**Лабораторна робота № 4**

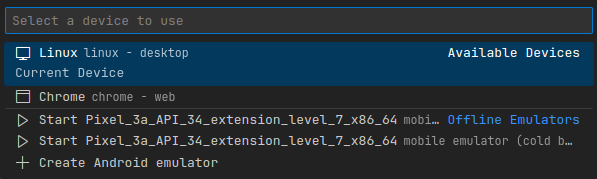
*Тема:* Створення десктопного додатку засобами Flutter. Взаємодія з API

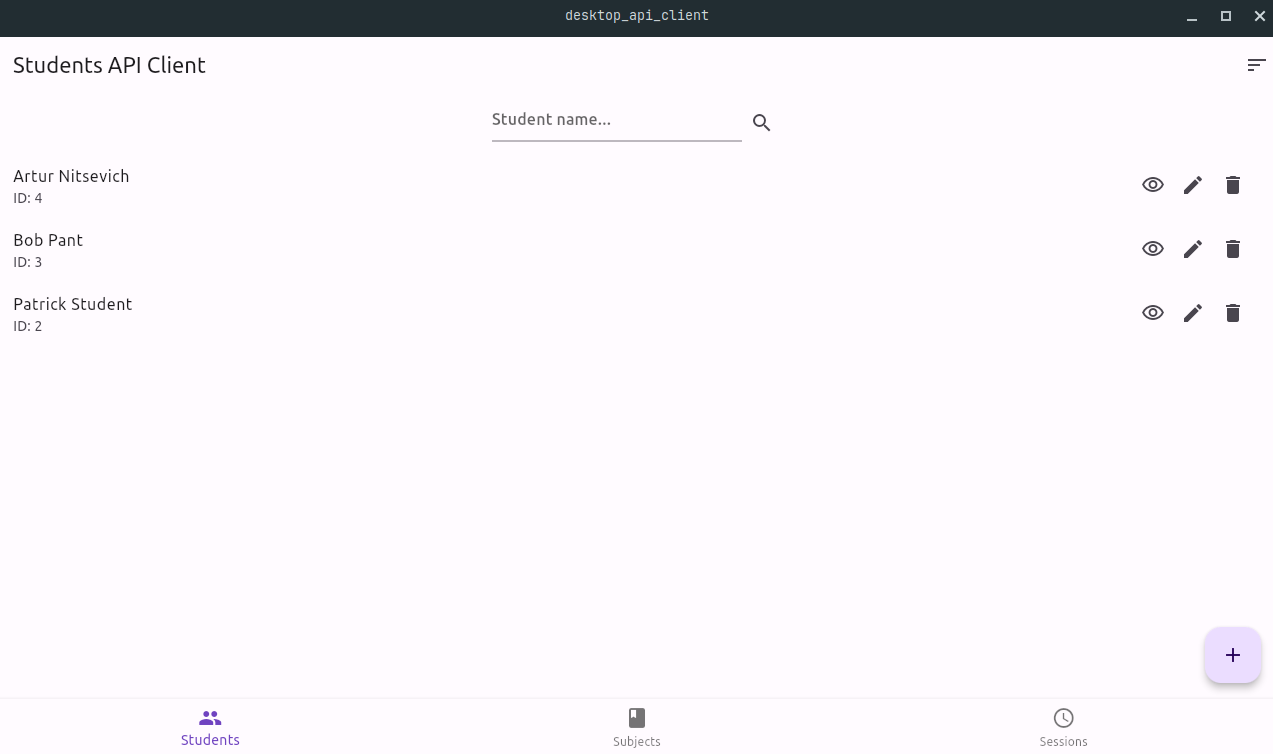
**Завдання**

Розробити проект для роботи із API, що створювалась на **практичній роботі №1**. Забезпечити CRUD, перегляд інформації про записи, загальний список.

**Хід роботи**

Для компіляції додатка на десктоп потрібно вибрати відповідний пристрій в VS Code:





Як бачимо, все круто запустилось, але інформація про студентів виглядає трохи нудно.

Надамо можливість студентам мати фотографії. Для цього потрібно оновити модель студента:  
import 'package:json\_annotation/json\_annotation.dart';

part 'student.g.dart';

@JsonSerializable()

class Student {

int? id;

String firstName;

String lastName;

String? photo;

Student({

this.id,

required this.firstName,

required this.lastName,

this.photo,

});

factory Student.fromJson(Map<String, dynamic> json) =>

\_$StudentFromJson(json);

Map<String, dynamic> toJson() => \_$StudentToJson(this);

Student copyWith({

int? id,

String? firstName,

String? lastName,

String? photo,

}) {

return Student(

id: id ?? this.id,

firstName: firstName ?? this.firstName,

lastName: lastName ?? this.lastName,

photo: photo ?? this.photo,

);

}

}

Після оновлення обов’язково потрібно запустити команду

dart run build\_runner build

в консолі, аби перегенерувати методи для сереалізації та десереалізації моделі в JSON.

