**ДОДАТОК**

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ

СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт по лабораторній роботі No 3

Створення проекту «Todo».

З дисципліни: «Реактивне програмування»

Студент: Трофимов Данило Олегович

Група: ІП-02

Дата захисту роботи:

Викладач: доц. Полупан Юлія Вікторівна

Захищено з оцінкою:

Київ, 2023

# **ЗМІСТ**

[**ЗМІСТ** 2](#_Toc148684176)

[**Загальний хід роботи:** 3](#_Toc148684178)

[**1)Модель даних у проєкті «Todo»** 3](#_Toc148684179)

[**2)Пакет Angular Material: Призначення та використання.** 4](#_Toc148684180)

[**3)Двосторонні прив’язки даних: призначення, використання** 6](#_Toc148684181)

[**4)Функція filter у JavaScript: призначення, застосування, приклади** 7](#_Toc148684182)

[**5)** **Посилання на розгорнутий додаток** 8](#_Toc148684183)

[**Висновки** 9](#_Toc148684184)

[**Список літератури** 10](#_Toc148684185)

# **Загальний хід роботи:**

1. Згідно з інструкціями ініціалізовано angular проєкт з назвою Todo
2. Було опрацьовано завдання по створенню проєкту Todo
3. Було розгорнуто дотаток на платформі Firebase

# **Модель даних у проєкті «Todo»**

Модель даних у проекті "Todo" складається з двох класів: TodoItem та TodoList. Давайте розглянемо кожен з них окремо.

1. Клас TodoItem:

Цей клас відповідає за представлення окремого завдання (точніше, елементу списку завдань) у вашому додатку. Він має наступні властивості:

* + task (string): Представляє текстовий опис завдання.
  + complete (boolean): Показує, чи завдання вже виконано. За замовчуванням, це значення встановлено на false.

Конструктор класу дозволяє створити новий об'єкт TodoItem з текстовим описом завдання та опціональним прапорцем complete, який за замовчуванням встановлюється на false.

1. Клас TodoList:

Цей клас відповідає за управління списком завдань користувача. Він має наступні властивості та методи:

* + user (string): Представляє ім'я користувача, якому належить список завдань.
  + todoItems (TodoItem[]): Це масив екземплярів класу TodoItem, що представляють список завдань.

Конструктор класу дозволяє створити новий об'єкт TodoList з ім'ям користувача та опціональним масивом завдань.

* + items: Це властивість (getter), яка повертає список завдань в режимі тільки для читання (readonly). Вона дозволяє іншим частинам коду отримувати доступ до списку завдань, але не дозволяє його змінювати.
  + addItem(task: string): Цей метод додає нове завдання до списку завдань. Він створює новий об'єкт TodoItem з переданим текстовим описом завдання та додає його до масиву todoItems.

Ця модель даних дозволяє створювати і управляти списком завдань для конкретного користувача. Клас TodoList містить метод addItem, який дозволяє додавати нові завдання до списку, а властивість items дозволяє звертатися до списку завдань для читання.

# **Пакет Angular Material: Призначення та використання.**

Angular Material - це бібліотека для Angular, яка надає готові інтерфейсні компоненти та інструменти для створення стильних та функціональних веб-додатків.

Основні компоненти та модулі, які ми імпортуємо в AppModule:

1. MatButtonModule: Цей модуль містить компонент MatButton, який дозволяє створювати кнопки у додатку.
2. MatToolbarModule: Цей модуль містить компонент MatToolbar, який дозволяє створювати панелі інструментів у верхній частині додатку.
3. MatIconModule: Цей модуль містить компонент MatIcon, який дозволяє вставляти значки у додатку.
4. MatBadgeModule: Цей модуль містить компонент MatBadge, який дозволяє додавати бейджі до інших компонентів для позначення кількості повідомлень або сповіщень.
5. MatTableModule: Цей модуль містить компоненти для створення таблиць у додатку, такі як MatTable.
6. MatCheckboxModule: Цей модуль містить компонент MatCheckbox, який дозволяє створювати чекбокси.
7. MatFormFieldModule: Цей модуль містить компоненти для створення форм та текстових полів, такі як MatFormField і MatInputModule.
8. MatInputModule: Цей модуль містить компонент MatInput, який дозволяє створювати текстові поля.
9. MatSlideToggleModule: Цей модуль містить компонент MatSlideToggle, який дозволяє створювати перемикачі.

Ці компоненти допомагають вам створити стильний та інтерактивний інтерфейс для Angular додатку. Ви можете використовувати їх у шаблонах та логіці вашого компоненту для розробки функціональності нашого додатку.

