Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 6

по курсу «Разработка мобильных приложений»

«Реализация взаимодействия клиента и сервера через WebSockets»

Студент группы ИУ9-71Б Окутин Д. А.

Преподаватель Посевин Д. П.

Москва 2024

1 Цель

Цель данной лабораторной работы: разработка мобильного приложения и сервера для обмена командами посредством WebSockets с целью сохранения и чтения числового значения на сервере.

2 Задание

Необходимо реализовать мобильное приложение, которое через WebSockets отправляет команды на сервер для сохранения определенного числа и его последующего чтения. Сервер также должен быть реализован с использованием WebSockets и обрабатывать соответствующие команды от клиента. Приложение должно предоставлять интерфейс для ввода числа, отправки команды сохранения, а также для запроса и отображения текущего значения числа, хранящегося на сервере.

3 Реализация

В данной секции представлены основные компоненты реализации проекта: серверная часть и клиентское мобильное приложение.

3.1 Серверная часть

Сервер реализован с использованием WebSockets на языке программирования *Dart*.

Листинг 1: Серверный код

```
1 import 'dart:io';
2
3 Future<void> main() async {
4   final server = await HttpServer.bind(InternetAddress.anyIPv4, 8090);
5   print('                                    http://${server.address.host}:
      ${server.port}/');
6
7   await for (var request in server) {
8     if (WebSocketTransformer.isUpgradeRequest(request)) {
9       final socket = await WebSocketTransformer.upgrade(request);
```

```
10        handleWebSocket(socket);
11      } else {
12        request.response.statusCode = HttpStatus.methodNotAllowed;
13        request.response.write('                                    ');
14        request.response.close();
15      }
16    }
17  }
18
19  void handleWebSocket(WebSocket socket) async {
20    print('                                ');
21    socket.listen((message) async {
22      try {
23        if (message == 'GET_NUMBER') {
24          await sendSavedNumber(socket);
25        } else {
26          int number = int.parse(message);
27
28          final file = File('data.txt');
29          await file.writeAsString('$number');
30
31          print('                              : $number');
32          socket.add('                                  : $number');
33        }
34      } catch (e) {
35        socket.add('
                        : ${e.toString()}');
36      }
37    }, onDone: () {
38      print('                                ');
39    }, onError: (error) {
40      print('             : $error');
41    });
42  }
43
44  Future<void> sendSavedNumber(WebSocket socket) async {
45    final file = File('data.txt');
46    if (await file.exists()) {
47      String content = await file.readAsString();
48      print('                                                    :
      $content');
49      socket.add(content);
50    } else {
51      socket.add('                            .');
52    }
53  }
```

3.2 Клиентское мобильное приложение

Мобильное приложение разработано с использованием фреймворка *Flutter* и подключается к серверу через WebSockets для отправки и получения команд.

Листинг 2: Клиентский код

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Lab6',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: NumberForm(),
      debugShowCheckedModeBanner: false,
    );
  }
}

class NumberForm extends StatefulWidget {
  @override
  _NumberFormState createState() => _NumberFormState();
}

class _NumberFormState extends State<NumberForm> {
  final _formKey = GlobalKey<FormState>();
  final _numberController = TextEditingController();
  int _currentNumber = 0;
  late WebSocketChannel channel;

  @override
  void initState() {
    super.initState();
    //                                WebSocket-
    channel = WebSocketChannel.connect(
      Uri.parse('ws://195.19.41.28:8090/'), //
                    ,                  WebSocket URL
    );
```

```
40
41      channel.stream.listen((data) {
42        setState(() {
43          _currentNumber = int.parse(data);
44        });
45      });
46    }
47
48    @override
49    void dispose() {
50      channel.sink.close(); //
51      super.dispose();
52    }
53
54    void sendNumber() {
55      channel.sink.add(_currentNumber.toString());
56    }
57
58    void sendNumberFromNumberController() {
59      channel.sink.add(_numberController.text);
60    }
61
62    void increment() {
63      setState(() {
64        _currentNumber++;
65      });
66      sendNumber();
67    }
68
69    void decrement() {
70      setState(() {
71        _currentNumber--;
72      });
73      sendNumber();
74    }
75
76    void getNumber() {
77      channel.sink.add('GET_NUMBER');
78    }
79
80    @override
81    Widget build(BuildContext context) {
82      return Scaffold(
83        appBar: AppBar(
84          title: Text('Lab6'),
```