Після цього, потрібно оновити сторінку для додавання/редагування студентів. На цю сторінку потрібно додати можливітсь вибирати фото з пристрою. Для цього додамо пакет

image\_picker наступною командою:  
  
 flutter pub add image\_picker  
  
Коли пакет вдало встановився, можна переходити до модифікації сторінки. Замінимо класдля сторінки **StudentAddEditPageState** на такий:

class \_StudentAddEditPageState extends State<StudentAddEditPage> {

late TextEditingController \_firstNameController;

late TextEditingController \_lastNameController;

final \_repository = const StudentProvider();

String \_selectedPhoto = "";

@override

void initState() {

super.initState();

\_firstNameController =

TextEditingController(text: widget.studentToEdit?.firstName ?? '');

\_lastNameController =

TextEditingController(text: widget.studentToEdit?.lastName ?? '');

\_selectedPhoto = widget.studentToEdit?.photo ?? "";

}

Future<void> \_pickImage() async {

final pickedFile =

await ImagePicker().pickImage(source: ImageSource.gallery);

if (pickedFile != null) {

setState(() {

\_selectedPhoto = pickedFile.path;

});

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text(widget.studentToEdit == null

? 'Add Student'

: 'Edit Student № ${widget.studentToEdit!.id}'),

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

children: [

TextField(

controller: \_firstNameController,

decoration: const InputDecoration(labelText: 'First Name'),

),

TextField(

controller: \_lastNameController,

decoration: const InputDecoration(labelText: 'Last Name'),

),

const SizedBox(height: 16),

ElevatedButton(

onPressed: \_pickImage,

child: const Text('Pick Photo'),

),

if (\_selectedPhoto.isNotEmpty)

Image.file(

File(\_selectedPhoto),

height: 100,

width: 100,

fit: BoxFit.cover,

),

const SizedBox(height: 16),

ElevatedButton(

onPressed: () async {

if (widget.studentToEdit == null) {

final newStudent = Student(

firstName: \_firstNameController.text,

lastName: \_lastNameController.text,

photo: \_selectedPhoto,

);

await \_repository.create(newStudent);

// ignore: use\_build\_context\_synchronously

Navigator.of(context).pop(newStudent);

} else {

final updatedStudent = widget.studentToEdit!.copyWith(

firstName: \_firstNameController.text,

lastName: \_lastNameController.text,

photo: \_selectedPhoto,

);

await \_repository.update(updatedStudent.id!, updatedStudent);

// ignore: use\_build\_context\_synchronously

Navigator.of(context).pop(updatedStudent);

}

},

child:

Text(widget.studentToEdit == null ? 'Add' : 'Save Changes'),

),

],

),

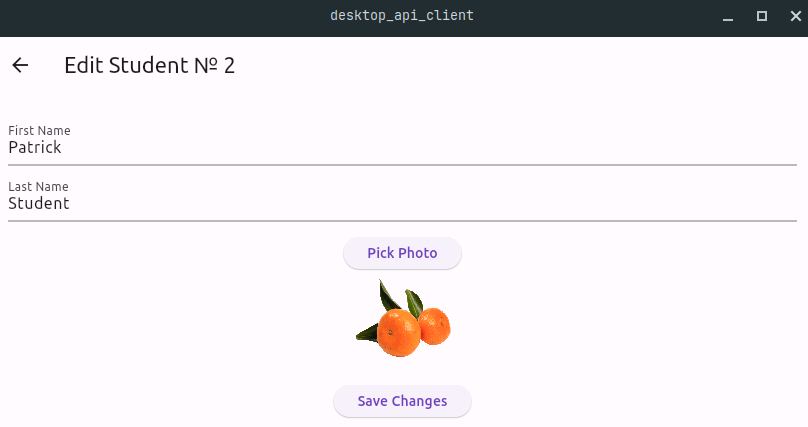
),

);

}

}

За допомогою цього коду, на сторінці з’являється можливість вибрати фото з пристрою після натиснення на відповідну кнопку. Також після вибору фото, його буде показано на сторінці.



Тепер, треба змінити діалогове вікно з інформацією про студента, аби там також можна було переглянути фото. Замінимо метод **build** класу **MyStudentDialogState** на такий:  
@override

Widget build(BuildContext context) {

return Center(

child: Container(

padding: const EdgeInsets.all(16.0),

decoration: BoxDecoration(

color: Colors.white,

borderRadius: BorderRadius.circular(16),

),

constraints: const BoxConstraints(maxWidth: 400, maxHeight: 300),

child: FutureBuilder(

future: \_repository.getById(widget.id),

builder: (context, AsyncSnapshot<Student?> snapshot) {

if (snapshot.connectionState == ConnectionState.done) {

if (snapshot.hasData) {

final Student student = snapshot.data!;

return Row(

children: [

if (student.photo != null && student.photo!.isNotEmpty)

SizedBox(

width: 200,

height: 200,

child: Image.file(

File(student.photo!),

fit: BoxFit.contain,

),

),

Column(

mainAxisAlignment: MainAxisAlignment.start,

crossAxisAlignment: CrossAxisAlignment.start,

children: [

const Row(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Text(

'Student',

style: TextStyle(

fontWeight: FontWeight.bold, fontSize: 16),

),

],

),

Text('ID: ${student.id}'),

Text('First name: ${student.firstName}'),

Text('Last name: ${student.lastName}'),

const Spacer(),

Row(

mainAxisAlignment: MainAxisAlignment.end,

children: [

TextButton(

onPressed: () => Navigator.of(context).pop(),

child: const Text('OK'),

)

],

),

],

),

],

);

} else if (snapshot.hasError) {

return Text('Error: ${snapshot.error}');

} else {

return const Text('Student not found.');

}

} else {

return const CircularProgressIndicator();

}

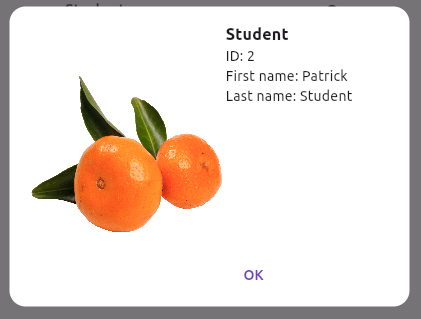
},

),

),

);

}

  
Відтепер, діалогові вікна студентів будуть виглядати приблизно так: