Міністерство освіти і науки України.

Національний університет “Львівська Політехніка”

Кафедра ЕОМ

Звіт про виконання лабораторної роботи №2
“ Розробка мобільних додатків**”**
на тему: «Створення власних віджетів у Flutter(візуальна складова аплікації)»

Виконав: ст. групи КІ-405

Петрович В.А.

Прийняв: Ремінний О.А.

Львів – 2025

Мета: Закріпити навички створення мобільних застосунків на Flutter, ознайомитися зі структурою проєкту, навчитися розробляти та перевикористовувати власні віджети, а також реалізувати навігацію між екранами.

Теоретичні відомості

1. Flutter як фреймворк розробки

Flutter — це фреймворк від компанії Google, призначений для створення кросплатформних мобільних, веб- та десктопних застосунків із єдиною базою коду.
Його головна особливість — використання мови програмування Dart і власного високопродуктивного графічного рушія Skia, який забезпечує відображення інтерфейсу без посередників (таких як Android View або UIKit).

Основні переваги Flutter:

- кросплатформність (один код для Android, iOS, Web, Desktop);
- швидке оновлення через механізм Hot Reload;
- висока продуктивність завдяки нативній компіляції;
- широка бібліотека готових віджетів (widgets) для побудови UI;
- зручна система навігації та станів.

---

2. Концепція віджетів у Flutter

Flutter побудований на системі віджетів — це основні елементи будь-якого інтерфейсу.

Віджет може бути:

- структурним (наприклад, Scaffold, AppBar, Column),
- відображальним (Text, Image, Icon),
- інтерактивним (TextField, ElevatedButton, GestureDetector).

Віджети бувають двох типів:

- StatelessWidget — не зберігає стану, використовується для статичних елементів інтерфейсу;

- StatefulWidget — має об’єкт стану State, що дозволяє змінювати вигляд віджета під час виконання програми (наприклад, при зміні значення повзунка, натисканні кнопки тощо).

---

3. Перевикористання віджетів

У Flutter зручно створювати власні користувацькі віджети, що дозволяє:

- спростити структуру коду;
- уникнути дублювання;
- покращити читабельність і підтримуваність;
- реалізувати єдиний стиль застосунку.

Наприклад, замість багаторазового використання стандартного TextField у різних місцях, можна створити свій CustomInput із фіксованим дизайном. Аналогічно створюються CustomButton чи PageTitle, які використовуються на багатьох екранах.

---

4. Навігація між екранами

Для переходу між сторінками застосунку у Flutter використовується система маршрутів (routes), реалізована через клас Navigator. Перехід виконується командами:

- Navigator.push(context, MaterialPageRoute(...)) — додавання нового екрана в стек;
- Navigator.pop(context) — повернення назад;
- або за допомогою іменованих маршрутів:
- Navigator.pushNamed(context, '/home');

Іменовані маршрути задаються у MaterialApp:

```
routes: {
  '/login': (_) => const LoginScreen(),
  '/home': (_) => const HomeScreen(),
  '/profile': (_) => const ProfileScreen(),
}
```

---

5. Адаптивність інтерфейсу

Оскільки Flutter дозволяє запускати застосунки на різних пристроях, важливо забезпечити адаптивний дизайн, який коректно масштабується під різні екрани:

- Використання віджетів Expanded, Flexible, GridView дозволяє уникати фіксованих розмірів.
- Використання MediaQuery дозволяє отримати розміри екрана.
- Віджети автоматично масштабуються під роздільну здатність пристрою завдяки системі logical pixels.

---

6. Дизайн і теми

Flutter підтримує Material Design — набір стандартів візуального оформлення від Google.
Тема застосунку задається через ThemeData, де можна визначити кольорову схему, стилі кнопок, тексту тощо:

```
theme: ThemeData(
  colorScheme: const ColorScheme.light(
    primary: Color(0xFF4CAF50),
    secondary: Color(0xFF81D4FA),
  ),
  scaffoldBackgroundColor: const Color(0xFFF9FBE7),
),
```

---

7. Лінтер і стандарти кодування

Лінтер (Linter) — це інструмент перевірки стилю коду.
Він допомагає підтримувати однаковий стиль у всьому проєкті, виявляє потенційні помилки та застарілі практики.
Перед здачею роботи всі зауваження лінтера повинні бути усунені за допомогою:

flutter analyze -> flutter format .

