

Лабораторна робота 12.

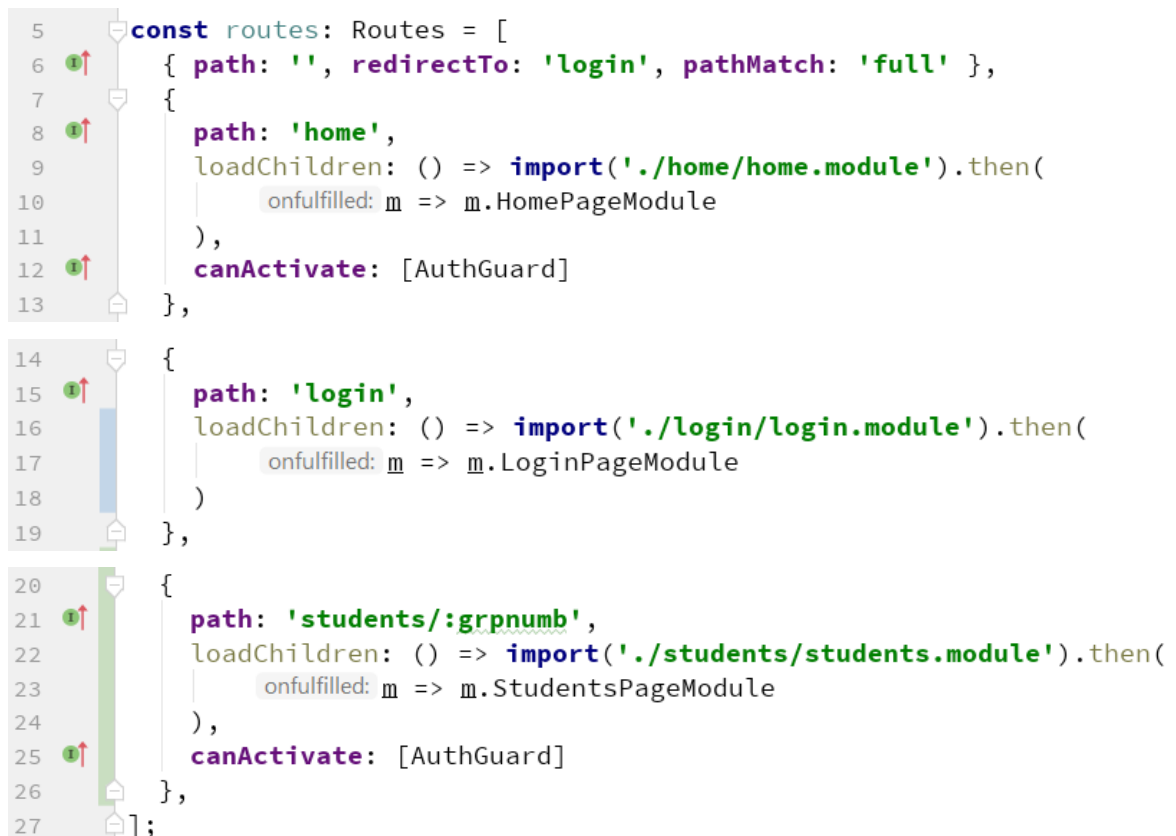
Ionic Framework. Додавання сторінок та передача даних.

Продовжимо роботу над проектом «Студентські групи». Додаємо нову сторінку, на якій будемо відображати список студентів певної групи (рис. 1).

```
ng g page students
```

Рис. 1. Додавання сторінки списку студентів.

Далі перейдемо до файлу маршрутизації `app-routing.module.ts` та трохи змінимо маршрут для сторінки виводу списку студентів для можливості передачі у маршрут номеру студентської групи. Також підключимо до цього маршруту створений у попередній роботі `guard` для заборони роботи неаутентифікованих користувачів (рис. 2).



```
5  const routes: Routes = [  
6    { path: '', redirectTo: 'login', pathMatch: 'full' },  
7  ],  
8  {  
9    path: 'home',  
10   loadChildren: () => import('./home/home.module').then(  
11     m => m.HomePageModule  
12   ),  
13   canActivate: [AuthGuard]  
14 },  
15 {  
16   path: 'login',  
17   loadChildren: () => import('./login/login.module').then(  
18     m => m.LoginPageModule  
19   ),  
20 },  
21 {  
22   path: 'students/:grpnumb',  
23   loadChildren: () => import('./students/students.module').then(  
24     m => m.StudentsPageModule  
25   ),  
26   canActivate: [AuthGuard]  
27 },  
28 ];
```

Рис. 2. Зміни у `app-routing.module.ts`.

У сервіс `DataGetter` додаємо масив студентів та метод для отримання даних студентів певної групи (рис. 3).

```

31 private students = [
32     {name: 'Іванов Василь', groupNumb: 201,
33       gender: 'man', rating: 91},
34     {name: 'Дмитренко Петро', groupNumb: 201,
35       gender: 'man', rating: 85},
36     {name: 'Петренко Дарина', groupNumb: 201,
37       gender: 'woman', rating: 68},
38     {name: 'Васильєва Марина', groupNumb: 308,
39       gender: 'woman', rating: 78},
40     {name: 'Павлов Микита', groupNumb: 308,
41       gender: 'man', rating: 82},
42     {name: 'Васін Андрій', groupNumb: 308,
43       gender: 'man', rating: 75},
44 ];

77
78 getStudents(groupNumber: number): Observable<any[]> {
79     return of(this.students.filter( callbackfn: elem => {
80         return elem.groupNumb === groupNumber;
81     }));
82 }

```

Рис. 3. Зміни у сервісі data-getter.service.ts.

У класі StudentsPage підключаємо інстанцію ActivatedRoute для отримання номеру групи із маршруту та сервіс отримання даних DataGetter для отримання списку студентів за цим номером (рис. 4).

```

1  import {Component, OnInit} from '@angular/core';
2  import {DataGetterService} from '../service/data-getter.service';
3  import {ActivatedRoute} from '@angular/router';
4
5  @Component({
6      selector: 'app-students',
7      templateUrl: './students.page.html',
8      styleUrls: ['./students.page.scss'],
9  })
10
11  export class StudentsPage implements OnInit {
12      grpnumb: number;
13      students: any[];
14
15      constructor(private dataGetter: DataGetterService,
16                  private route: ActivatedRoute) { }

```

```

17  ngOnInit() {
18      this.grpnumb = +this.route.snapshot.paramMap.get('grpnumb');
19      this.dataGetter.getStudents(this.grpnumb).subscribe(
20          next: data => {
21              this.students = data;
22          }
23      );
24  }
25  }

```

Рис. 4. Файл students.page.ts.

Також реалізуємо представлення для новоствореної сторінки (рис. 5).

```

1  <ion-header>
2      <ion-toolbar>
3          <ion-buttons slot="start">
4              <ion-back-button></ion-back-button>
5          </ion-buttons>
6          <ion-title>Студенти групи {{grpnumb}}</ion-title>
7      </ion-toolbar>
8  </ion-header>
9
10 <ion-content>
11     <ion-list>
12         <ion-item *ngFor="let student of students" text-wrap>
13             <ion-icon [name]="student.gender" slot="start"></ion-icon>
14             <ion-label text-wrap>{{student.name}}</ion-label>
15             <ion-note slot="end">{{student.rating}}</ion-note>
16         </ion-item>
17     </ion-list>
18 </ion-content>

```

Рис. 5. Представлення students.page.html.

У представленні списку студентських груп у елементі ion-item-sliding додаємо посилання для переходу до сторінки списку студентів. Нижче наводимо повний код елементу списку ion-item-sliding (рис. 6).

```

31 <ion-item-sliding>
32     <ion-item-options side="start">
33         <ion-item-option color="primary" (click)="showEdit=i">
34             <ion-icon name="create"></ion-icon>
35             Змінити
36         </ion-item-option>

```

```

37 <ion-item-option color="danger" (click)="delete(i)">
38   <ion-icon name="trash"></ion-icon>
39   Видалити
40 </ion-item-option>
41 </ion-item-options>
42 <ion-item (click)="showEdit=-1">
43   <ion-icon name="people" slot="start"></ion-icon>
44   <ion-label>{{group.number}}</ion-label>
45   <ion-note slot="end">{{group.specialty}}</ion-note>
46 </ion-item>
47 <ion-item-options side="end">
48   <ion-item-option color="secondary"
49     routerLink="/students/{{group.number}}"
50     routerDirection="forward">
51     <ion-icon name="reorder"></ion-icon>
52     Студенти
53   </ion-item-option>
54 </ion-item-options>
55 </ion-item-sliding>

```

Рис. 6. Елемент списку студентських груп.

Переглянемо результат, завантажимо сторінку списку студентських груп на перейдемо до списку студентів однієї з груп (рис. 7).

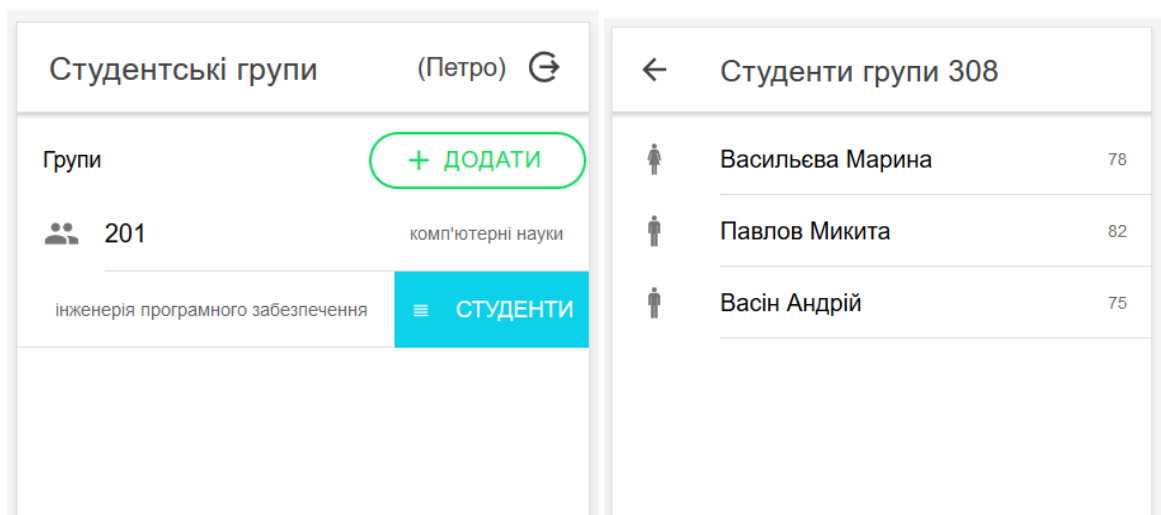


Рис. 7. Перехід до сторінки списку студентів.

Організуємо передачу даних із сторінки списку студентів до головної сторінки. Для цього скористаємось інструментом публікації та підписки на події в angular. На сторінку списку студентів додаємо поле вводу, у яке будемо заносити дані, які хочемо передати до головної сторінки (рис. 8)

```

15     <ion-note slot="end">{{student.rating}}</ion-note>
16   </ion-item>
17 </ion-list>
18 <ion-item>
19   <ion-label position="floating">Дані на home</ion-label>
20   <ion-input type="text" #dataToHome></ion-input>
21 </ion-item>
22 <ion-button color="primary" class="ion-float-right" (click)="passData
   (dataToHome.value)">
23   <ion-icon slot="start" name="send"></ion-icon> Переслати на home
24 </ion-button>
25 </ion-content>

```

Рис. 8. Додавання поля вводу до макету students.page.html.

Після виконаних змін сторінка студентів матиме наступний вигляд (рис. 9).

