**Файл: task.dart**

Клас **Task:**

**Опис:** Цей клас представляє собою модель задачі.

Властивості:

1. **title** **(String):** Назва задачі.
2. **description** **(String):** Рядкове значення, що описує задачу.
3. **dueDate (DateTime):** Дата, яка позначає день закінчення задачі.
4. **category** **(String):** Рядкове значення - категорія задачі (за замовчуванням - **"Other"**).
5. **isCompleted** **(bool):** Логічне значення, що позначає, чи виконана задача (за замовчуванням - false).

Методи:

1. **factory Task.fromJson(Map<String, dynamic> json)**: Метод приймає вхідні дані у вигляді об'єкта **Map<String, dynamic> json**, який містить дані про завдання у форматі JSON та повертає новий екземпляр класу **Task**, зібраний з даних об'єкта JSON.
2. **Map<String, dynamic> toJson()**: Метод перетворює об'єкт класу **Task** і повертає створений об'єкт JSON, який представляє дані про завдання у форматі JSON.

**Файл: task\_list\_provider.dart**

Клас **TaskListProvider:**

**Опис:** Цей клас відповідає за управління списком задач, зберігання та завантаження їх у локальне сховище.

**Властивості:**

1. **tasks** (List<Task>): Активний список об'єктів задач, які зберігаються у сховищі.

**Методи:**

1. **void loadTasks() async**: Не приймає параметрів. Асинхронно завантажує задачі з локального сховища (SharedPreferences) та оновлює список задач.
2. **void saveTasks() async**: Не приймає параметрів. Асинхронно зберігає поточний список задач у локальному сховищі.
3. **void addTask(Task task, [String sortBy = 'By Date (Newest First)'])**: Метод приймає параметр **task**, який представляє нове завдання для додавання та необов'язковий параметр **sortBy,** який вказує критерій сортування списку після додавання завдання до списку.
4. **void updateTask(int index, Task task, [String sortBy = 'By Date (Newest First)'])**: Метод приймає числовий параметр **index**, який вказує на позицію завдання у списку, яке потрібно оновити, об'єкт класу **Task** та необов'язковий параметр **sortBy** **,** який вказує критерій сортування списку списку після оновлення завдання.
5. **void deleteTask(int index)**: Метод приймає параметр **index**, який вказує на позицію завдання у списку, яке потрібно видалити.
6. **void sortTasks(String sortBy)**: Метод приймає параметр **sortBy**, за яким сортується список задач.

**Файл: task\_item.dart**

Клас **TaskItem:**

**Опис:** Цей клас відповідає за відображення окремої задачі у списку задач.

**Властивості:**

* **final task** **(Task):** Об’єкт класу Task - Задача, яку потрібно відобразити.
* **final onCheckboxChanged** **(Function(bool?)):** Функція, яка викликається при зміні стану завдання (виконане чи невиконане)
* **final onEditPressed** **(Function()):** Функція, яка викликається при натисканні на кнопку редагування задачі.
* **final onDeletePressed** **(Function()):** Функція, яка викликається при натисканні на кнопку видалення задачі.

**Методи:**

1. **Widget build(BuildContext context)**: Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертає віджет, що відображає окрему задачу у списку.
2. **static void showCompletionDialog(BuildContext context, Task task)**: Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет та об’єкт класу Task, що представляє завдання для якого відображається діалогове вікно, яке показує повідомлення про завершення виконання задачі та час, який пройшов з моменту завершення. Є статичним методом.

**Файл: floating\_button.dart**

Клас **FloatingButton:**

**Опис:** Цей клас відповідає за відображення плаваючої кнопки додавання нової задачі.

**Методи:**

1. **Widget build(BuildContext context)**: Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертає віджет, що відображає плаваючу кнопку. При натисканні на кнопку викликається функція **showTaskDialog**.
2. **static void showTaskDialog(BuildContext context, TaskListProvider taskProvider, String \_selectedFilter, [int? index])**: **)**: Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет, об’єкт класу **TaskListProvider**, рядкове значення **\_selectedFilter,** за яким сортуються завдання у списку та необов'язковий параметр **index** – числове значення, яке вказує на індекс завдання у списку, якщо необхідно редагувати існуюче завдання. Даний метод відображає діалогове вікно для додавання або редагування задачі. Є статичним методом.

**Файл: custom\_dropdown.dart**

Клас **CustomDropdown:**

**Опис:** Цей клас відповідає за відображення випадаючого списку.

**Властивості:**

1. **final options (List<T>)**: Список елементів довільного типу, які можна вибрати у випадаючому списку.
2. **final selectedValue (T)**: Елемент, який може приймати будь-який тпи даних - поточно обраний елемент зі списку.
3. **final onChanged (Function(T?))**: Функція, яка викликається при зміні обраного довільного значення.
4. **showBorder (bool)**: Логічне значення, яке вказує, чи слід відображати межу випадаючого списку. Є статичним полем.

**Методи:**

1. **Widget build(BuildContext context)**: Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертає віджет, що відображає редагований випадаючий список.

**Файл: custom\_bottom\_navigation\_bar.dart**

Клас **CustomBottomNavigationBar:**

**Опис:** Цей клас відповідає за відображення нижньої навігаційної панелі з двома кнопками: "Tasks" і "Calendar".

**Властивості:**

1. **final currentIndex (int):** Числове значення - поточний індекс активного елемента в навігаційній панелі.
2. **final onTap (Function(int)):** Функція, яка викликається при натисканні на елемент навігаційної панелі. В якості параметра приймає певне числове значення.

**Методи:**

1. **Widget build(BuildContext context):** Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертає віджет, що відображає навігаційну панель **BottomNavigationBar** з двома кнопками.
2. **static void navigateToScreen(BuildContext context, int index):** Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет та числове значення **index**, яке вказує на те, який екран має бути відкритий. Метод викликається для навігації до відповідного екрану за допомогою навігаційної панелі. Є статичним методом.

