Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа №9

по дисциплине "Технологии разработки мобильных приложений"

ТЕМА РАБОТЫ:

Разработка клиент-серверного мобильного приложения

Выполнил:

студент гр. ПРИ-120

Парахин К.В.

Принял:

преподаватель кафедры ИСПИ

Рощина А.И.

Владимир 2023 г.

Цель работы:

Научиться разрабатывать мобильное приложение с использованием клиент-серверной архитектуры и протокола Rest API.

Выполнение работы:

Серверная часть приложения была написана на фреймворке ASP.NET Core, предметная область приложения – «Многозадачный календарь».

Она включает в себя пользователя (без ролей – разделение происходит только на аутентифицированных и неаутентифицированных пользователей). Аутентифицированные пользователи имеют доступ к своему личному кабинету (и могут взаимодействовать с остальными сущностями системы – а именно создавать, редактировать и просматривать свои группы, создавать и просматривать свои запланированные мероприятия и задачи в календаре, а также получать отчеты).

Неаутентифицированный пользователь должен войти в систему или зарегистрироваться в ней – чтобы получить свой личный токен (по которому на уровне middleware сервера осуществляется проверка аутентификации запроса каждого пользователя). В случае, если пользователь отправляет запрос без токена (или токен устарел) – то сервер дает ответ HTTP 401 Unathorized. В случае, если же токен существует – но пользователь не имеет доступа к какой-то информации (например, не может редактировать чужие группы и задачи, участвовать в мероприятиях, в которых он не был приглашен – то тогда сервер возвращает ответ HTTP 403 Forbidden.

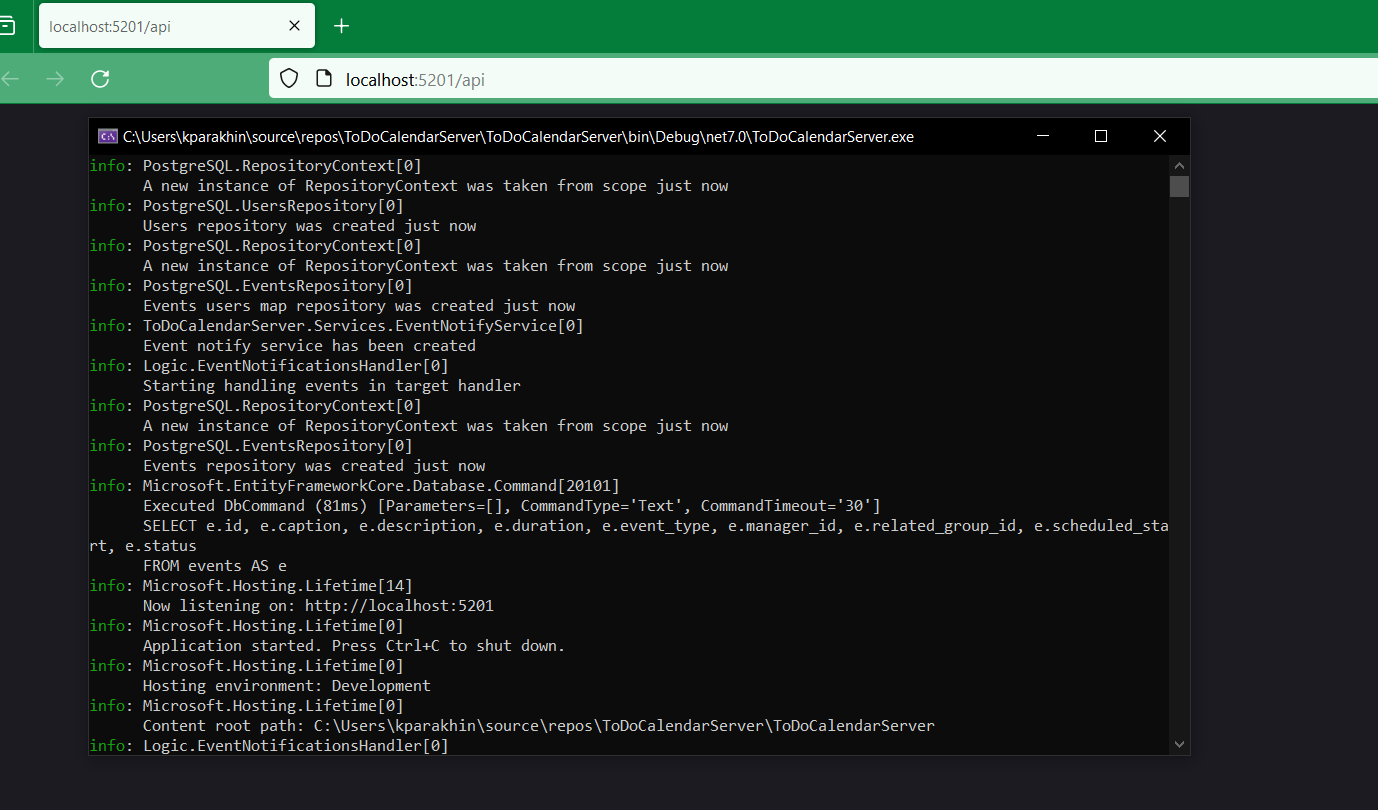


Рисунок 1. Запуск серверного приложения на порту localhost:5201

Клиентская часть приложения будет написана с использованием фреймворка Flutter.

