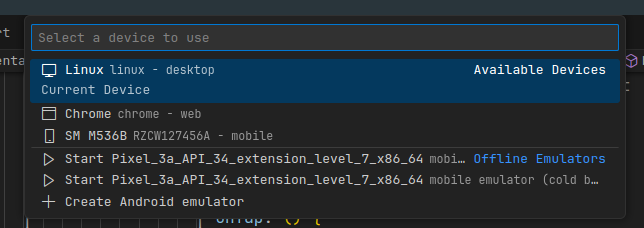
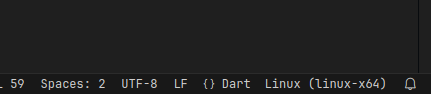
**Лабораторна робота № 6**

*Тема:* Створення мобільного додатку засобами Flutter, взаємодія з API

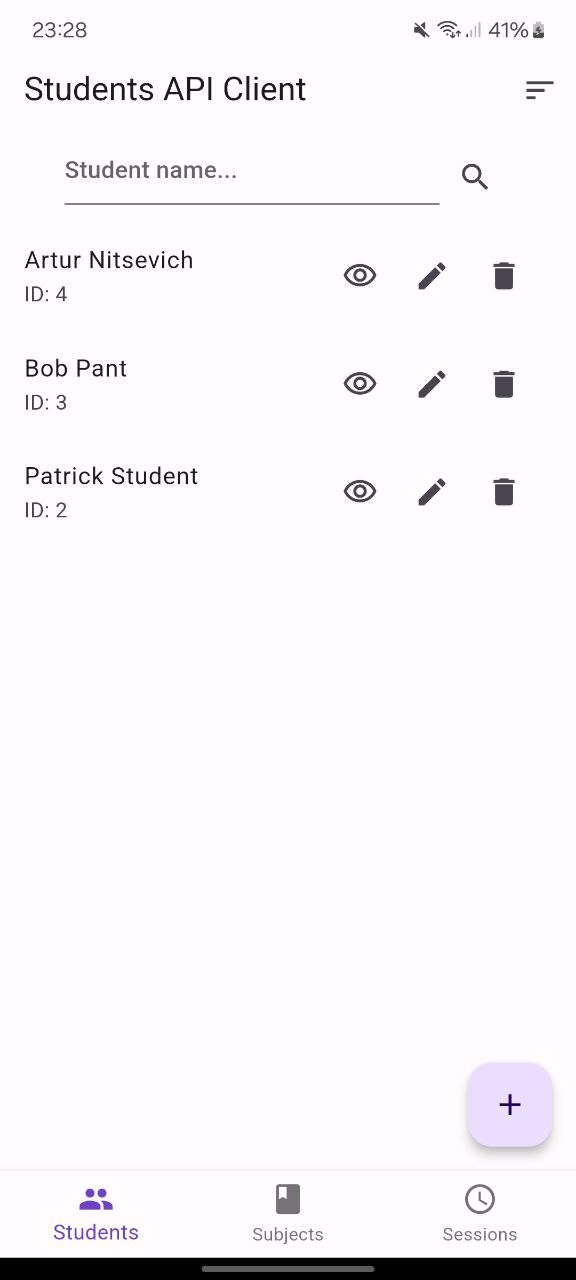
**Хід роботи**

Для компіляції проекта під мобільний пристрій, потрібно виконати наступне:

1. Вибрати мобільний пристрій, або емулятор, в VS Code:



1. Почати компіляцію за допомогою **F5** або flutter run в консолі
2. Побачити, що додаток встановився та запустився на пристрої

Показувати дані в вигляді списку дуже круто, але з точки зору UX дизайну не дуже правильно, бо за великого обсягу даних отримати до них доступ буде не дуже зручно, тому переробимо сторінку в вигляд сітки.

Для цього змініть метод **build** віджета **StudentPage** на наступний:  
@override

Widget build(BuildContext context) {

return Scaffold(

body: Column(

children: [

MySearchBar(

searchController: \_searchController,

onSearched: () {

setState(() {

\_searchQuery = \_searchController.text;

});

},

hintText: 'Student name...',

),

Expanded(

child: FutureBuilder<List<Student>?>(

future: \_repository.getAll(

sortingInput: widget.sortingValue,

searchInput: \_searchQuery,

),

builder: (context, snapshot) {

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

return const Center(child: CircularProgressIndicator());

} else if (snapshot.hasError) {

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

} else if (!snapshot.hasData || snapshot.data!.isEmpty) {

return const Center(child: Text('No students found.'));