Завдання

Завдання

1. **Реалізувати мінімум 4 екрани:**
    - Екран входу (Login)
    - Екран реєстрації (Register)
    - Головний екран (Home) із візуалізацією ідеї застосунку
    - Екран профілю користувача (Profile)
2. **Створити перевикористовувані віджети**
3. **Реалізувати навігацію між усіма екранами.**
4. **Забезпечити адаптивний дизайн для різних розмірів екранів.**
5. **Виправити всі зауваження лінтера та форматувати код.**
6. **Не додавати складну бізнес-логіку (лише UI та навігацію).**

Виконання завдання

1. **Налаштування середовища Flutter**

Було встановлено **Flutter SDK** згідно з офіційною документацією за посиланням https://flutter.dev/docs/get-started/install.
Для розробки використано середовище **Visual Studio Code** з установленими плагінами *Flutter* та *Dart*.

Команда перевірки встановлення:

flutter doctor

підтвердила коректність налаштування середовища.

```
Microsoft Windows [Version 10.0.26100.6899]
(c) Корпорація Майкрософт. Усі права захищені.

C:\Users\Володя>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.35.6, on Microsoft Windows [Version 10.0.26100.6899], locale uk-UA)
[√] Windows Version (Ь ©Єа®6®дв Windows 11 Pro 64-bit, 24H2, 2009)
[√] Android toolchain - develop for Android devices (Android SDK version 36.1.0)
[√] Chrome - develop for the web
[√] Visual Studio - develop Windows apps (Visual Studio Community 2022 17.13.3)
[√] Android Studio (version 2025.1.4)
[√] Connected device (3 available)
[√] Network resources

• No issues found!
```

---

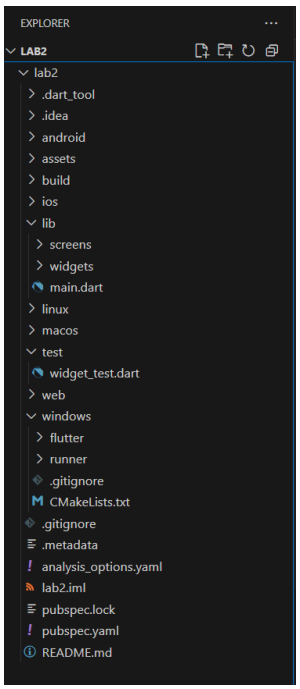
2. **Створення проєкту**

У терміналі було виконано команди:

flutter create lab2

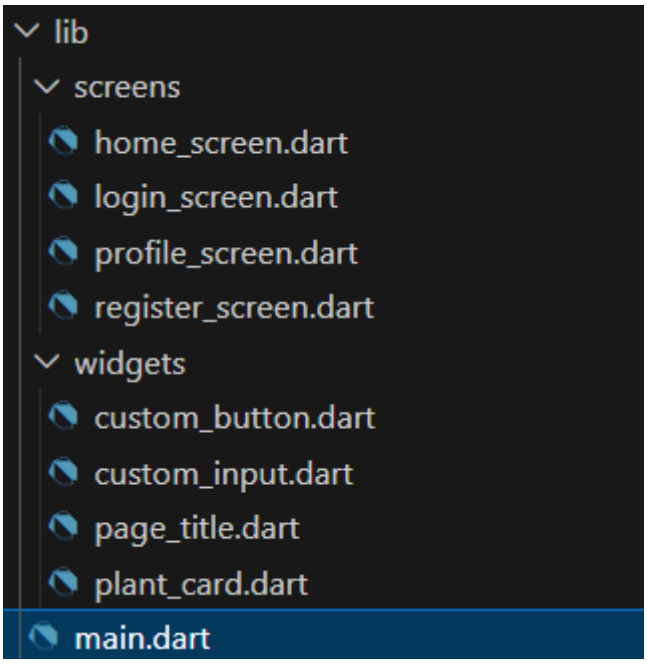
cd lab2

flutter run



Після створення структура проєкту була впорядкована:

у директорії **lib/** додано підпапку screens/ для окремих екранів (login, register, profile, home), а також widgets/ для перевикористовуваних компонентів.