Для выполнения http запросов к API нашего сервера будем использовать пакет: *http: ^0.13.3*

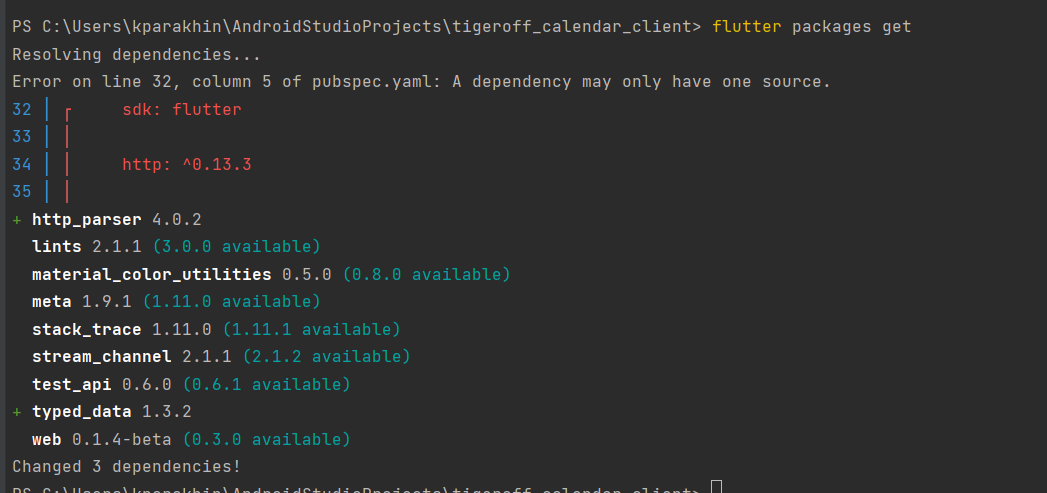


Рисунок 2. Получение зависимостей для работы с http в терминале Flutter

Далее создаем в папке lib файл api\_service.dart, в котором создаем асинхронные методы, делающие HTTP запрос к API и отдающие ответ клиенту.

Фрагмент кода из файла api\_service.dart:

import 'package:http/http.dart' as http;  
  
// запросы связанные с пользователем  
Future<String> registerNewUser() async {  
 final url = Uri.*parse*('http://localhost:5201/users/register');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'email': 'parahinvaler5@gmail.com',  
 'name': 'valer5',  
 'password': 'tigeroff2002',  
 'phone\_number': '8-903-225-50-27'  
 }  
 );  
 if (response.statusCode == 200) {  
 print(response.body);  
 return response.body.toString();  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 return response.body.toString();  
 }  
}  
  
Future<String> loginUser() async {  
 final url = Uri.*parse*('http://localhost:5201/users/login');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'email': 'parahinvaleri5@gmail.com',  
 'password': 'tigeroff2002'  
 }  
 );  
 if (response.statusCode == 200) {  
 print(response.body);  
 return response.body.toString();  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 return response.body.toString();  
 }  
}  
  
Future<void> updateUserInfo() async {  
 final url = Uri.*parse*('http://localhost:5201/users/get\_info');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'name': 'Kiriusha'  
 },  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 }  
}  
  
Future<void> getUserInfo() async {  
 final url = Uri.*parse*('http://localhost:5201/users/get\_info');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408'  
 },  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 }  
}

Создадим также аналогичные запросы к эндпоинтам групп, мероприятий, задач и отчетов (для добавления, изменения, просмотра информации).

Future<String> addNewGroup() async {  
 final url = Uri.*parse*('http://localhost:5201/groups/create');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'group\_name': 'Add dot net please',  
 'group\_type': 'Job'  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 return response.body.toString();  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 return response.body.toString();  
 }  
}

Future<void> getGroupInfo() async {  
 final url = Uri.*parse*('http://localhost:5201/groups/get\_group\_info');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'group\_id': 10  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 }  
}

Future<String> addNewTask() async {  
 final url = Uri.*parse*('http://localhost:5201/tasks/create');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'caption': 'Create new api specification',  
 'description': 'Create new api asp .net core specification',  
 'task\_type': 'JobComplete',  
 'task\_status': 'ToDo',  
 'implementer\_id': 3  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 return response.body.toString();  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 return response.body.toString();  
 }  
}

Future<void> getTaskInfo() async {  
 final url = Uri.*parse*('http://localhost:5201/tasks/get\_task\_info');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'task\_id': 4  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 }  
}

// запросы связанные с мерприятиями  
Future<String> scheduleNewEvent() async {  
 final url = Uri.*parse*('http://localhost:5201/events/schedule\_new');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'caption': 'New december olimpiad discussion',  
 'description': 'Discussion about ICPC decemper tour olimpiad',  
 'scheduled\_start': '2023-11-05T18:00+00:00',  
 'duration': '00:30:00',  
 'event\_type': 'OneToOne',  
 'event\_status': 'NotStarted',  
 'group\_id': 10,  
 'guests\_ids': [2]  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 return response.body.toString();  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 return response.body.toString();  
 }  
}

Future<void> getEventInfo() async {  
 final url = Uri.*parse*('http://localhost:5201/events/get\_event\_info');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'event\_id': 3  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 }  
}

// запросы связанные с мерприятиями  
Future<String> addNewReport() async {  
 final url = Uri.*parse*('http://localhost:5201/reports/perform\_new');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 2,  
 'token': '0696142000',  
 'report\_type': 'EventsReport',  
 'begin\_moment': '2023-10-22T00:00:00+00:00',  
 'end\_moment': '2023-11-06T00:00:00+00:00'  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 return response.body.toString();  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 return response.body.toString();  
 }  
}  
  
Future<void> getReportInfo() async {  
 final url = Uri.*parse*('http://localhost:5201/reports/get\_report\_info');  
 final response = await http.post(  
 url,  
 headers: {'Content-Type': 'application/json'},  
 body:  
 {  
 'user\_id': 1,  
 'token': '0895439408',  
 'report\_id': 11  
 }  
 );  
  
 if (response.statusCode == 200) {  
 print(response.body);  
 } else {  
 print('Request failed with status: ${response.statusCode}.');  
 }  
}