} else {

List<Student> students = snapshot.data!;

return GridView.builder(

gridDelegate:

const SliverGridDelegateWithFixedCrossAxisCount(

crossAxisCount: 2,

crossAxisSpacing: 8.0,

mainAxisSpacing: 8.0,

),

itemCount: students.length,

itemBuilder: (context, index) {

final item = students[index];

return InkWell(

onTap: () {

showAdaptiveDialog(

context: context,

builder: (context) => MyStudentDialog(id: item.id!),

);

},

child: Card(

elevation: 2.0,

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

Text(

'${item.firstName} ${item.lastName}',

textAlign: TextAlign.center,

),

const SizedBox(height: 8.0),

Text('ID: ${item.id}',

textAlign: TextAlign.center),

Row(

mainAxisAlignment:

MainAxisAlignment.spaceEvenly,

children: [

IconButton(

icon: const Icon(Icons.edit),

onPressed: () async {

final result = await Navigator.of(context)

.push<Student?>(

MaterialPageRoute(

builder: (ctx) {

return StudentAddEditPage(

studentToEdit: item);

},

),

);

setState(() {

if (result != null) {

ScaffoldMessenger.of(context)

.showSnackBar(

SnackBar(

content: Text(

'Student ${result.id} updated',

),

),

);

}

});

},

),

IconButton(

icon: const Icon(Icons.delete),

onPressed: () async {

final response =

await \_repository.delete(item.id!);

setState(() {

ScaffoldMessenger.of(context)

.clearSnackBars();

ScaffoldMessenger.of(context)

.showSnackBar(

SnackBar(

content: Text(response),

),

);

});

},

),

],

),

],

),

),

);

},

);

}

},

),

),

],

),

floatingActionButton: FloatingActionButton(

onPressed: () async {

final result = await Navigator.of(context).push<Student?>(

MaterialPageRoute(

builder: (context) => const StudentAddEditPage(),

),

);

setState(() {

if (result != null) {

ScaffoldMessenger.of(context).showSnackBar(SnackBar(

content: Text(

'Student ${result.firstName} ${result.lastName} created'),

));

}

});