3. Реалізація головного екрана (HomePage)

Створено файл **home_screen.dart**, у якому реалізовано головну сторінку застосунку.

Вона містить навігаційні кнопки до всіх інших екранів та просту візуалізацію ідеї застосунку.

```dart
import 'package:flutter/material.dart';

class Plant {
  Plant({
    required this.name,
    required this.image,
    required this.waterLevel,
  });

  final String name;
  final String image;
  final double waterLevel; // 0.0-1.0
}

class HomeScreen extends StatefulWidget {
  const HomeScreen({super.key});

  @override
  State<HomeScreen> createState() => _HomeScreenState();
```

```
}

class _HomeScreenState extends State<HomeScreen> {
  final List<Plant> _plants = [
    Plant(name: 'Фікус', image: 'assets/ficus.jpg', waterLevel: 0.8),
    Plant(name: 'Монстера', image: 'assets/monstera.jpg', waterLevel: 0.6),
    Plant(name: 'Кактус', image: 'assets/cactus.jpg', waterLevel: 0.9),
    Plant(name: 'Папороть', image: 'assets/fern.jpg', waterLevel: 0.3),
  ];

  void _showPlantDetails(Plant plant) {
    showModalBottomSheet<void>(
      context: context,
      shape: const RoundedRectangleBorder(
        borderRadius: BorderRadius.vertical(top: Radius.circular(16)),
      ),
      builder: (context) {
        return Padding(
          padding: const EdgeInsets.all(20),
          child: Column(
            mainAxisSize: MainAxisSize.min,
            children: [
              Text(
                plant.name,
                style: Theme.of(context).textTheme.headlineSmall,
              ),
              const SizedBox(height: 12),
              ClipRRect(
                borderRadius: BorderRadius.circular(12),
                child: Image.asset(
                  plant.image,
                  height: 150,
                  fit: BoxFit.cover,
                ),
              ),
              const SizedBox(height: 12),
              Text('Рівень поливу: ${(plant.waterLevel * 100).toInt()}%'),
              const SizedBox(height: 20),
              ElevatedButton.icon(
                onPressed: () {
                  setState(() => _plants.remove(plant));
                  Navigator.pop(context);
                },
                icon: const Icon(Icons.delete),
                label: const Text('Видалити'),
                style: ElevatedButton.styleFrom(
                  backgroundColor: Colors.redAccent,
                ),
              ),
            ],
.....................
```

4. Створення екрана входу

Реалізовано просту форму авторизації з полями для введення електронної пошти та пароля. У подальших лабораторних буде додана логіка перевірки.

```dart
import 'package:flutter/material.dart';
import 'package:lab2/widgets/custom_button.dart';
import 'package:lab2/widgets/custom_input.dart';
import 'package:lab2/widgets/page_title.dart';

class LoginScreen extends StatelessWidget {
  const LoginScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Padding(
          padding: const EdgeInsets.all(20),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              const PageTitle(title: 'Вхід'),
              const SizedBox(height: 20),
              const CustomInput(hint: 'Email'),
              const SizedBox(height: 12),
              const CustomInput(hint: 'Пароль', obscure: true),
              const SizedBox(height: 20),
              CustomButton(
                text: 'Увійти',
                onTap: () {
                  Navigator.pushNamed(context, '/home');
                },
              ),
              const SizedBox(height: 12),
              TextButton(
                onPressed: () {
                  Navigator.pushNamed(context, '/register');
                },
                child: const Text('Створити акаунт'),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

5. Створення екрана реєстрації

Додано поля для створення нового облікового запису користувача.