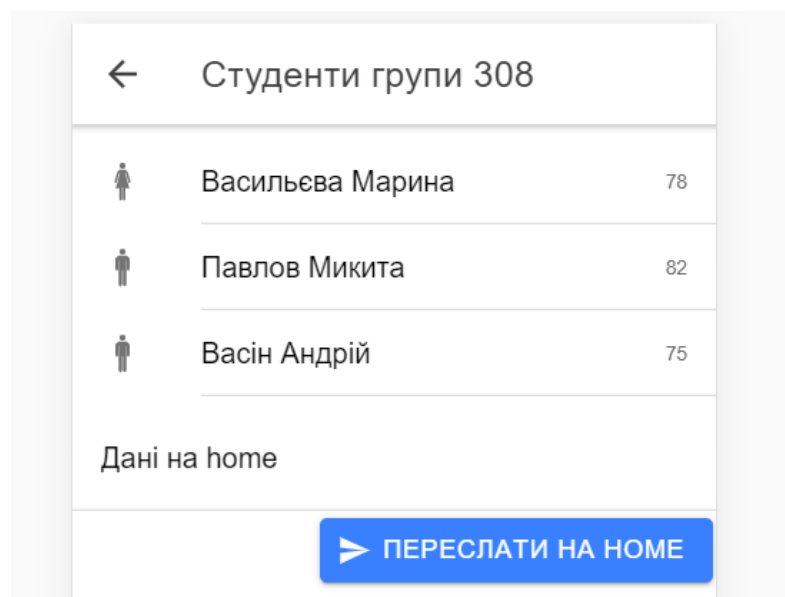


Рис. 9. Сторінка студентів після додавання поля вводу.

У клас students додаємо метод passData, у якому ініціюємо подію, передаємо туди дані та повертаємось на домашню сторінку. Також у конструкторі підключаємо необхідні залежності (рис. 10).

```

4   import {Events, NavController} from '@ionic/angular';
5
15   constructor(private dataGetter: DataGetterService,
16               private route: ActivatedRoute,
17               private events: Events,
18               private navCtrl: NavController) { }
19
28
29   passData(data: string) {
30     this.events.publish( topic: 'students:data', data);
31     this.navCtrl.back();
32   }

```

Рис. 10. Зміни у класі students.page.ts.

На home сторінці також додаємо label для відображення даних, переданих зі сторінки студентів (рис. 11).

```
57 <app-stud-group [studentgroup]="group"></app-stud-group>
58 </div>
59 </ion-list>
60 <ion-item text-center>
61   <ion-label color="primary">{{dataFromStudents}}</ion-label>
62 </ion-item>
63 </ion-content>
```

Рис. 11. Додавання label до home.page.html.

У конструкторі класу home підписуємось на подію, що ініціюємо на сторінці студентів та передаємо отримані дані у змінну, яка відображається у полі label (рис. 12).

```
3 import {Events} from '@ionic/angular';
4
19 dataFromStudents: string;
20
21 constructor(private dataGetter: DataGetterService, private events: Events) {
22   this.dataGetter.getGroups().subscribe(
23     next: (data) => {
24       this.groups = data;
25     }
26   );
27
28   this.userName = this.dataGetter.getUser();
29   this.events.subscribe( topic: 'students:data', handlers: data => {
30     this.dataFromStudents = data;
31   });
32 }
```

Рис. 12. Підписка на подію у home.page.ts.

Перевіримо роботу передачі даних на домашню сторінку (рис. 13).