**Файл: calendar\_task\_list.dart**

Клас **CalendarTaskList:**

**Опис:** Цей клас відповідає за відображення списку завдань на вибраний день в календарі.

**Властивості:**

1. **final selectedDate (DateTime):** Дата, обраний день для якого потрібно відобразити список завдань.
2. **final selectedFilter (String):** Рядок, що являє собою поточний фільтр сортування завдань. За замовчуванням має значення **'By Date (Newest First)**'.

**Методи:**

1. **Widget build(BuildContext context):** Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертає віджет, що відображає список завдань для вибраної дати.
2. **List<Task> filterTasks(List<Task> tasks):** Метод приймає список всіх завдань **tasks** і повертає відфільтрований список завдань, які мають дату, ідентичну вибраній даті.
3. **bool isSameDay(DateTime date1, DateTime date2):** Метод приймає дві дати **date1** та **date2** і повертає логічне значення true, якщо вони ідентичні, і false в іншому випадку.

**Файл: task\_list\_screen.dart**

Клас **TaskListScreenState:**

**Опис:** Цей клас відповідає за відображення екрану списку завдань.

**Властивості:**

1. **showCompletedTasks (bool):** Логічне значення, що вказує, чи показувати завершені завдання. За замовчуванням – **false**.
2. **selectedMonth (String):** Рядкове значення, яке вказує обраний місяць для фільтрації завдань. За замовчуванням – **‘All’**.
3. **selectedYear (int):** Числове значення, яке вказує обраний рік для фільтрації завдань. За замовчуванням – **DateTime.now().year**.
4. **static selectedFilter (String):** Рядкове значення, яке вказує обраний фільтр сортування завдань. Є статичним полем. За замовчуванням – **‘By Date (Newest First)**.
5. **\_currentIndex (int):** Числове значення, яке вказує індекс обраного елементу нижньої навігаційної панелі. За замовчуванням – **0.**

**Методи:**

1. **void initState():** Метод життєвого циклу, який викликається при ініціалізації стану класу. Встановлює значення для статичного поля **showBorder** в класі **CustomDropdown** на значення **true**.
2. **Widget build(BuildContext context):** Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертає віджет, що відображає екран списку завдань.
3. **Widget buildMonthDropdown():** Метод будує віджет, що являє собою випадаючий список вибору місяця. Повертає CustomDropdown зі списком місяців.
4. **Widget buildYearDropdown():** Метод будує віджет, що являє собою випадаючий список вибору року. Повертає CustomDropdown зі списком років.
5. **Widget buildFilterDropdown():** Метод будує віджет, що являє собою випадаючий список вибору фільтра. Повертає CustomDropdown зі списком фільтрів.
6. **List<Task> filterTasks(List<Task> tasks):** Метод приймає список завдань **tasks**, фільтрує їх за обраним місяцем та роком та повертає вже відфільтрований список.
7. **void onFilterChanged(String? newValue):** Метод приймає нове значення фільтру **newValue**, після чого оновлює обраний фільтр та викликає метод сортування завдань у провайдера TaskListProvider.

**Файл: task\_dialog.dart**

Клас **TaskDialogState:**

**Опис:** Цей клас відповідає за відображення діалогового вікна для додавання або редагування завдання.

**Властивості:**

1. **final task (Task):** Об'єкт завдання, яке буде редагуватися або додаватися.
2. **final taskProvider (TaskListProvider):** Провайдер для керування списком завдань.
3. **final selectedFilter (String):** Рядкове значення, що вказує на обраний фільтр сортування завдань.
4. **final index (int?):** Необов'язкове ціле число, яке вказує індекс редагованого завдання у списку.
5. **late selectedCategory (String):** Рядкове значення, що використовується для збереження обраної категорії завдання у діалоговому вікні.

**Методи:**

1. **void initState():** Метод життєвого циклу, який викликається при ініціалізації стану класу. Встановлює початкове значення для властивості **selectedCategory** та встановлює поле **showBorder** в класі **CustomDropdown** на значення **false**.
2. **Widget build(BuildContext context):** Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертає віджета для відображення діалогового вікна. Начало формы

**Файл: calendar\_screen.dart**

Клас **CalendarScreenState**:

**Опис:** Цей коас відповідає за відображення екрану з календарем та списком завдань за обрану дату.

**Властивості:**

1. **\_calendarFormat** (CalendarFormat): Формат календаря. За замовчуванням має формат місяця.
2. **\_focusedDay** (DateTime): Обраний день, на якому міститься фокус у календарі.
3. **\_selectedDay** (DateTime): Обраний день у календарі – сьогоднішній день.
4. **\_currentIndex** (int): Числове значення, яке вказує індекс обраного елементу нижньої навігаційної панелі. За замовчуванням – **1.**

**Методи:**

1. **Widget build(BuildContext context)**: Метод приймає об'єкт **BuildContext**, який представляє контекст, у якому будується віджет. Будує та повертай екран з календарем та списком завдань.
2. **Widget buildTaskList()**: Метод будує та повертає віджет, що містить список завдань.
3. **bool hasTasksForDate(List<Task> tasks, DateTime date)**: Метод приймає список завдань **tasks** та дату **date** та перевіряє наявність завдань за певну дату. Він повертає **true**, якщо серед заданих **tasks** є хоча б одне завдання, яке має дату, яка співпадає з **date**.
4. **bool isSameDay(DateTime date1, DateTime date2)**: Метод приймає дві дати **date1** та **date2** і повертає логічне значення true, якщо вони ідентичні, і false в іншому випадку.

Оновлена архітектура:

