Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное 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»

Летучка № 5

по курсу «Разработка мобильных приложений»

«SMTP-клиент»

Студент группы ИУ9-72Б Шемякин В.А.

Преподаватель Посевин Д. П.

Москва 2025

1 Задание

Реализовать SMTP-клиента для отправки сообщений.

2 Результаты

Исходный код программы представлен в листинге 1.

code.dart

```dart
import 'package:flutter/material.dart';
import 'package:mailer/mailer.dart';
import 'package:mailer/smtp_server.dart';
import 'package:image_picker/image_picker.dart';
import 'dart:io';
import 'package:path/path.dart' as path;

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

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      debugShowCheckedModeBanner: false,
      home: SmtpScreen(),
    );
  }
}

class Recipient {
  String name;
  String email;

  Recipient({required this.name, required this.email});
}

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

  @override
  State<SmtpScreen> createState() => _SmtpScreenState();
}

class _SmtpScreenState extends State<SmtpScreen> {
  final _formKey = GlobalKey<FormState>();
```

```
36    final _subjectController = TextEditingController();
37    final _bodyController = TextEditingController();
38
39    final List<Recipient> _recipients = [];
40    final _nameController = TextEditingController();
41    final _emailController = TextEditingController();
42
43    //                    SMTP
44    final _smtpHostController = TextEditingController(text: 'smtp.yandex.
       ru');
45    final _smtpPortController = TextEditingController(text: '465');
46    final _smtpUserController = TextEditingController(text: '
       shemyakinveniamin@yandex.ru');
47    final _smtpPasswordController = TextEditingController(text: '
       tfotiwnnspxypcqc');
48
49    //
50    File? _selectedImage;
51    final _imageAltController = TextEditingController(text: '
                                                          ');
52
53    bool _isSending = false;
54    int _currentProgress = 0;
55    int _totalProgress = 0;
56    bool _showSmtpSettings = false;
57    bool _includeImage = false;
58    bool _useImageUrl = true;
59    final _imageUrlController = TextEditingController();
60    final ImagePicker _imagePicker = ImagePicker();
61
62    @override
63    void dispose() {
64      _subjectController.dispose();
65      _bodyController.dispose();
66      _nameController.dispose();
67      _emailController.dispose();
68      _smtpHostController.dispose();
69      _smtpPortController.dispose();
70      _smtpUserController.dispose();
71      _smtpPasswordController.dispose();
72      _imageAltController.dispose();
73      super.dispose();
74    }
75
76    Future<void> _pickImage() async {
77      try {
```

```
78        final XFile? image = await _imagePicker.pickImage(
79          source: ImageSource.gallery,
80          maxWidth: 1024,
81          maxHeight: 1024,
82          imageQuality: 85,
83        );
84
85        if (image != null) {
86          setState(() {
87            _selectedImage = File(image.path);
88          });
89        }
90      } catch (e) {
91        _showError('
                              : $e');
92      }
93    }
94
95    void _removeImage() {
96      setState(() {
97        _selectedImage = null;
98      });
99    }
100
101   void _addRecipient() {
102     if (_nameController.text.isEmpty || _emailController.text.isEmpty) {
103       _showError('                                    email');
104       return;
105     }
106
107     if (!_emailController.text.contains('@')) {
108       _showError('                                              email');
109       return;
110     }
111
112     setState(() {
113       _recipients.add(Recipient(
114         name: _nameController.text,
115         email: _emailController.text,
116       ));
117       _nameController.clear();
118       _emailController.clear();
119     });
120   }
121
122   void _removeRecipient(int index) {
```

```
123     setState(() {
124       _recipients.removeAt(index);
125     });
126   }
127
128   void _clearAllRecipients() {
129     setState(() {
130       _recipients.clear();
131     });
132   }
133
134   void _showError(String message) {
135     ScaffoldMessenger.of(context).showSnackBar(
136       SnackBar(
137         content: Text(message),
138         backgroundColor: Colors.red,
139         duration: const Duration(seconds: 3),
140       ),
141     );
142   }
143
144   void _showSuccess(String message) {
145     ScaffoldMessenger.of(context).showSnackBar(
146       SnackBar(
147         content: Text(message),
148         backgroundColor: Colors.green,
149       ),
150     );
151   }
152
153   Future<void> _sendEmails() async {
154     if (_recipients.isEmpty) {
155       _showError('
                        ');
156       return;
157     }
158
159     if (!_formKey.currentState!.validate()) return;
160
161     //                        SMTP
162     if (_smtpHostController.text.isEmpty ||
163         _smtpUserController.text.isEmpty ||
164         _smtpPasswordController.text.isEmpty) {
165       _showError('                    SMTP                    ');
166       return;
167     }
```

```
168
169     //
170     if (_includeImage && _selectedImage == null) {
171       _showError('
                                                                  ');
172       return;
173     }
174
175     setState(() {
176       _isSending = true;
177       _currentProgress = 0;
178       _totalProgress = _recipients.length;
179     });
180
181     try {
182       final port = int.tryParse(_smtpPortController.text) ?? 465;
183
184       final smtpServer = SmtpServer(
185         _smtpHostController.text,
186         port: port,
187         ssl: true,
188         username: _smtpUserController.text,
189         password: _smtpPasswordController.text,
190       );
191
192       int successCount = 0;
193
194       for (int i = 0; i < _recipients.length; i++) {
195         final recipient = _recipients[i];
196
197         try {
198           final subject = '            , ${recipient.name}! ${
    _subjectController.text}';
199
200           final message = Message()
201             ..from = Address(_smtpUserController.text)
202             ..recipients.add(recipient.email)
203             ..subject = subject
204             ..html = _buildHtmlTemplate(
205               recipient.name,
206               _bodyController.text,
207               includeImage: _includeImage,
208               imageAlt: _imageAltController.text,
209             );
210
```

```
211           //                                                attachment

212           if (_includeImage && _selectedImage != null) {
213             //

214             final imageName = 'image_${DateTime.now().
    millisecondsSinceEpoch}.jpg';
215             message.attachments = [
216               FileAttachment(_selectedImage!)
217                 ..fileName = imageName
218                 ..location = Location.inline
219                 ..cid = '<image_cid>'
220             ];
221           }
222
223           await send(message, smtpServer);
224           successCount++;
225
226         } catch (e) {
227           print('                                        ${recipient.email
    }: $e');
228           //

229         }
230
231         setState(() {
232           _currentProgress = i + 1;
233         });
234
235         //                1

236         if (i < _recipients.length - 1) {
237           await Future.delayed(const Duration(seconds: 1));
238         }
239       }
240
241       if (!mounted) return;
242
243       if (successCount == _recipients.length) {
244         _showSuccess('                                                (
    $successCount           )');
245       } else {
246         _showSuccess('                    $successCount      ${
    _recipients.length}          ');
247       }
248
```

```
    } on MailerException catch (e) {
      if (!mounted) return;

      String errorMessage = '                                    : ${e.toString
    ()}';
      if (e.toString().contains('535')) {
        errorMessage = '''
                                          (535).
                  :
1.
2.

3.

4.
''';
      } else if (e.toString().contains('550')) {
        errorMessage = '            :
                       ';
      }

      _showError(errorMessage);
    } catch (e) {
      if (!mounted) return;
      _showError('                                  : $e');
    } finally {
      if (!mounted) return;
      setState(() {
        _isSending = false;
      });
    }
  }

  String _buildHtmlTemplate(
    String name,
    String bodyText, {
    bool includeImage = false,
    String imageAlt = 'Image',
  }) {
    String imageSection = '';

    if (includeImage) {
      imageSection = '''
      <div class="image-section">
        <img
          src="cid:image_cid"
```

```
290             alt="$imageAlt"
291             style="max-width: 100%; height: auto; border-radius: 8px;
    margin: 20px 0;"
292         >
293         <p style="text-align: center; color: #666; font-size: 14px;
    margin-top: 8px;">$imageAlt</p>
294       </div>
295       ''';
296     }
297
298     return '''
299 <html>
300   <head>
301     <meta charset="utf-8">
302     <style>
303       body {
304         font-family: Arial, sans-serif;
305         line-height: 1.6;
306         color: #333;
307         max-width: 600px;
308         margin: 0 auto;
309         padding: 20px;
310         background-color: #f5f5f5;
311       }
312       .container {
313         background: white;
314         border-radius: 10px;
315         overflow: hidden;
316         box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
317       }
318       .header {
319         background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
320         color: white;
321         padding: 30px 20px;
322         text-align: center;
323       }
324       .header h1 {
325         margin: 0;
326         font-size: 28px;
327         font-weight: 300;
328       }
329       .content {
330         padding: 30px;
331       }
332       .greeting {
333         font-size: 20px;
```

```
        font-weight: 600;
        color: #2c3e50;
        margin-bottom: 20px;
      }
      .message {
        background: #f8f9fa;
        padding: 20px;
        border-radius: 8px;
        border-left: 4px solid #667eea;
        font-size: 16px;
        line-height: 1.7;
      }
      .image-section {
        text-align: center;
        margin: 25px 0;
        padding: 15px;
        background: #f8f9fa;
        border-radius: 8px;
      }
      .footer {
        text-align: center;
        color: #666;
        font-size: 14px;
        margin-top: 30px;
        padding-top: 20px;
        border-top: 1px solid #eee;
      }
      .signature {
        font-style: italic;
        color: #7f8c8d;
        margin-top: 10px;
      }
    </style>
  </head>
  <body>
    <div class="container">
      <div class="header">
        <h1>             , $name!</h1>
      </div>
      <div class="content">
        <div class="message">
          ${bodyText.replaceAll('\n', '<br>')}
        </div>
        $imageSection
      </div>
    </div>
```

```
380   </body>
381 </html>
382 ''';
383   }
384
385   InputDecoration _dec(String label) {
386     return InputDecoration(
387       border: const OutlineInputBorder(),
388       labelText: label,
389       filled: _isSending,
390       fillColor: _isSending ? Colors.grey[100] : null,
391     );
392   }
393
394   @override
395   Widget build(BuildContext context) {
396     return Scaffold(
397       appBar: AppBar(
398         title: const Text('SMTP Mass Sender'),
399         backgroundColor: Colors.blue,
400         foregroundColor: Colors.white,
401         actions: [
402           if (_recipients.isNotEmpty)
403             IconButton(
404               icon: const Icon(Icons.delete_sweep),
405               onPressed: _clearAllRecipients,
406               tooltip: '
    ',
407             ),
408           IconButton(
409             icon: Icon(_showSmtpSettings ? Icons.settings : Icons.
    settings_outlined),
410             onPressed: () {
411               setState(() {
412                 _showSmtpSettings = !_showSmtpSettings;
413               });
414             },
415             tooltip: '                    SMTP',
416           ),
417         ],
418       ),
419       body: SafeArea(
420         child: SingleChildScrollView(
421           padding: const EdgeInsets.all(16),
422           child: Form(
423             key: _formKey,
