

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ
СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Звіт по лабораторній роботі № 3
Створення проекту «Todo».
з дисципліни: «Реактивне програмування»

Студент: Черпак Андрій Вадимович

Група: ІП-01

Дата захисту роботи: 20.10.2023

Викладач: доц. Полупан Юлія Вікторівна

Захищено з оцінкою: _____

Київ, 2023

Зміст

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО» 1

Встановлення необхідних пакетів	3
Ініціалізація проекту	3
Створення базових класів.	5
Додавання стилів до за стосунку.	7
Відображення списку справ.	10
Додавання двосторонньої прив'язки до чекбоксів.	12
Фільтрування завершених завдань.	14
Додавання нових завдань.....	16
Відображення виконаних завдань.....	20
Хостинг.....	24
Висновок:	25
Список використаних джерел:	26

Мета: Знайомство з основним процесом розробки Angular-додатку з використанням редактора Visual Studio Code. Навчитися застосовувати компоненти Angular Material для стилізації контенту та створювати моделі даних. Навчитися створювати двосторонні прив'язки даних.

Завдання:

I) Встановити Visual Studio Code, пакет Angular Material. Створити каркас додатку «Todo». Додати наступні можливості до проекту «Todo»: відображення списку завдань у вигляді таблиці; додавання нових завдань користувачем; фільтрування завдань користувачем.

II) Зробити звіт по роботі.

III) Angular-додаток «Todo» розгорнути на платформі Firebase у проекті з ім'ям «ПрізвищеГрупаLaba3»

Встановлення необхідних пакетів

Перевіримо версії node.js та npm, а також встановимо пакет AngularCli, або ж переконаємося у наявності:

```
D:\Education\7 sem\reactive\Lab3>node -v
v21.0.0

D:\Education\7 sem\reactive\Lab3>npm -v
10.1.0

D:\Education\7 sem\reactive\Lab3>npm install --global @angular/cli@13.0.3
npm WARN deprecated @npmcli/move-file@1.1.2: This functionality has been moved to @npmcli/fs
npm WARN deprecated source-map-codec@1.4.8: Please use @jridgewell/source-map-codec instead

changed 183 packages in 26s

23 packages are looking for funding
  run 'npm fund' for details
```

Рисунок 1.1. Встановлення необхідних пакетів.

Ініціалізація проекту

Скористаємося командою «ng new todo --routing false --style css --skip-git --skip-tests», аби створити новий порожній проект зі стандартною структурою та файлами конфігурації:

```
C:\WINDOWS\system32\cmd. x + v
Node.js version v21.0.0 detected.
Odd numbered Node.js versions will not enter LTS status and should not be used for production. For more information, please see https://nodejs.org/en/about/releases/.
CREATE todo/angular.json (3588 bytes)
CREATE todo/package.json (1068 bytes)
CREATE todo/README.md (1050 bytes)
CREATE todo/tsconfig.json (863 bytes)
CREATE todo/.editorconfig (274 bytes)
CREATE todo/.gitignore (620 bytes)
CREATE todo/.browserslistrc (600 bytes)
CREATE todo/karma.conf.js (1421 bytes)
CREATE todo/tsconfig.app.json (287 bytes)
CREATE todo/tsconfig.spec.json (333 bytes)
CREATE todo/src/favicon.ico (948 bytes)
CREATE todo/src/index.html (290 bytes)
CREATE todo/src/main.ts (372 bytes)
CREATE todo/src/polyfills.ts (2338 bytes)
CREATE todo/src/styles.css (80 bytes)
CREATE todo/src/test.ts (745 bytes)
CREATE todo/src/assets/.gitkeep (0 bytes)
CREATE todo/src/environments/environment.prod.ts (51 bytes)
CREATE todo/src/environments/environment.ts (658 bytes)
CREATE todo/src/app/app.module.ts (314 bytes)
CREATE todo/src/app/app.component.html (23332 bytes)
CREATE todo/src/app/app.component.ts (208 bytes)
CREATE todo/src/app/app.component.css (0 bytes)
✓ Packages installed successfully.
```

Рисунок 1.2. Результат виконання команди ініціалізації проекту.

Тепер відкриємо середовище розробки, а саме JB Rider, і переглянемо структуру новоствореного проекту:

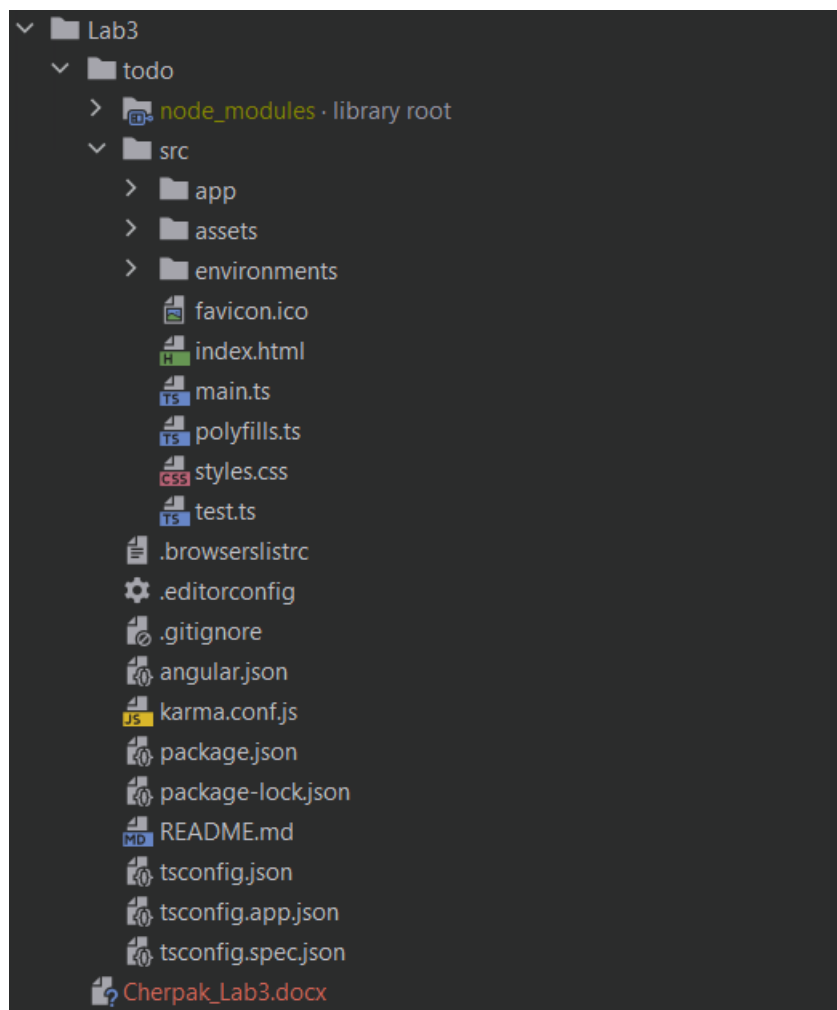


Рисунок 1.3. Структура новоствореного проекту.

Виконаємо у терміналі з кореневої директорії проекту команду «ng serve --open», аби побудувати, локально розгорнути та одразу відкрити у браузері наш проект:

```
PS D:\Education\7 sem\reactive\Lab3\todo> ng serve --open
Node.js version v21.0.0 detected.
Odd numbered Node.js versions will not enter LTS status and should not be used for production. For more information, see https://node.dev/en/adopting-new-versions/.
✓ Browser application bundle generation complete.

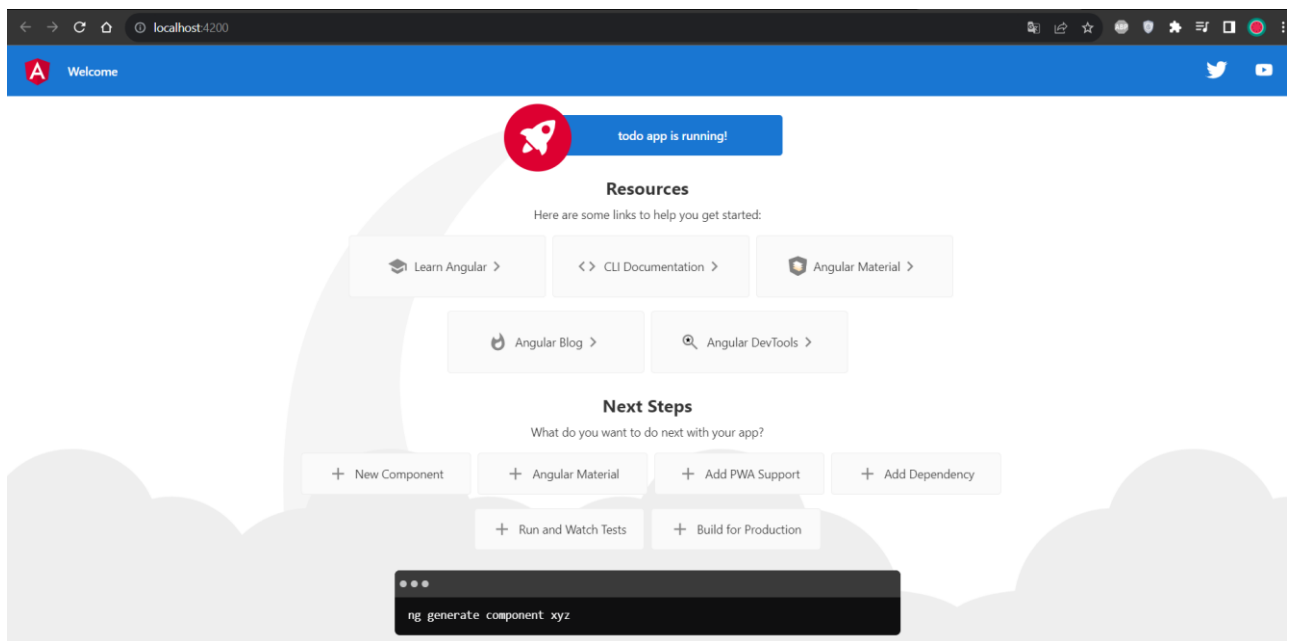
Initial Chunk Files | Names          | Size
vendor.js           | vendor         | 1.74 MB
polyfills.js        | polyfills      | 409.31 kB
styles.css, styles.js | styles        | 285.53 kB
main.js             | main           | 51.63 kB
runtime.js          | runtime        | 6.84 kB

| Initial Total | 2.48 MB

Build at: 2023-10-19T23:17:39.519Z - Hash: 9a16e3c228e990e6 - Time: 2903ms

** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **

✓ Compiled successfully.
```



Рисунки 1.4-1.5. Результат запуску нового проекту.

Створення базових класів.

Додамо файл todoItem.ts:

```
export class TodoItem {
  constructor(
    public task: string,
    public complete: boolean = false) {
```

```
// оголошення не потрібні  
}  
}
```

Потім створимо клас todoList.ts:

```
import { TodoItem } from "../todoItem";  
export class TodoList {  
  constructor (public user: string, private todoItems:  
TodoItem[] = [])  
  {  
    // no statements required  
  }  
  get items(): readonly TodoItem[] {  
    return this.todoItems;  
  }  
  addItem(task: string) {  
    this.todoItems.push(new TodoItem(task));  
  }  
}
```

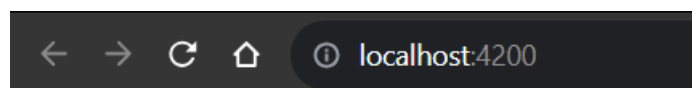
А також внесемо невеликі зміни у app.component.ts та app.component.html
app.component.ts:

```
import { Component } from '@angular/core';  
import { TodoList } from "../todoList";  
import { TodoItem } from "../todoItem";  
  
@Component({  
  selector: 'app-root',  
  templateUrl: '../app.component.html',  
  styleUrls: ['../app.component.css']  
})
```

```
export class AppComponent {
  private list = new TodoList("Андрій", [
    new TodoItem("Зробити пробіжку", true),
    new TodoItem("Купити квіти"),
    new TodoItem("Забрати квитки"),
  ]);
  get username(): string {
    return this.list.user;
  }
  get itemCount(): number {
    return this.list.items
      .filter(item => !item.complete).length;
  }
}
```

app.component.html:

```
<h3>
  {{ username }}: список справ
  <h5>{{ itemCount }} елементів</h5>
</h3>
```



Андрій: список справ

2 елементів

Рисунок 1.6. Результат запуску оновленого застосунку.

Додавання стилів до застосунку.

Для початку, завантажимо пакет angular/materials:

```
PS D:\Education\7 sem\reactive\Lab3\todo> ng add @angular/material@13.0.2 --defaults
Node.js version v21.0.0 detected.
Odd numbered Node.js versions will not enter LTS status and should not be used for production
Skipping installation: Package already installed
UPDATE package.json (1041 bytes)
√ Packages installed successfully.
PS D:\Education\7 sem\reactive\Lab3\todo> █
```

Рисунок 1.7. Встановлення пакету angular/materials

Тепер додамо нові залежності у за стосунок:

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-
browser';
import { AppComponent } from './app.component';
import { BrowserAnimationsModule } from
 '@angular/platform-browser/animations';
import { FormsModule } from '@angular/forms'
import { MatButtonModule } from
 '@angular/material/button';
import { MatToolbarModule } from
 '@angular/material/toolbar';
import { MatIconModule } from '@angular/material/icon';
import { MatBadgeModule } from '@angular/material/badge';
import { MatTableModule } from '@angular/material/table';
import { MatCheckboxModule } from
 '@angular/material/checkbox';
import { MatFormFieldModule } from
 '@angular/material/form-field';
import { MatInputModule } from '@angular/material/input';
import { MatSlideToggleModule } from
 '@angular/material/slide-toggle';

@NgModule({
  declarations: [
```



```

    AppComponent
  ],
  imports: [
    BrowserModule,
    BrowserAnimationsModule,
    FormsModule,
    MatButtonModule, MatToolbarModule, MatIconModule,
MatBadgeModule,
    MatTableModule, MatCheckboxModule,
MatFormFieldModule, MatInputModule,
    MatSlideToggleModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }

```

Тепер змінимо код сторінки:

```

<mat-toolbar color="primary" class="mat-elevation-z3">
  <span>{{ username }}: список справ</span>
  <mat-icon matBadge="{{ itemCount }}"
matBadgeColor="accent">checklist</mat-icon>
</mat-toolbar>

```

Додамо також стилі:

```

spacer { flex: 1 1 auto }

```

І ще раз внесемо зміни у код сторінки:

```

<mat-toolbar color="primary" class="mat-elevation-z3">
  <span class="spacer"></span>
  <span> {{ username }}'s To Do List </span>
  <span class="spacer"></span>
  <mat-icon matBadge="{{ itemCount }}"

```

```
matBadgeColor="accent"> checklist </mat-icon>
</mat-toolbar>
```

Відтепер наша сторінка виглядатиме так:

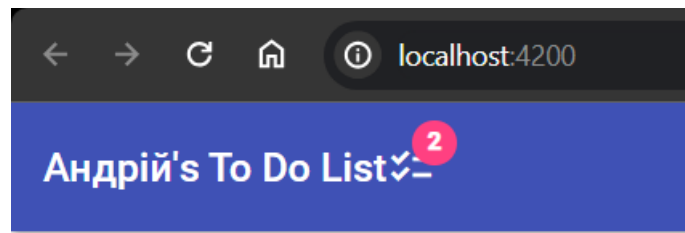


Рисунок 1.8. Результат додавання стилів.

Відображення списку справ.

Тепер внесемо зміни у код додатка, або відображати список справ:

Змінимо код файлу `app.component.ts`:

```
import { Component } from '@angular/core';
import { TodoList } from './todoList';
import { TodoItem } from './todoItem';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  private list = new TodoList("Андрій", [
    new TodoItem("Зробити пробіжку", true),
    new TodoItem("Купити квіти"),
    new TodoItem("Забрати квитки"),
  ]);

  get username(): string {
    return this.list.user;
  }

  get itemCount(): number {

```

```

    return this.list.items
        .filter(item => !item.complete).length;
}
get items(): readonly TodoItem[] {
    return this.list.items;
}
}

```

Також змінимо сам шаблон app.component.html:

```

<mat-toolbar color="primary" class="mat-elevation-z3">
  <span class="spacer"></span>
  <span>{{ username }}'s To Do List </span>
  <span class="spacer"></span>
  <mat-icon matBadge="{{ itemCount }}"
matBadgeColor="accent"> checklist </mat-icon>
</mat-toolbar>