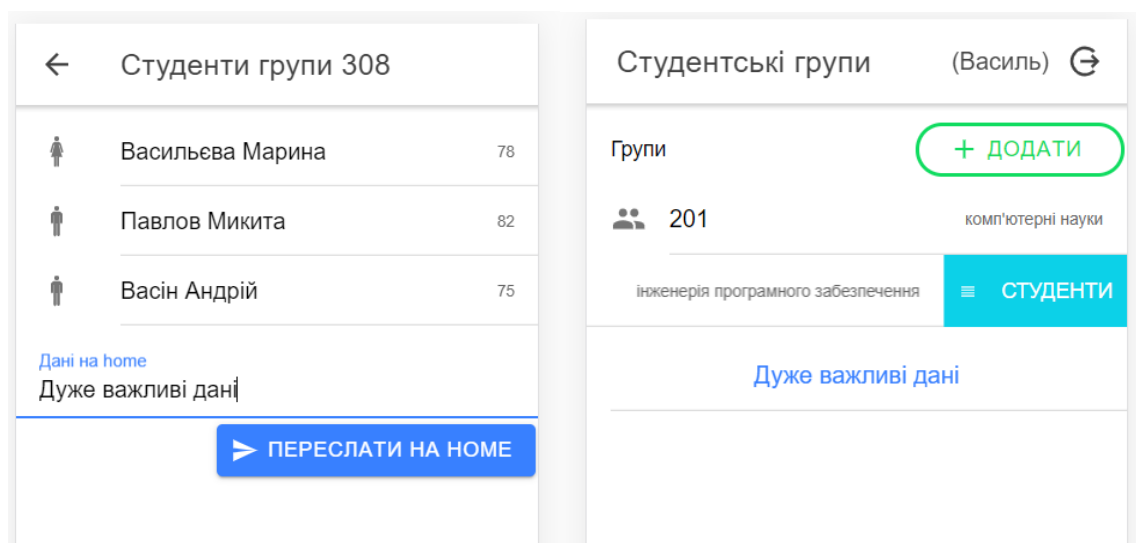


Рис. 13. Перевірка передачі даних.

Виділимо функціонал передачі та повернення даних у окрему сторінку. Для цього створимо сторінку DataSender (рис. 14).

ionic generate page DataSender

Рис. 14. Створення сторінки.

На рис. 15 та 16 наведено представлення та клас нової сторінки відповідно.

```
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button></ion-back-button>
5     </ion-buttons>
6     <ion-title>Повернення даних</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <ion-item>
12     <ion-label position="floating">Дані на home</ion-label>
13     <ion-input type="text" [(ngModel)]="textData"></ion-input>
14   </ion-item>
15   <ion-button color="primary" class="ion-float-right" (click)="passData()">
16     <ion-icon slot="start" name="sync"></ion-icon> Повернути дані
17   </ion-button>
18 </ion-content>
```

Рис. 15. data-sender.page.html.

```
1 import { Component, OnInit } from '@angular/core';
2 import { Events, NavController } from '@ionic/angular';
3 import { ActivatedRoute, Router } from '@angular/router';
4
5 @Component({
6   selector: 'app-data-sender',
7   templateUrl: './data-sender.page.html',
8   styleUrls: ['./data-sender.page.scss'],
9 })
10 export class DataSenderPage implements OnInit {
11
12   textData: string;
13
14   constructor(private events: Events,
15               private navCtrl: NavController,
16               private route: ActivatedRoute) {
17     this.textData = this.route.snapshot.paramMap.get('data');
18   }
```

```

19
20 ngOnInit() {
21 }
22
23 passData() {
24   this.events.publish( topic: 'students:data', this.textData);
25   this.navCtrl.back();
26 }
27 }

```

Рис. 16. data-sender.page.ts.

Також на сторінці home змінимо label для відображення даних на input, та виконаємо початкову передачу даних із домашньої сторінки через параметр маршруту (рис. 17 – 18).

```

58 </div>
59 </ion-list>
60 <ion-item text-center>
61   <ion-input type="text" [(ngModel)]="extraData"></ion-input>
62 </ion-item>
63 <ion-button color="primary" class="ion-float-right" (click)="getData()">
64   <ion-icon slot="start" name="sync"></ion-icon> Передати дані
65 </ion-button>
66 </ion-content>

```

Рис. 17. home.page.html.

```

4 import {Router} from '@angular/router';
5
20 extraData: string;
21
22 constructor(private dataGetter: DataGetterService,
23             private events: Events,
24             private router: Router) {
25   this.dataGetter.getGroups().subscribe(
26     next: (data) => {
27       this.groups = data;
28     }
29   );
30   this.userName = this.dataGetter.getUser();
31   this.events.subscribe( topic: 'students:data', handlers: data => {
32     this.extraData = data;
33   });
34 }
35
52
53 getData() {
54   this.router.navigate( commands: ['/data-sender', {data: this.extraData}]);
55 }

```

Рис. 18. home.page.ts.