},

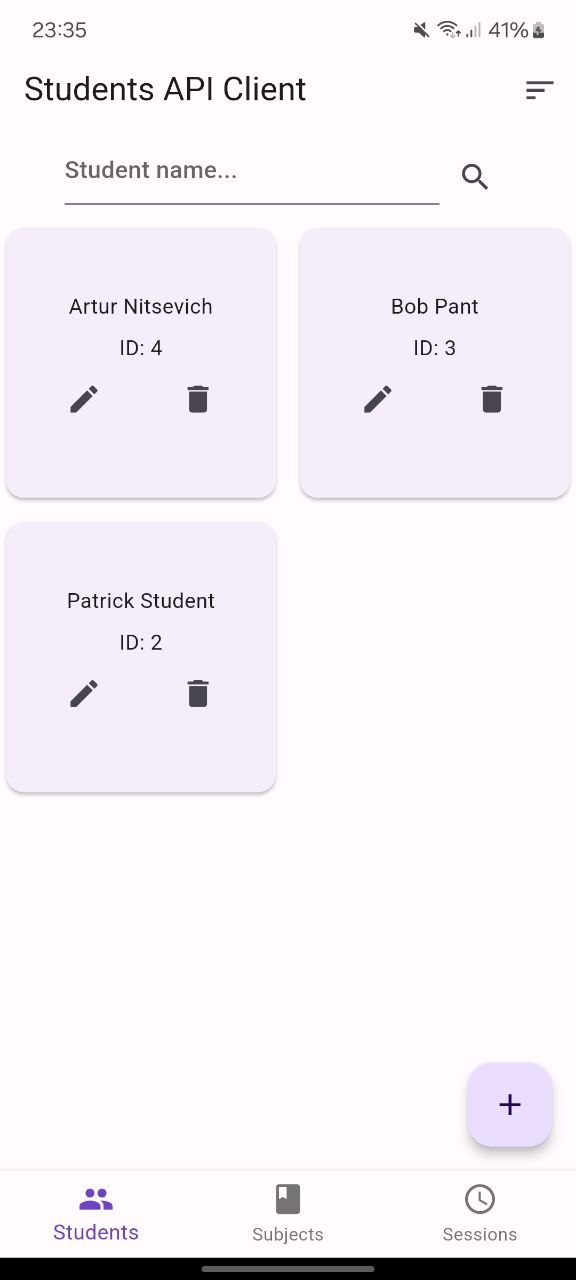
child: const Icon(Icons.add),

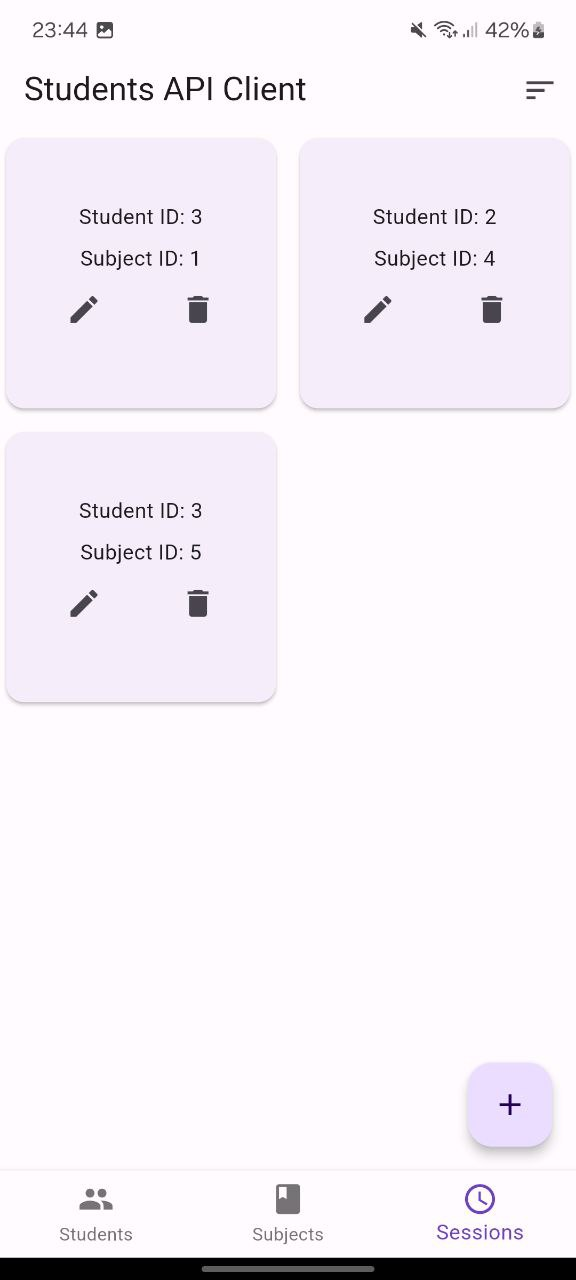
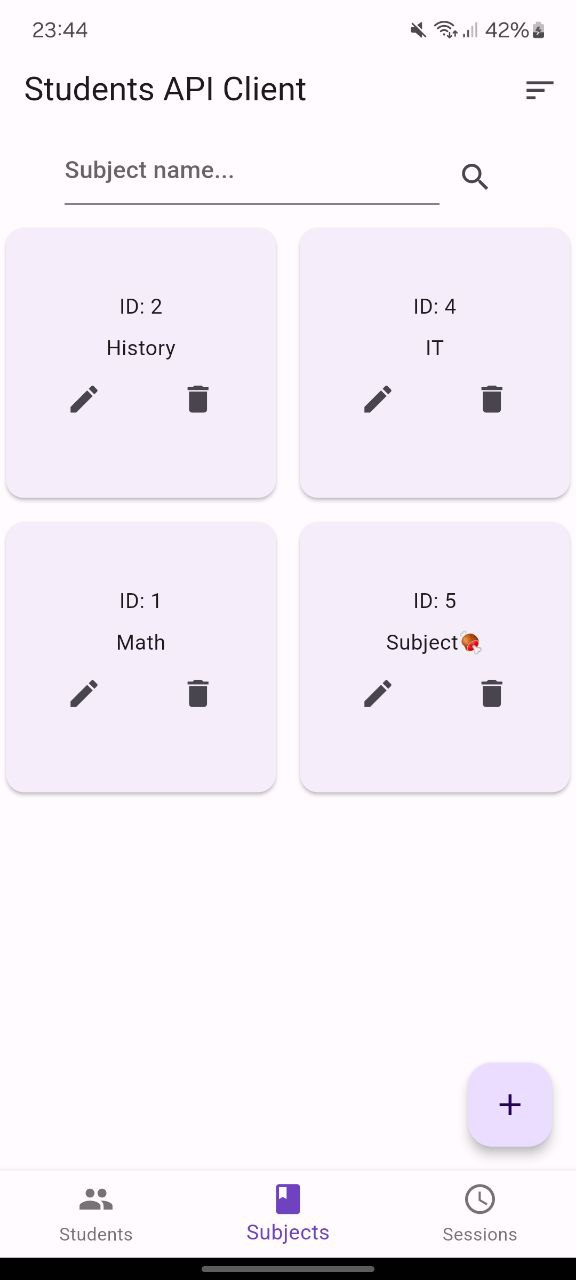
),

);

}

Після того як замінили вміст сторінки, збережіть файл та дочекайтеся перезавантаження додатка. Після повного завантаження сторінка буде виглядати приблизно так:

  
Це вже точно виглядає краще. За тим ж принципом можна переробити і інші сторінки.



**Завдання**

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

**Література**