<div class="tableContainer">
  <table mat-table [dataSource]="items" class="mat-
elevation-z3 fullWidth">

    <ng-container matColumnDef="id">
      <th mat-header-cell *matHeaderCellDef>№</th>
      <td mat-cell *matCellDef="let i = index"> {{ i + 1
  }} </td>
    </ng-container>

    <ng-container matColumnDef="task">
      <th mat-header-cell *matHeaderCellDef>Task</th>
      <td mat-cell *matCellDef="let item"> {{ item.task
  }} </td>
    </ng-container>

    <ng-container matColumnDef="done">
      <th mat-header-cell *matHeaderCellDef>Done</th>
      <td mat-cell *matCellDef="let item"> {{

```

```

item.complete }} </td>
    </ng-container>
    <tr mat-header-row *matHeaderRowDef="['id', 'task',
'done']"></tr>
    <tr mat-row *matRowDef="let row; columns: ['id',
'task', 'done'];"></tr>
  </table>
</div>

```

І додамо додаткові стилі до app.component.css:

```

.spacer { flex: 1 1 auto }
.tableContainer { padding: 15px }
.fullWidth { width: 100% }

```

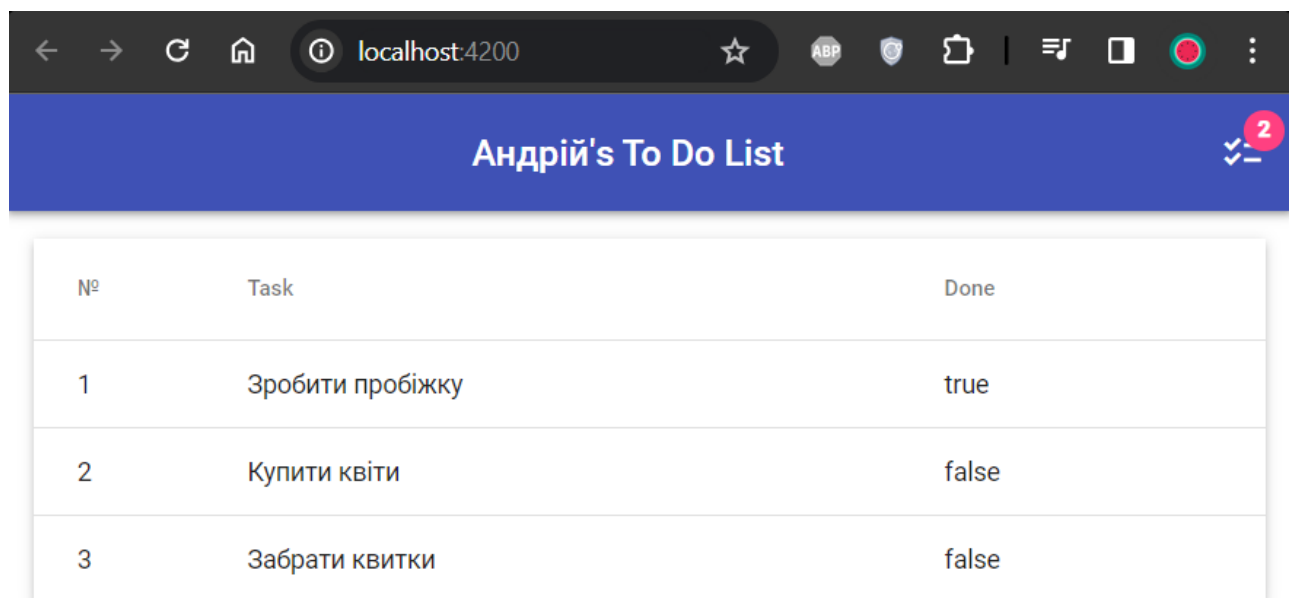


Рисунок 1.9. Результат відображення списку завдань.

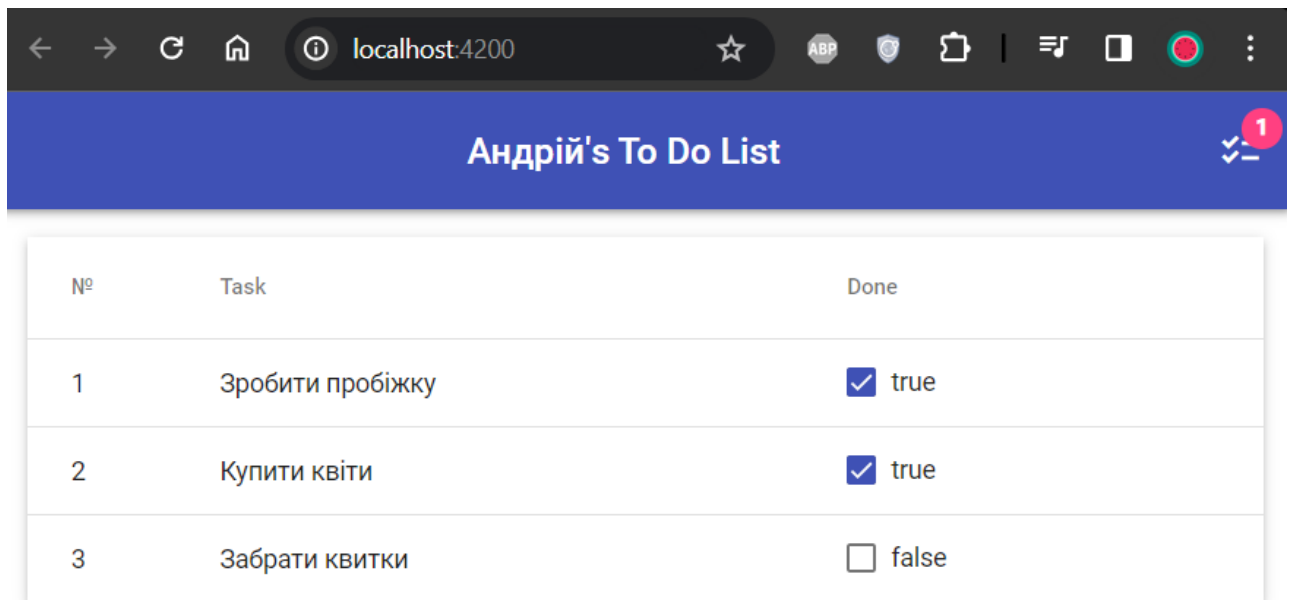
Додавання двосторонньої прив'язки до чекбоксів.

Ми хочемо відображати статус завдання у вигляді галочки, а також мати можливість позначати завдання як виконане. Для цього змінимо файл app.component.html:


```

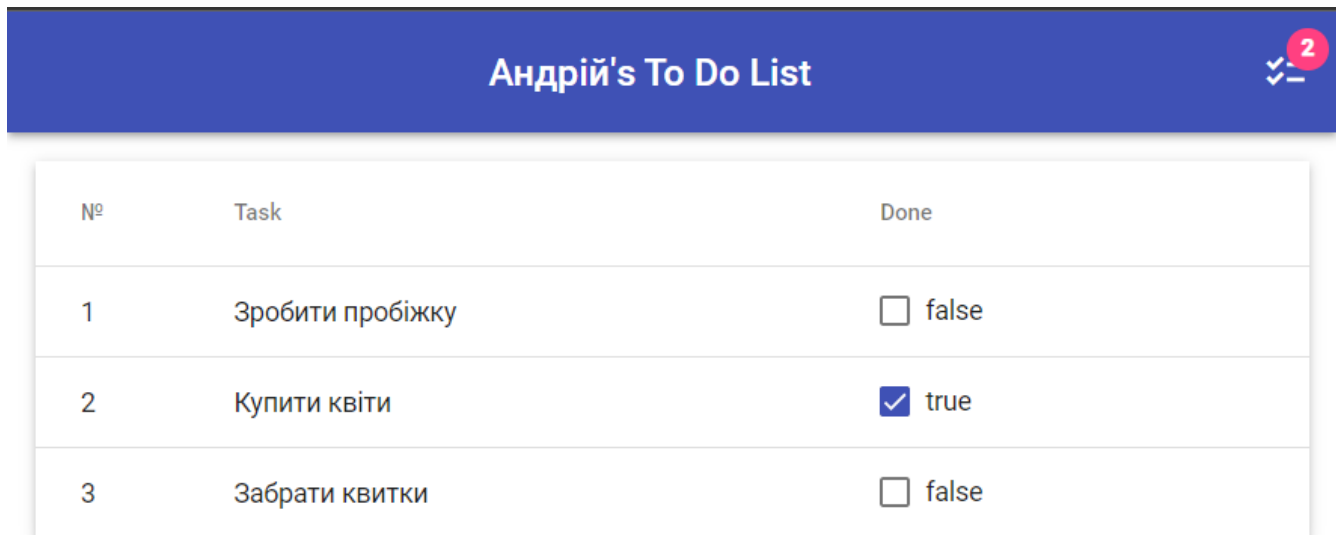
    <tr mat-row *matRowDef="let row; columns: ['id',
'task', 'done'];"></tr>
  </table>
</div>

```



Андрій's To Do List 1

№	Task	Done
1	Зробити пробіжку	<input checked="" type="checkbox"/> true
2	Купити квіти	<input checked="" type="checkbox"/> true
3	Забрати квитки	<input type="checkbox"/> false



Андрій's To Do List 2

№	Task	Done
1	Зробити пробіжку	<input type="checkbox"/> false
2	Купити квіти	<input checked="" type="checkbox"/> true
3	Забрати квитки	<input type="checkbox"/> false

Рисунки 1.10-1.11. Результати додавання чек боксів з двосторонньою прив'язкою.

Фільтрування завершених завдань.

. Оскільки ми не хочемо відображати неактуальні завдання, додамо у файл `app.components.ts` фільтрування:

```

import { Component } from '@angular/core';
import { TodoList } from "../todoList";
import { TodoItem } from "../todoItem";

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  private list = new TodoList("Андрій", [
    new TodoItem("Зробити пробіжку", true),
    new TodoItem("Купити квіти"),
    new TodoItem("Забрати квитки"),
  ]);

  get username(): string {
    return this.list.user;
  }

  get itemCount(): number {
    return this.list.items
      .filter(item => !item.complete).length;
  }

  get items(): readonly TodoItem[] {
    return this.list.items.filter(item =>
!item.complete);
  }
}

```

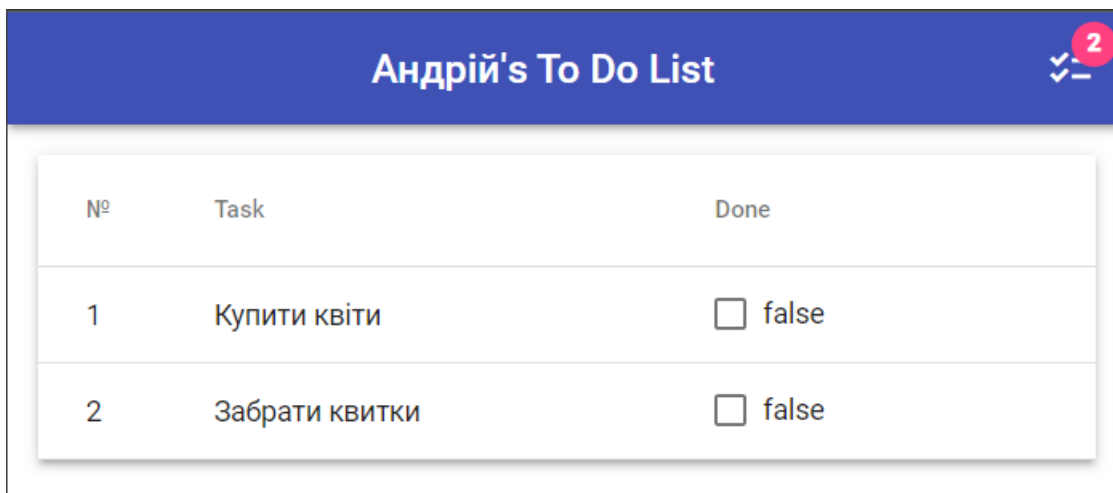


Рисунок 1.12. Відображення лише незавершених завдань.

Додавання нових завдань.

Очевидно, користувач повинен мати змогу додавати нові завдання. Для цього створимо нову функцію у файлі `app.component.ts`:

```
import { Component } from '@angular/core';
import { TodoList } from "../todoList";
import { TodoItem } from "../todoItem";

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  private list = new TodoList("Андрій", [
    new TodoItem("Зробити пробіжку", true),
    new TodoItem("Купити квіти"),
    new TodoItem("Забрати квитки"),
  ]);

  get username(): string {
    return this.list.user;
  }
}
```



```

    get itemCount(): number {
        return this.list.items
            .filter(item => !item.complete).length;
    }

    get items(): readonly TodoItem[] {
        return this.list.items.filter(item =>
!item.complete);
    }

    addItem(newItem: string) {
        if (newItem !== "") {
            this.list.addItem(newItem);
        }
    }
}

```

Тепер додамо поле вводу та кнопку до app.component.html:

```

<mat-toolbar color="primary" class="mat-elevation-z3">
    <span class="spacer"></span>
    <span>{{ username }}'s To Do List </span>
    <span class="spacer"></span>
    <mat-icon matBadge="{{ itemCount }}"
matBadgeColor="accent"> checklist </mat-icon>
</mat-toolbar>

<div class="inputContainer">
    <mat-form-field class="fullWidth">
        <mat-label style="padding-left: 5px;">Нова
справа</mat-label>

        <input matInput placeholder="Enter to-do description"
#todoText>

        <button matSuffix mat-raised-button color="accent"
class="addButton" (click)="addItem(todoText.value);
todoText.value = ''">

```

```

        Додати
    </button>
</mat-form-field></div>
<div class="tableContainer">
    <table mat-table [dataSource]="items" class="mat-
elevation-z3 fullWidth">
        <ng-container matColumnDef="id">
            <th mat-header-cell *matHeaderCellDef>№</th>
            <td mat-cell *matCellDef="let i = index"> {{ i + 1
}} </td>
        </ng-container>
        <ng-container matColumnDef="task">
            <th mat-header-cell *matHeaderCellDef>Task</th>
            <td mat-cell *matCellDef="let item"> {{ item.task
}} </td>
        </ng-container>
        <ng-container matColumnDef="done">
            <th mat-header-cell *matHeaderCellDef>Done</th>
            <td mat-cell *matCellDef="let item">
                <mat-checkbox [(ngModel)]="item.complete"
color="primary">
                    {{ item.complete }}
                </mat-checkbox>
            </td>
        </ng-container>
        <tr mat-header-row *matHeaderRowDef="['id', 'task',
'done']"></tr>
        <tr mat-row *matRowDef="let row; columns: ['id',
'task', 'done'];"></tr>
    </table>
</div>

```

Також пропишемо стилі у app.component.css для нових елементів:

```
.spacer { flex: 1 1 auto }  
.tableContainer { padding: 15px }  
.fullWidth { width: 100% }  
.inputContainer { margin: 15px 15px 5px }  
.addButton { margin: 5px }
```

Відтепер після введення тексту завдання у поле та натисканні кнопки у списку з'являтиметься нова позиція:

Андрій's To Do List

4

Нова справа

Захистити лабораторну

Додати

№	Task	Done
1	Купити квіти	<input type="checkbox"/> false
2	Забрати квитки	<input type="checkbox"/> false
3	Зробити звіт з лабораторної	<input type="checkbox"/> false
4	розгорнути додаток на firebase	<input type="checkbox"/> false

№	Task	Done
1	Купити квіти	<input type="checkbox"/> false
2	Забрати квитки	<input type="checkbox"/> false
3	Зробити звіт з лабораторної	<input type="checkbox"/> false
4	розгорнути додаток на firebase	<input type="checkbox"/> false
5	Захистити лабораторну	<input type="checkbox"/> false

Рисунки 1.13-1.14. Додавання нових завдань.

Відображення виконаних завдань.

Звісно, іноді ми хочемо переглянути і раніше виконані завдання. Додамо на сторінку перемикач, який ховатиме або показуватиме закриті позиції. Також приберемо відображення true/false біля галочки:

app.component.html:

```
<mat-toolbar color="primary" class="mat-elevation-z3">
  <span class="spacer"></span>
  <span>{{ username }}'s To Do List </span>
  <span class="spacer"></span>
  <mat-icon matBadge="{{ itemCount }}"
matBadgeColor="accent"> checklist </mat-icon>
</mat-toolbar>
```

```

<div class="inputContainer">
  <mat-form-field class="fullWidth">
    <mat-label style="padding-left: 5px;">Нова
справа</mat-label>
    <input matInput placeholder="Enter to-do description"
#todoText>
    <button matSuffix mat-raised-button color="accent"
class="addButton" (click)="addItem(todoText.value);
todoText.value = ''">
      Додати
    </button>
  </mat-form-field></div>
<div class="tableContainer">
  <table mat-table [dataSource]="items" class="mat-
elevation-z3 fullWidth">
    <ng-container matColumnDef="id">
      <th mat-header-cell *matHeaderCellDef>№</th>
      <td mat-cell *matCellDef="let i = index"> {{ i + 1
}} </td>
    </ng-container>
    <ng-container matColumnDef="task">
      <th mat-header-cell *matHeaderCellDef>Task</th>
      <td mat-cell *matCellDef="let item"> {{ item.task
}} </td>
    </ng-container>
    <ng-container matColumnDef="done">
      <th mat-header-cell *matHeaderCellDef>Done</th>
      <td mat-cell *matCellDef="let item">
        <mat-checkbox [(ngModel)]="item.complete"
color="primary">
          <!-- {{ item.complete }} -->

```

```

        </mat-checkbox>
      </td>
    </ng-container>
    <tr mat-header-row *matHeaderRowDef="['id', 'task',
'done']"></tr>
    <tr mat-row *matRowDef="let row; columns: ['id',
'task', 'done'];"></tr>
  </table>
</div>
<div class="toggleContainer">
  <span class="spacer"></span>
  <mat-slide-toggle [(ngModel)]="showComplete">
    Show Completed Items
  </mat-slide-toggle>
  <span class="spacer"></span>
</div>

```

У app.component.ts введемо нову модель showComplete та трішки модифікуємо умову фільтрації:

```

import { Component } from '@angular/core';
import { TodoList } from './todoList';
import { TodoItem } from './todoItem';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  private list = new TodoList("Андрій", [
    new TodoItem("Зробити пробіжку", true),

```

```

    new TodoItem("Купити квіти"),
    new TodoItem("Забрати квитки"),
  ]);
  get username(): string {
    return this.list.user;
  }
  get itemCount(): number {
    return this.list.items
      .filter(item => !item.complete).length;
  }
  get items(): readonly TodoItem[] {
    return this.list.items.filter(item =>
this.showComplete || !item.complete);
  }
  addItem(newItem: string) {
    if (newItem !== "") {
      this.list.addItem(newItem);
    }
  }
  showComplete: boolean = false;
}

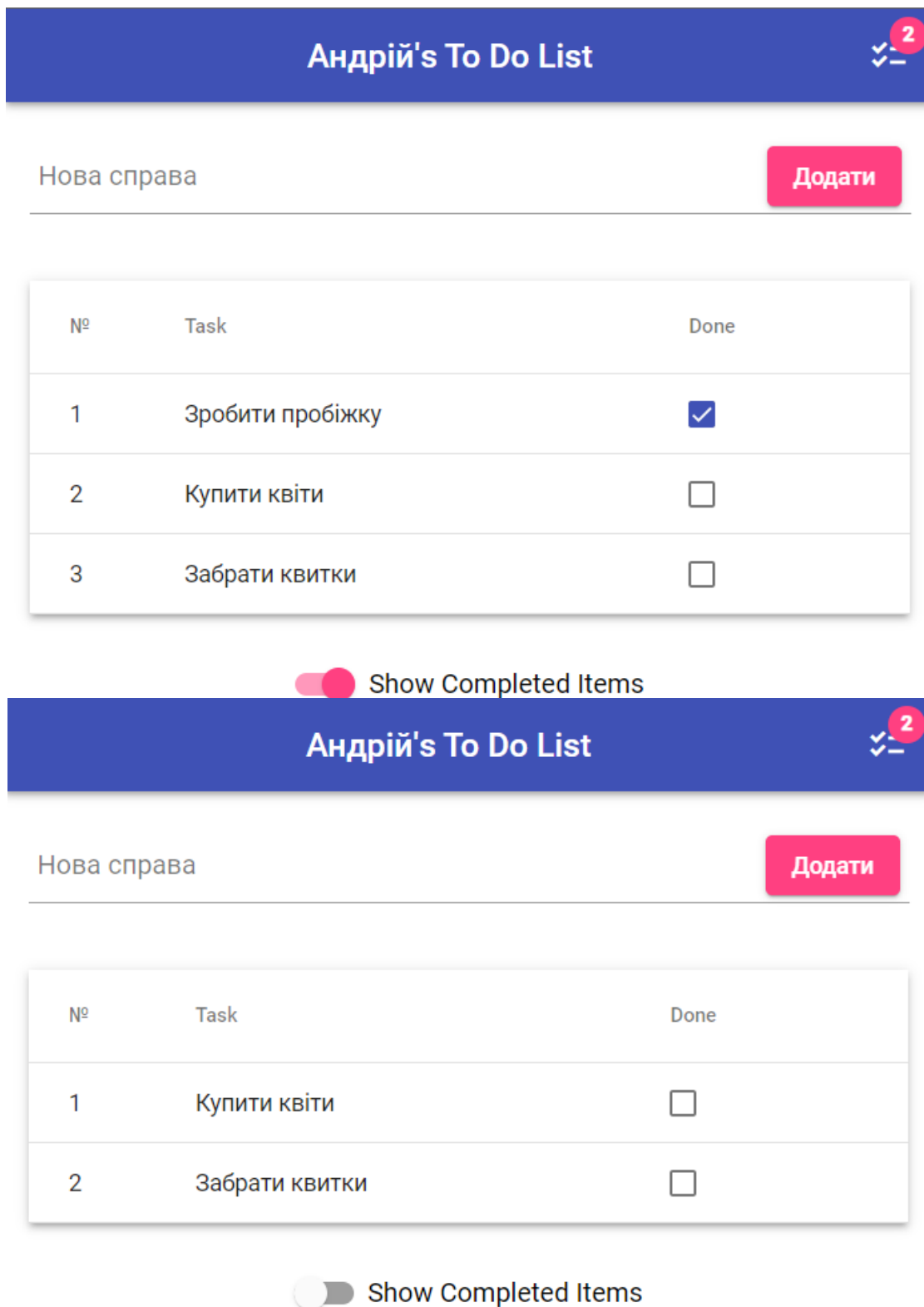
```

Також додамо у app.component.css стилі для перемикача:

```

.spacer { flex: 1 1 auto }
.tableContainer { padding: 15px }
.fullWidth { width: 100% }
.inputContainer { margin: 15px 15px 5px }
.addButton { margin: 5px }
.toggleContainer { margin: 15px; display: flex }

```



Рисунки 1.15-1.16. Приховування та відображення виконаних завдань
Хостинг.

Розгорнемо наш за стосунок на платформі firebase з допомогою тих же команд, що і для минулих лабораторних.

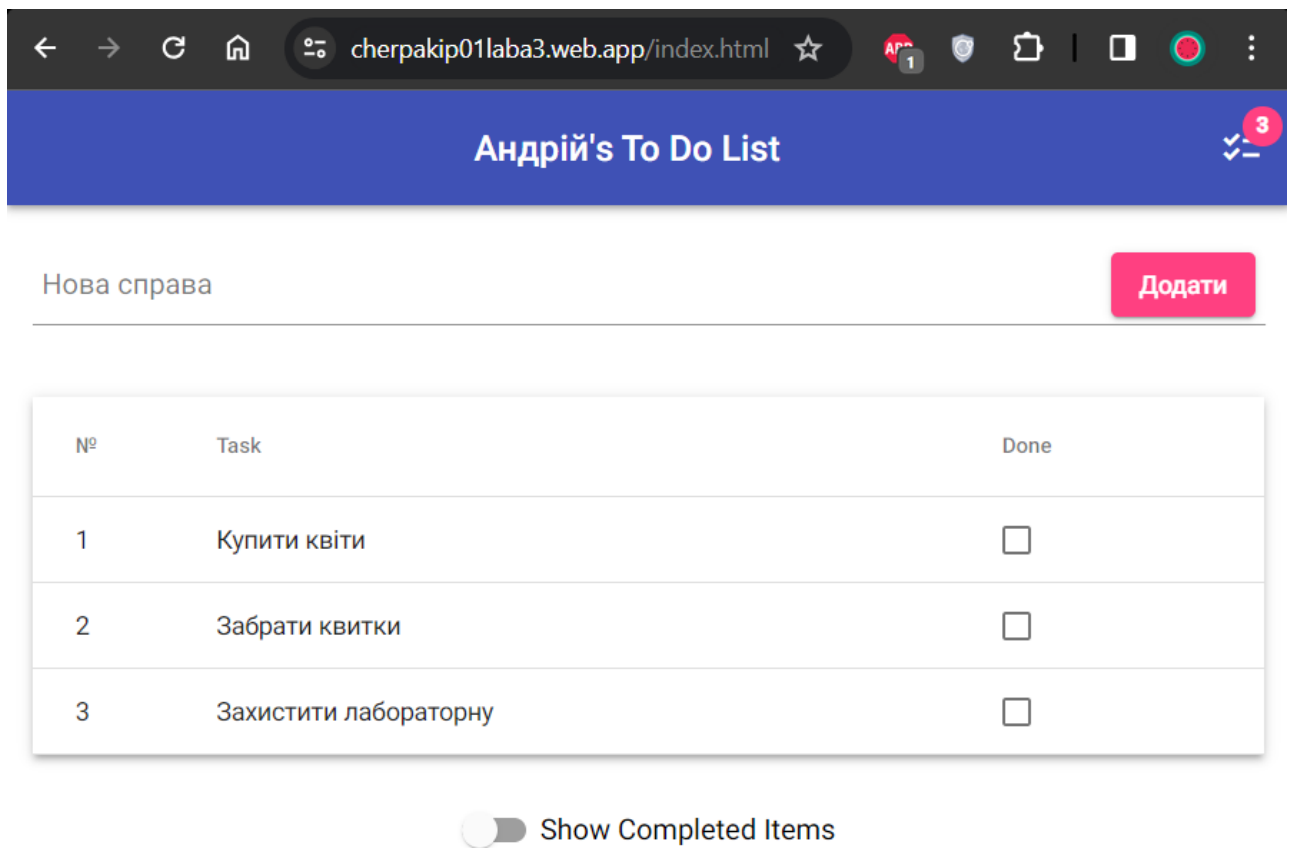


Рисунок 1.17. Розгорнутий на firebase застосунок

Посилання на додаток: cherpakip01laba3.web.app

Висновок:

Під час виконання комп'ютерного практикуму ми створили застосунок «Список завдань» засобами Angular з використанням angular Material та розгорнули його на платформі firebase.

Список використаних джерел:

1. Component Lifecycle: <https://angular.io/guide/lifecycle-hooks>