```

```
424                 child: Column(
425                   crossAxisAlignment: CrossAxisAlignment.start,
426                   children: [
427                     //                         SMTP
428                     if (_showSmtpSettings) ...[
429                       Card(
430                         elevation: 2,
431                         child: Padding(
432                           padding: const EdgeInsets.all(16),
433                           child: Column(
434                             crossAxisAlignment: CrossAxisAlignment.start,
435                             children: [
436                               const Text(
437                                 '                         SMTP',
438                                 style: TextStyle(
439                                   fontWeight: FontWeight.bold,
440                                   fontSize: 16,
441                                 ),
442                               ),
443                               const SizedBox(height: 12),
444                               TextField(
445                                 controller: _smtpHostController,
446                                 decoration: const InputDecoration(
447                                   labelText: 'SMTP                ',
448                                   border: OutlineInputBorder(),
449                                 ),
450                               ),
451                               const SizedBox(height: 8),
452                               Row(
453                                 children: [
454                                   Expanded(
455                                     flex: 1,
456                                     child: TextField(
457                                       controller: _smtpPortController,
458                                       decoration: const InputDecoration(
459                                         labelText: '          ',
460                                         border: OutlineInputBorder(),
461                                       ),
462                                       keyboardType: TextInputType.number,
463                                     ),
464                                   ),
465                                   const SizedBox(width: 8),
466                                   Expanded(
467                                     flex: 3,
468                                     child: TextField(
469                                       controller: _smtpUserController,
```

```
                                decoration: const InputDecoration(
                                  labelText: '          /Email',
                                  border: OutlineInputBorder(),
                                ),
                              ),
                            ),
                          ],
                        ),
                        const SizedBox(height: 8),
                        TextField(
                          controller: _smtpPasswordController,
                          decoration: const InputDecoration(
                            labelText: '            ',
                            border: OutlineInputBorder(),
                          ),
                          obscureText: true,
                        ),
                        const SizedBox(height: 8),
                        const Text(
                          '       Yandex:

                        ',
                          style: TextStyle(
                            fontSize: 12,
                            color: Colors.grey,
                          ),
                        ),
                      ],
                    ),
                  ),
                ),
                const SizedBox(height: 16),
              ],

              //

              Card(
                elevation: 2,
                child: Padding(
                  padding: const EdgeInsets.all(16),
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      const Text(
                        '                                        ',
                        style: TextStyle(
```

```
513                             fontWeight: FontWeight.bold,
514                             fontSize: 16,
515                           ),
516                         ),
517                         const SizedBox(height: 12),
518                         Row(
519                           children: [
520                             Expanded(
521                               flex: 2,
522                               child: TextField(
523                                 controller: _nameController,
524                                 decoration: const InputDecoration(
525                                   labelText: '       ',
526                                   border: OutlineInputBorder(),
527                                   contentPadding: EdgeInsets.symmetric(
    horizontal: 12),
528                                 ),
529                               ),
530                             ),
531                             const SizedBox(width: 8),
532                             Expanded(
533                               flex: 3,
534                               child: TextField(
535                                 controller: _emailController,
536                                 decoration: const InputDecoration(
537                                   labelText: 'Email',
538                                   border: OutlineInputBorder(),
539                                   contentPadding: EdgeInsets.symmetric(
    horizontal: 12),
540                                 ),
541                                 keyboardType: TextInputType.emailAddress
    ,
542                               ),
543                             ),
544                             const SizedBox(width: 8),
545                             ElevatedButton(
546                               onPressed: _addRecipient,
547                               style: ElevatedButton.styleFrom(
548                                 backgroundColor: Colors.green,
549                                 foregroundColor: Colors.white,
550                                 padding: const EdgeInsets.symmetric(
    horizontal: 16, vertical: 16),