Перевіримо передачу даних зі головної сторінки, та повернення їх назад (рис. 19).

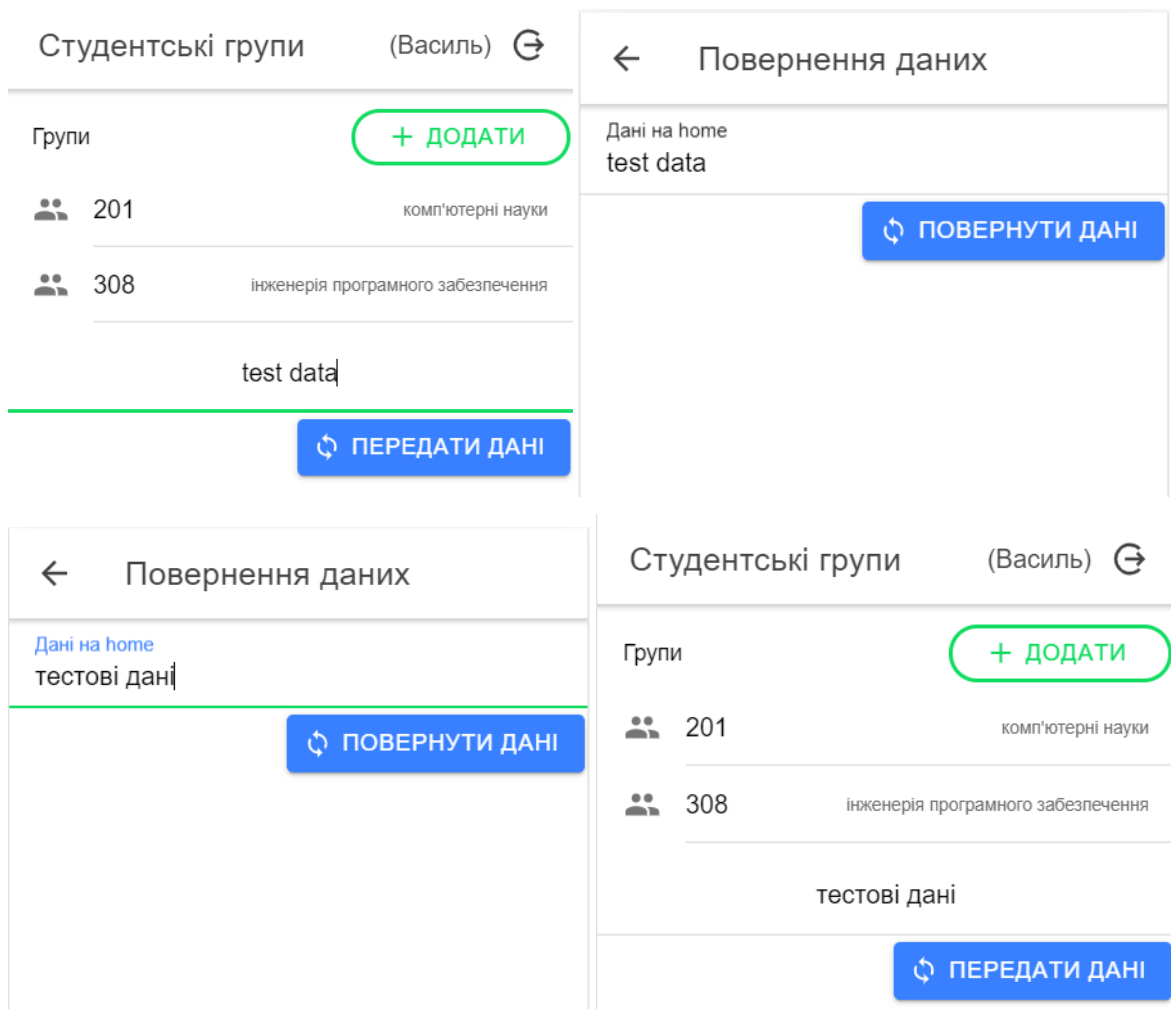


Рис. 19. Передача та повернення даних.

Змінимо спосіб повернення даних на головну сторінку із події та передачу параметру маршруту. Для цього виконаємо невеликі зміни у класі data-sender (рис. 20) та у класі home (рис. 21).

```

2   import {ActivatedRoute, Router} from '@angular/router';
13
14   constructor(private route: ActivatedRoute,
15               private router: Router) {
16       this.textData = this.route.snapshot.paramMap.get('data');
17   }
21
22   passData() {
23       this.router.navigate(['home'], {data: this.textData});

```

Рис. 20. Зміни у data-sender.page.ts.

```

4   import {ActivatedRoute, Router} from '@angular/router';
5

```

```

22     constructor(private dataGetter: DataGetterService,
23                 private events: Events,
24                 private router: Router,
25                 private route: ActivatedRoute) {
26         this.dataGetter.getGroups().subscribe(
27             next: (data) => {
28                 this.groups = data;
29             }
30         );
31         this.userName = this.dataGetter.getUser();
32         this.extraData = this.route.snapshot.paramMap.get('data');
33     }

```

Рис. 21. Зміни у home.page.ts.

При перевірці передачі та повернення даних все має працювати, як і раніше.

І, наостанок, розглянемо передачу даних із використанням власного сервісу. Створимо новий сервіс (рис. 22).

ionic generate service service/DataExchanger

```

1  import { Injectable } from '@angular/core';
2
3  interface DataElem {
4      k: string;
5      d: any;
6  }
7
8  @Injectable({
9      providedIn: 'root'
10 })
11
12 export class DataExchangerService {
13
14     data: DataElem[] = [];
15
16     constructor() { }
17
18     getData(key: string) {
19         const arr = this.data.filter( callbackfn: elem => elem.k === key);
20         if (arr.length) {
21             return arr[0].d;
22         }
23         return '';
24     }

```

```

25
26 setData(key: string, data: any) {
27     const arr = this.data.filter( callbackfn: elem => elem.k === key);
28     if (arr.length) {
29         arr[0].d = data;
30     } else {
31         this.data.push({
32             k: key, d: data
33         });
34     }
35 }
36 }

```

Рис. 22. Створення нового сервісу data-exchanger.service.ts.

Також виконаємо необхідні зміни у класах home та data-sender відповідно (рис. 23-24).

```

22 constructor(private dataGetter: DataGetterService,
23             private router: Router,
24             private dataExchanger: DataExchangerService) {
25     this.dataGetter.getGroups().subscribe(
26         next: (data) => {
27             this.groups = data;
28         }
29     );
30     this.userName = this.dataGetter.getUser();
31 }

32
33 ionViewDidEnter() {
34     console.log('home');
35     this.extraData = this.dataExchanger.getData( key: 'myData');
36 }

53
54 getData() {
55     this.dataExchanger.setData( key: 'myData', this.extraData);
56     this.router.navigate( commands: ['/data-sender']);
57 }

```

Рис. 23. Зміни у home.page.ts.

```

13
14 constructor(private dataExchanger: DataExchangerService,
15             private router: Router) {
16     this.textData = this.dataExchanger.getData( key: 'myData');
17 }

21
22 passData() {
23     this.dataExchanger.setData( key: 'myData', this.textData);
24     this.router.navigate( commands: ['/home']);
25 }

```

Рис. 24. Зміни у data-sender.page.ts.

Завдання для самостійного виконання.

Реалізувати у застосунку перехід до нової сторінки передачею параметру до маршруту та фільтрацією даних, що відображаються згідно переданого параметру. Реалізувати редагування даних на дочірній сторінці. Передбачити передачу та повернення даних із дочірньої сторінки (будь-яким способом на вибір).