Лабораторна робота №6

**Тема:** Створення менеджеру справ з допомогою Angular/

**Мета:** Створити менеджер справ з допомогою Yii2

Хід роботи

Для розгортання Angular проекту потрібно щоб Ваше середовище розробки включало Node.js (8.x чи 9.x) та npm.

Також для зручнішої роботи з Angular використовується Angular CLI.

Тепер за допомогою Angular CLI розгорнимо новий проект, який вже буде включати потрібну нас структуру.

Створимо клас **Task** використовуючи Angular CLI

ng generate class tasks/Task

У створений файл внесемо зміни:

src/app/tasks/task.ts

**export class** Task {  
 id: **number**;  
 title: **string**;  
 completed: **false**;  
  
 **constructor**() {  
 **this**.completed = **false**;  
 }  
}

Далі створимо TaskDataService, що буде відповідати за управління класу Task. Це дозволить нам створити / видалити та змінити статус об'єкта task.

ng generate service tasks/TaskData

src/app/todos/todo-data.service.ts

**import** {Injectable} **from** '@angular/core';  
**import** {Task} **from** './task';  
  
@Injectable()  
**export class** TaskDataService {  
  
 // Placeholder for last id so we can simulate  
 // automatic incrementing of id's  
 lastId = 0;

// Placeholder for task's  
 tasks: Task[] = [];  
  
 **constructor**() {  
 }  
  
 // Simulate POST /tasks  
 addTask(task: Task): TaskDataService {  
 **if** (!task.id) {  
 task.id = ++**this**.lastId;  
 }  
 **this**.tasks.push(task);  
 **return this**;  
 }  
  
 // Simulate DELETE /tasks/:id  
 deleteTaskById(id: **number**): TaskDataService {  
 **this**.tasks = **this**.tasks  
 .filter(task => task.id !== id);  
 **return this**;  
 }  
  
 // Simulate PUT /tasks/:id  
 updateTaskById(id: **number**, values: Object = {}): Task | **any** {  
 **const** task = **this**.getTaskById(id);  
 **if** (!task) {  
 **return this**;  
 }  
 Object.assign(task, values);  
 **return** task;  
 }  
  
 // Simulate GET /tasks  
 getAllTasks(): Task[] {  
 **return this**.tasks;  
 }  
  
 // Simulate GET /tasks/:id  
 getTaskById(id: **number**): Task | **any** {  
 **return this**.tasks  
 .filter(task => task.id === id)  
 .pop();  
 }  
  
 // Toggle task complete  
 changeState(task: Task) {  
 **const** updatedTask = **this**.updateTaskById(task.id, {  
 completed: !task.completed  
 });  
 **return** updatedTask;  
 }  
  
 getTaskByState(completed: **boolean**): Task[] {  
 **return this**.tasks  
 .filter(task => task.completed === completed);  
 }  
}

Під час ініціалізації проекту, Angular CLI автоматично генерує головний компонент AppComponent для нас:

src/app/app.component.css

src/app/app.component.html

src/app/app.component.spec.ts

src/app/app.component.ts

Так як проект не складний всі зміни будуть вносяться тільки у межах AppComponent-а.