```
   import 'package:flutter/material.dart';

import 'package:lab2/widgets/custom_button.dart';
import 'package:lab2/widgets/custom_input.dart';
import 'package:lab2/widgets/page_title.dart';

class RegisterScreen extends StatelessWidget {
  const RegisterScreen({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Padding(
          padding: const EdgeInsets.all(20),
          child: ListView(
            children: [
              const PageTitle(title: 'Реєстрація'),
              const CustomInput(hint: 'Ім’я'),
              const CustomInput(hint: 'Email'),
              const CustomInput(hint: 'Пароль', obscure: true),
              CustomButton(
                text: 'Зареєструватись',
                onTap: () => Navigator.pushNamed(context, '/home'),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

---

6. Створення екрана профілю користувача

У кореневій директорії створено файл **analysis_options.yaml**.

```
import 'package:flutter/material.dart';
import 'package:lab2/widgets/page_title.dart';

class ProfileScreen extends StatelessWidget {
  const ProfileScreen({super.key});

  @override
```

```
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Профіль')),
      body: Padding(
        padding: const EdgeInsets.all(20),
        child: Column(
          children: [
            const PageTitle(title: 'Петрович 🌱'),
            const SizedBox(height: 20),
            const ListTile(
              leading: Icon(Icons.email),
              title: Text('user@example.com'),
            ),
            const ListTile(
              leading: Icon(Icons.settings),
              title: Text('Налаштування'),
            ),
            const Spacer(),
            ElevatedButton(
              onPressed: () => Navigator.pushNamedAndRemoveUntil(
                context, '/login', (r) => false),
              child: const Text('Вийти'),
            ),
          ],
        ),
      ),
    );
```

7. Додавання навігації між екранами

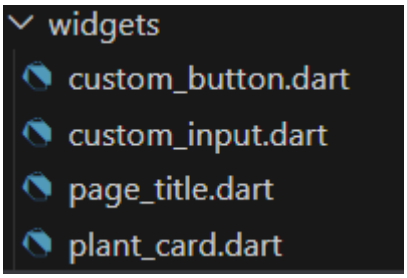
У файлі main.dart налаштовано маршрути для всіх сторінок застосунку:

```
import 'package:flutter/material.dart';
import 'package:lab2/screens/home_screen.dart';
import 'package:lab2/screens/login_screen.dart';
import 'package:lab2/screens/profile_screen.dart';
import 'package:lab2/screens/register_screen.dart';

void main() => runApp(const AutoWateringApp());

class AutoWateringApp extends StatelessWidget {
  const AutoWateringApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'AutoWatering',
      theme: ThemeData(
        colorScheme: const ColorScheme.light(
          primary: Color(0xFF4CAF50),
          secondary: Color(0xFF81D4FA),
          surface: Color(0xFFF9FBE7),
        ),
        scaffoldBackgroundColor: const Color(0xFFF9FBE7),
```

