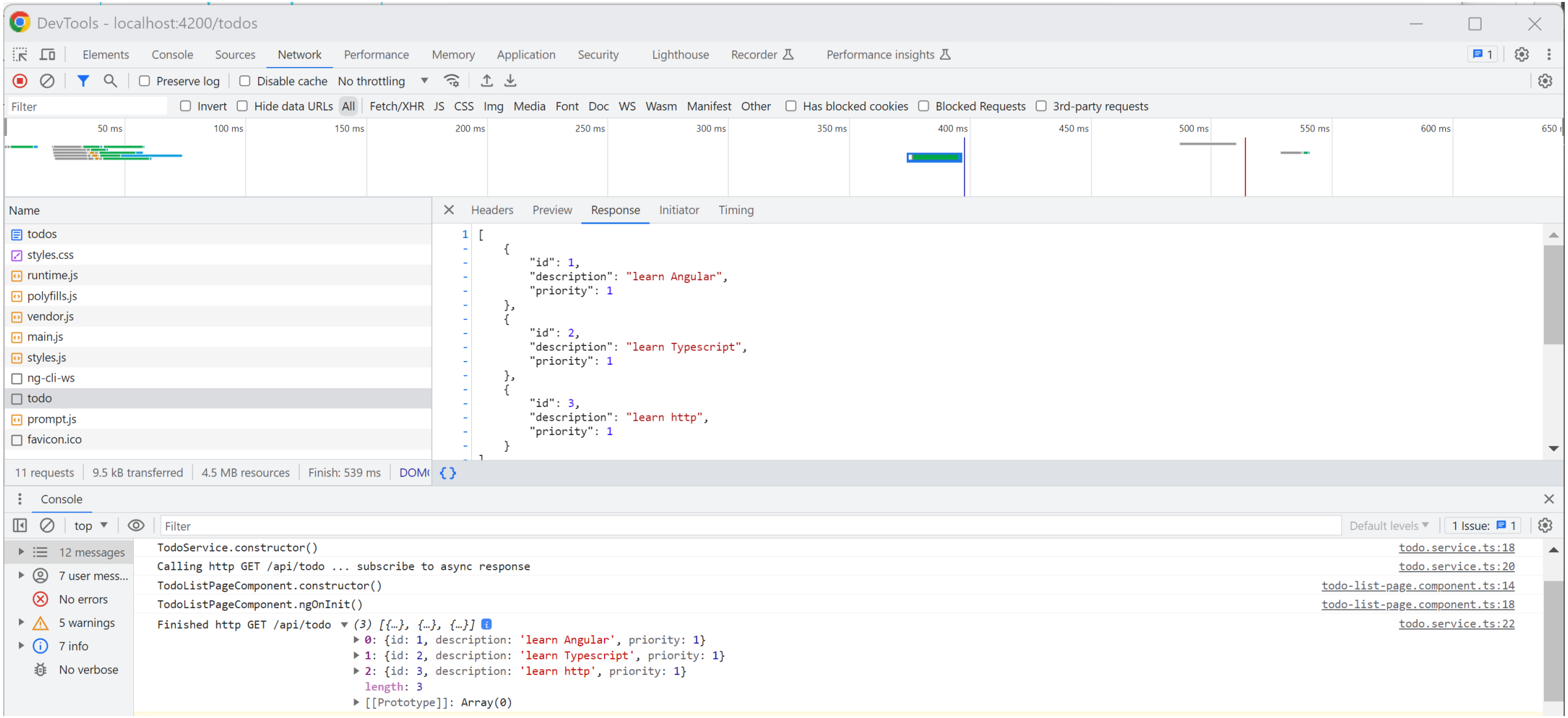


proxy.json

proxy.conf.json

```
{
  "/api": {
    "target": "http://localhost:3000",
    "secure": false,
    "pathRewrite": {
      "^/api": ""
    }
  }
}
```

Test http GET http://localhost:4200/api/todo => http GET http://localhost:3000/todo



Take Away

Questions

Next Steps …