```
85        ),
86        body: Padding(
87          padding: const EdgeInsets.all(16.0),
88          child: Form(
89            key: _formKey,
90            child: Column(
91              children: <Widget>[
92                TextFormField(
93                  controller: _numberController,
94                  decoration: InputDecoration(labelText: '
                  '),
95                  keyboardType: TextInputType.number,
96                  validator: (value) {
97                    if (value == null || value.isEmpty) {
98                      return '                              ';
99                    }
100                   return null;
101                 },
102               ),
103               const SizedBox(height: 20),
104               ElevatedButton(
105                 onPressed: () {
106                   if (_formKey.currentState!.validate()) {
107                     sendNumberFromNumberController();
108                   }
109                 },
110                 child: Text('                              '),
111               ),
112               SizedBox(height: 20),
113               ElevatedButton(
114                 onPressed: getNumber,
115                 child: Text('                              '),
116               ),
117               const SizedBox(height: 20),
118               Text('          : $_currentNumber', style: TextStyle(
      fontSize: 20)),
119               SizedBox(height: 20),
120               Row(
121                 mainAxisAlignment: MainAxisAlignment.center,
122                 children: [
123                   ElevatedButton(
124                     onPressed: decrement,
125                     child: Text('-                    '),
126                   ),
127                   SizedBox(width: 20),
128                   ElevatedButton(
```

```
129                         onPressed: increment,
130                         child: Text('+                    '),
131                       ),
132                     ],
133                   ),
134                 ],
135               ),
136             ),
137           ),
138       );
139     }
140 }
```

4 Результаты

В результате выполнения лабораторной работы было создано мобильное приложение, способное взаимодействовать с сервером через WebSockets. Приложение позволяет пользователю вводить число, отправлять его на сервер для сохранения, изменять его, а также запрашивать текущее значение числа с сервера. На рисунке 1 представлен интерфейс приложения.

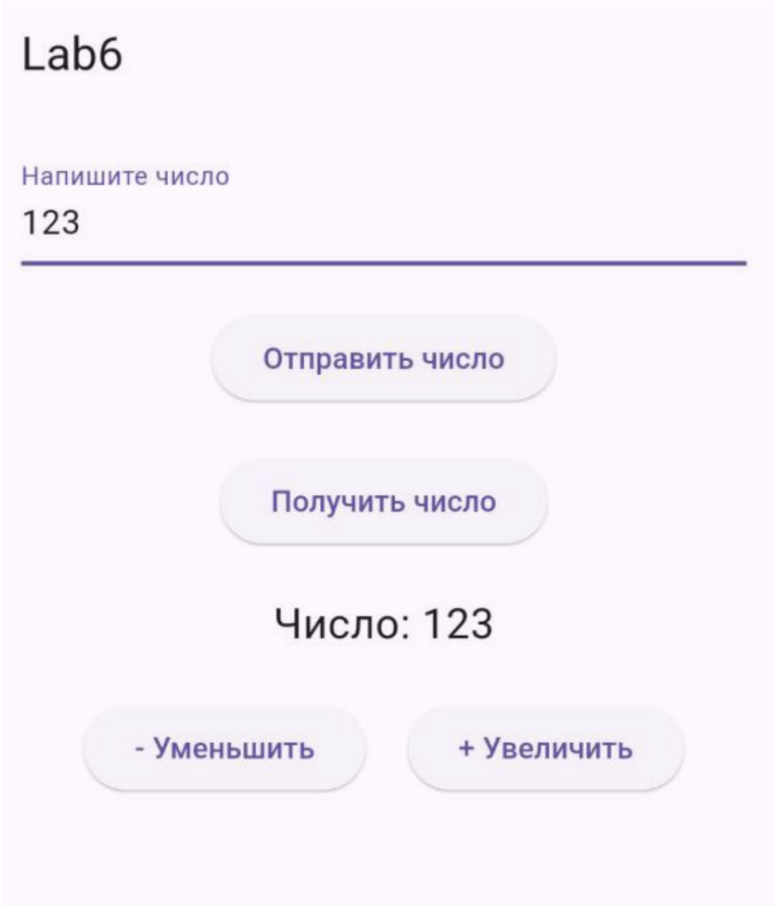

Рис. 1 — Интерфейс мобильного приложения

5 Выводы

В ходе выполнения лабораторной работы было успешно реализовано мобильное приложение и серверное решение, взаимодействующие посредством WebSockets. Достигнута поставленная цель по созданию системы для сохранения, изменения и чтения числовых данных на сервере через мобильное приложение. Полученные результаты демонстрируют возможность эффективного обмена данными в реальном времени между клиентом и сервером, используя протокол WebSockets.