Давайте розглянемо кожну частину коду та призначення використаних компонентів у файлі app.component.html:

1. <mat-toolbar>: Цей компонент створює панель інструментів (toolbar) в верхній частині вашого додатку. Властивість color="primary" встановлює колір панелі, а class="mat-elevation-z3" надає тінь для підняття панелі.
2. <mat-icon>: Цей компонент додає значок "checklist" до панелі інструментів. Властивість matBadge використовується для відображення кількості пунктів в To-Do списку як бейдж, а matBadgeColor="accent" встановлює колір бейджа.
3. <mat-form-field>: Цей компонент створює обгортку для введення тексту. Властивість class="fullWidth" розширює його на всю ширину контейнера.
4. <mat-label>: Властивість mat-label встановлює підпис для поля введення.
5. <input matInput>: Цей елемент вводу тексту створює текстове поле для введення опису завдання. #todoText - це локальна посилання на поле вводу, що використовується для отримання значення поля.
6. <button matSuffix>: Цей кнопка розміщена після поле вводу тексту і використовується для додавання завдань до списку To-Do. Властивості mat-raised-button і color="accent" встановлюють стиль кнопки.
7. <table mat-table>: Цей компонент створює таблицю для відображення списку завдань (To-Do List). Властивість [dataSource]="items" вказує на джерело даних для таблиці за допомогою односторонньої прив’язки даних.
8. <mat-checkbox>: Цей компонент створює чекбокс для позначення завдань як виконаних або невиконаних.
9. <mat-slide-toggle>: Цей компонент створює перемикач для включення або виключення відображення завершених завдань в списку. [(ngModel)]="showComplete" - використовується двостороння прив’язка даних для зв'язку перемикача зі змінною showComplete у вашому компоненті.

У цьому коді Angular Material використовується для створення стильного та інтерактивного інтерфейсу для нашого To-Do List додатку. Завдяки використанню Angular Material компонентів, ми можемо легко додавати функціональність та поліпшувати зовнішній вигляд нашого додатку.

# **Двосторонні прив’язки даних: призначення, використання**

Двосторонні прив'язки даних (two-way data binding) в Angular дозволяють автоматично оновлювати як вигляд (view) елементу у шаблоні, так і модель даних в компоненті при зміні одного з них. В Angular, двосторонні прив'язки даних використовуються за допомогою директиви ngModel.

У нашому коді є приклад використання двосторонньої прив'язки даних для компонента <mat-slide-toggle>. Давайте розглянемо цей приклад:



1. [(ngModel)]="showComplete": Ми встановюємо двосторонню прив'язку для компонента <mat-slide-toggle>. ngModel - це директива, яка дозволяє створити двосторонню прив'язку між властивістю showComplete вашого компоненту і значенням перемикача.
2. showComplete: Це властивість нашого компоненту, яка відповідає за стан перемикача (включено або виключено).

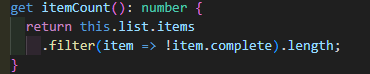
Коли користувач перемикає mat-slide-toggle, значення showComplete автоматично оновлюється в нашому компоненті. Це означає, що якщо перемикач включений, showComplete буде true, і навпаки, якщо він виключений, showComplete буде false. Навпаки, якщо ви змінюєте showComplete у вашому компоненті, то відображення перемикача в шаблоні також оновиться відповідно до нового значення. Це дозволяє забезпечити синхронізацію між даними в моделі та їх відображенням у вигляді на стороні користувача.

# **Функція filter у JavaScript: призначення, застосування, приклади**

Функція filter в JavaScript призначена для фільтрації масиву на основі заданої умови. Вона створює новий масив, який містить тільки ті елементи з вихідного масиву, для яких умова виконується (повертає true).

У вашому коді Angular, функція filter використовується для фільтрації списку завдань (list.items) залежно від умови, яку ви вказуєте. Ось приклади використання функції filter в вашому коді:

**get itemCount()**: Цей метод повертає кількість невиконаних завдань. Він використовує функцію filter для створення нового масиву, який містить тільки невиконані завдання (де item.complete дорівнює false), і обчислює довжину цього масиву.



**get items()**: Цей метод повертає масив завдань для відображення в вашому списку. Він використовує функцію filter, щоб вибрати тільки ті завдання, які повинні бути відображені в залежності від значення showComplete. Якщо showComplete встановлено в true, то усі завдання включаються; в іншому випадку включаються тільки невиконані завдання.



Функція filter дозволяє ефективно фільтрувати дані у масиві відповідно до заданих критеріїв. У вашому коді вона використовується для підрахунку кількості невиконаних завдань та відображення завдань залежно від значення showComplete.

# **Посилання на розгорнутий додаток**

Додаток розміщений за адресою: <https://trofymovip02laba3.web.app/>

# **Висновки**

В даному завданні був розглянутий додаток «Todo». У ньому ми розглянули роботу з Angular material, реалізували просту модель даних для цього додатку, опрацювали двосторонню прив’язку даних на реальному проєкті та застосували функцію filter для фільтрування масиву даних.

Загалом, Angular Material використовується для полегшення створення стильних та функціональних інтерфейсів у нашому Angular додатку, а двостороння прив'язка даних та функція filter допомагають досягти більшого функціонального рівня та взаємодії з користувачем.

# **Список літератури**

* + 1. Introduction to the Angular docs [Електронний ресурс] – Режим доступу до ресурсу: <https://angular.io/docs>.
    2. Angular. Angular tutorials [Електронний ресурс] / Angular – Режим доступу до ресурсу: <https://angular.io/tutorial>.
    3. The Angular Book [Електронний ресурс] – Режим доступу до ресурсу: <https://angular-book.dev/>.
    4. Daniel Schmitz. Angular 14 from Scratch / Daniel Schmitz., 2022.
    5. Denys Vuika. Developing with Angular / Denys Vuika., 2018.