src/app/app.component.html

<div class="app-root-content container mt-3">  
 <div class="row">  
 <div class="col-md-3">  
 <div class="card">  
 <div class="card-header">  
 Filter  
 </div>  
 <div class="card-body">  
 <div class="custom-control custom-radio">  
 <input type="radio" id="customRadioInline1" name="customRadioInline1" class="custom-control-input" checked (click)="onSelect(0)">  
 <label class="custom-control-label" for="customRadioInline1">All</label>  
 </div>  
 <div class="custom-control custom-radio">  
 <input type="radio" id="customRadioInline2" name="customRadioInline1" class="custom-control-input" (click)="onSelect(1)">  
 <label class="custom-control-label" for="customRadioInline2">Completed</label>  
 </div>  
 <div class="custom-control custom-radio">  
 <input type="radio" id="customRadioInline3" name="customRadioInline1" class="custom-control-input" (click)="onSelect(2)">  
 <label class="custom-control-label" for="customRadioInline3">Not completed</label>  
 </div>  
 </div>  
 </div>  
 </div>  
 <div class="col-md-9">  
 <div class="card">  
 <div class="card-header">  
 {{filters[currentFilter]}} tasks  
 </div>  
 <div class="card-body">  
 <div class="row">  
 <div class="col-md-12">  
 <div class="m-2" \*ngFor="let task of taskList; index as i">  
 <div class="custom-control custom-checkbox float-left">  
 <input type="checkbox" class="custom-control-input" id="task\_{{i}}"  
 [checked]="task.completed" (click)="changeTaskState(task)">  
 <label class="custom-control-label" for="task\_{{i}}"  
 [ngClass]="{'strikethrough' : task.completed}">{{task.title}}</label>  
 </div>  
 <button type="button" class="btn float-right"  
 (click)="removeTask(task)">  
 <i class="fa fa-trash"></i>  
 </button>  
 <div class="clearfix"></div>  
 </div>  
 </div>  
 </div>  
 <div class="row">  
 <div class="col-md-12">  
  
 <div class="m-2 text-muted">  
 <span \*ngIf="taskList.length > 0; else elseBlock">Amount of tasks:  
 {{taskList.length}}</span>  
 <ng-template #elseBlock>  
 <span>List is empty</span>  
 </ng-template>  
 </div>  
 <div class="clearfix"></div>  
  
 <form [formGroup]="form" (submit)="addTask()">  
 <div class="form-row">  
 <div class="col-md-8">  
 <label class="sr-only" for="inlineFormInput">Name</label>  
 <input type="text" class="form-control" id="inlineFormInput"  
 placeholder="Task description" formControlName="task" required>  
 </div>  
 <div class="col-auto">  
 <button type="submit" class="btn btn-success">Add item</button>  
 </div>  
 </div>  
 </form>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
</div>  
  
<router-outlet></router-outlet>

src/app/app.component.ts

**import** {Component} **from** '@angular/core';  
**import** {FormControl, FormGroup} **from** "@angular/forms";  
**import** {TaskDataService} **from** "./tasks/task-data.service";  
**import** {Task} **from** "./tasks/task";  
**import** {isUndefined} **from** "util";  
  
@Component({  
 selector: 'app-root',  
 templateUrl: './app.component.html',  
 styleUrls: ['./app.component.css'],  
 providers: [TaskDataService]  
})  
**export class** AppComponent {  
  
 **public** newTask: Task = **new** Task();  
  
 **public** currentFilter: **number** = 0;  
  
 **public** filters = {  
 0: "All",  
 1: "Completed",  
 2: "Not completed",  
 };  
  
 **public** form: FormGroup = **new** FormGroup({  
 task: **new** FormControl()  
 });  
  
 **constructor**(**private** taskDataService: TaskDataService) {  
 }  
  
 **get** taskList(): Task[] {  
 console.log(**this**.currentFilter);  
 **switch** (**this**.currentFilter) {  
 **case** 0: {  
 **return this**.taskDataService.getAllTasks();  
 }  
 **case** 1: {  
 **return this**.taskDataService.getTaskByState(**true**);  
 }  
 **case** 2: {  
 **return this**.taskDataService.getTaskByState(**false**);  
 }  
 }  
 }  
  
 addTask() {  
 **if** (**this**.form.value.task) {  
 **this**.newTask.title = **this**.form.value.task;  
 **this**.taskDataService.addTask(**this**.newTask);  
 **this**.newTask = **new** Task();  
 **this**.form.reset();  
 console.log(**this**.taskDataService.tasks);  
 }  
 }  
  
 removeTask(task: Task) {  
 **this**.taskDataService.deleteTaskById(task.id);  
 console.log(**this**.taskDataService.tasks);  
 }  
  
 changeTaskState(task: Task) {  
 **this**.taskDataService.changeState(task);  
 console.log(**this**.taskDataService.tasks);  
 }  
  
 onSelect(filter: **number**){  
 **this**.currentFilter = filter;  
 }  
}

src/app/app.component.css

.strikethrough {  
 text-decoration: line-through;  
}

Виконавши команду ***ng serve -o***. Ви побачите нашу нову програму в браузері.