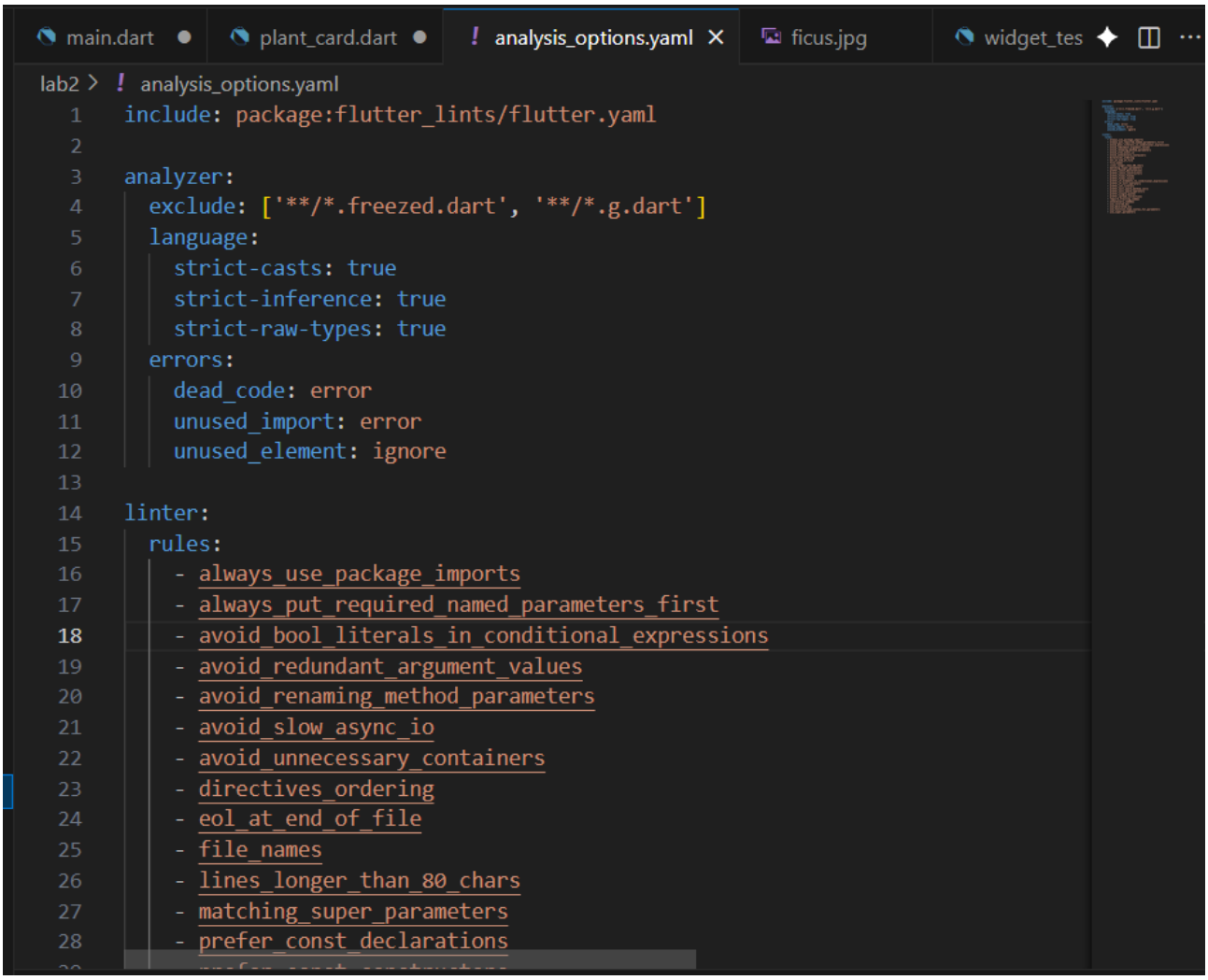
```
        appBarTheme: const AppBarTheme(
          backgroundColor: Color(0xFF4CAF50),
          foregroundColor: Colors.white,
          elevation: 2,
        ),
        elevatedButtonTheme: ElevatedButtonThemeData(
          style: ElevatedButton.styleFrom(
            backgroundColor: const Color(0xFF4CAF50),
            foregroundColor: Colors.white,
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(14),
            ),
          ),
        ),
      ),
      initialRoute: '/login',
      routes: {
        '/login': (_) => const LoginScreen(),
        '/register': (_) => const RegisterScreen(),
        '/home': (_) => const HomeScreen(),
        '/profile': (_) => const ProfileScreen(),
      },
    );
  }
}
```

8. Оптимізація та перевикористання віджетів

Було створено окремий файл **widgets/custom_button.dart, custom_imput.dart, page_title.dart, plant_card.dart**



9. Налаштування лінтера

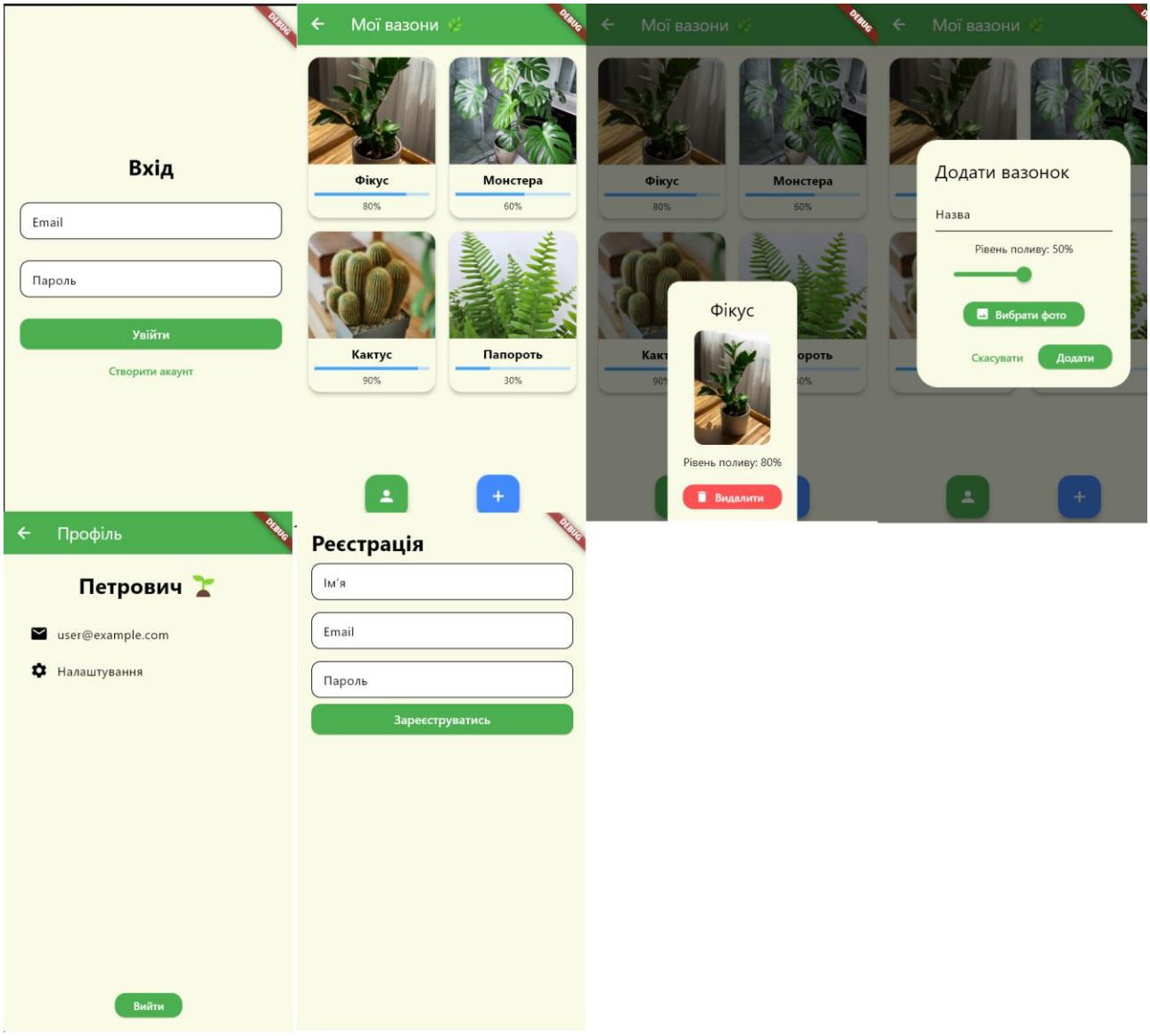


У нього було вставлено правила перевірки коду з наданого репозиторію. Після запуску перевірки усі попередження лінтера були виправлені.

8. Публікація проєкту

Після завершення роботи код було завантажено у власний репозиторій GitHub. Створено окрему гілку для лабораторної, зроблено **Pull Request**, додано коментар із ПІБ, посиланням на док і скріншотами роботи застосунку.

Скріншоти роботи програми



Висновок:

У ході виконання лабораторної роботи було створено мобільний застосунок на Flutter із чотирма основними екранами — логін, реєстрація, профіль та головна сторінка. Реалізовано навігацію між сторінками, перевикористання віджетів та базовий адаптивний дизайн. Усі вимоги лінтера дотримано, а структура проєкту організована згідно з принципами чистого коду.