Результат

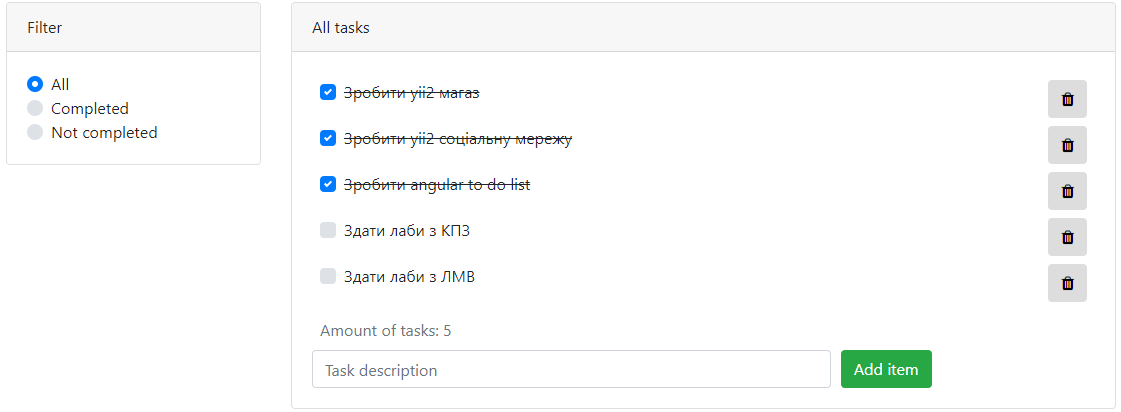


Рисунок 1 - Усі справи

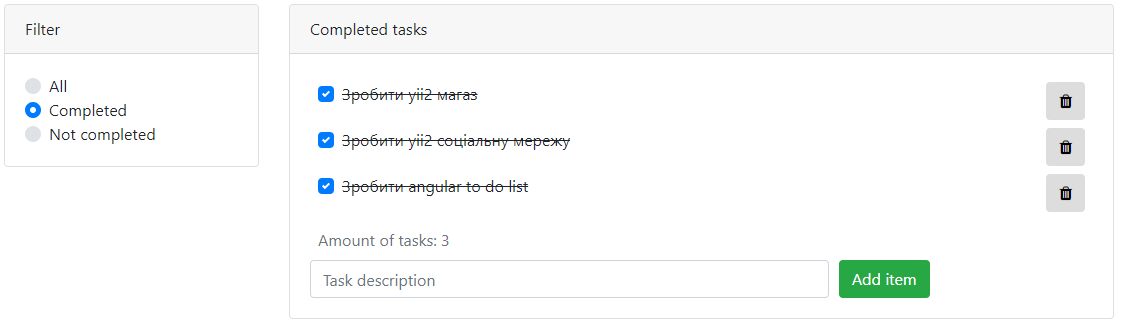


Рисунок 2 - Виконані справи

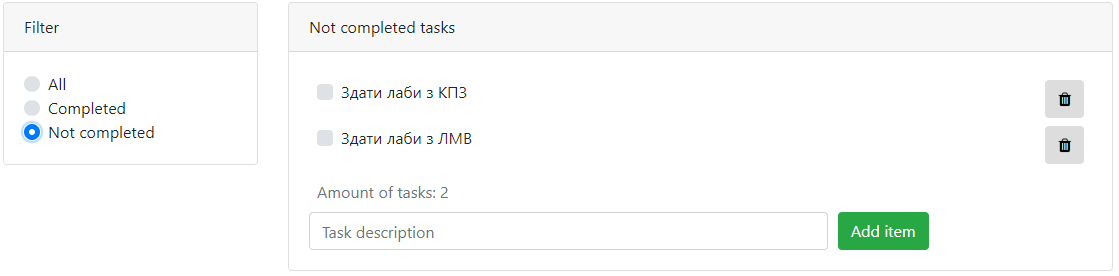


Рисунок 3 - Не виконані справи

Використані джерела:

* [https://angular.io/docs](https://angular.io/docs%20) | Документація Angular

Висновок.

На даній лабораторній роботі я спробував створити власну менедж справ на основі Angular.