551                               ),
552                               child: const Icon(Icons.add),
553                             ),
554                           ],
```

```
555                         ),
556                       ],
557                     ),
558                   ),
559                 ),
560                 const SizedBox(height: 16),
561
562                 //
563                 Card(
564                   elevation: 2,
565                   child: ConstrainedBox(
566                     constraints: BoxConstraints(
567                       maxHeight: MediaQuery.of(context).size.height *
    0.3,
568                     ),
569                     child: Column(
570                       children: [
571                         Padding(
572                           padding: const EdgeInsets.all(16),
573                           child: Row(
574                             children: [
575                               const Text(
576                                 '                                    ',
577                                 style: TextStyle(
578                                   fontWeight: FontWeight.bold,
579                                   fontSize: 16,
580                                 ),
581                               ),
582                               const Spacer(),
583                               Text(
584                                 '          : ${_recipients.length}',
585                                 style: TextStyle(
586                                   color: Colors.grey[600],
587                                   fontWeight: FontWeight.bold,
588                                 ),
589                               ),
590                             ],
591                           ),
592                         ),
593                         Expanded(
594                           child: _recipients.isEmpty
595                               ? const Center(
596                                   child: Column(
597                                     mainAxisAlignment: MainAxisAlignment
    .center,
598                                     children: [
```

```
599                                     Icon(Icons.people_outline, size:
    64, color: Colors.grey),
600                                     SizedBox(height: 8),
601                                     Text(
602                                       '                                ',
603                                       style: TextStyle(color: Colors.
    grey),
604                                     ),
605                                   ],
606                                 ),
607                               )
608                             : ListView.builder(
609                                 shrinkWrap: true,
610                                 itemCount: _recipients.length,
611                                 itemBuilder: (context, index) {
612                                   final recipient = _recipients[index
    ];
613                                   return ListTile(
614                                     leading: CircleAvatar(
615                                       backgroundColor: Colors.blue.
    shade100,
616                                       child: Text(
617                                         recipient.name.isNotEmpty
618                                             ? recipient.name[0].
    toUpperCase()
619                                             : '?',
620                                       ),
621                                     ),
622                                     title: Text(recipient.name),
623                                     subtitle: Text(recipient.email),
624                                     trailing: IconButton(
625                                       icon: const Icon(Icons.delete,
    color: Colors.red),
626                                       onPressed: () =>
    _removeRecipient(index),
627                                     ),
628                                   );
629                                 },
630                               ),
631                           ),
632                         ],
633                       ),
634                     ),
635                   ),
636                   const SizedBox(height: 16),
637
```

```
638                 //
639                 //                                                    (
                                ):
640
641 //                                  -              Expanded
        ConstrainedBox
642 Card(
643   elevation: 2,
644   child: Padding(
645     padding: const EdgeInsets.all(16),
646     child: Column(
647       children: [
648         const Text(
649           '                    ',
650           style: TextStyle(
651             fontWeight: FontWeight.bold,
652             fontSize: 16,
653           ),
654         ),
655         const SizedBox(height: 12),
656         TextFormField(
657           controller: _subjectController,
658           decoration: _dec('                                          (
                                  )'),
659           enabled: !_isSending,
660         ),
661         const SizedBox(height: 12),
662
663         //
664         Card(
665           color: Colors.grey[50],
666           child: Padding(
667             padding: const EdgeInsets.all(12),
668             child: Column(
669               children: [
670                 Row(
671                   children: [
672                     const Icon(Icons.image, size: 20),
673                     const SizedBox(width: 8),
674                     const Text(
675                       '                                    ',
676                       style: TextStyle(fontWeight: FontWeight.bold),
677                     ),
678                     const Spacer(),
679                     Switch(
680                       value: _includeImage,
```

```
681                       onChanged: _isSending ? null : (value) {
682                         setState(() {
683                           _includeImage = value;
684                         });
685                       },
686                     ),
687                   ],
688                 ),
689                 if (_includeImage) ...[
690                   const SizedBox(height: 12),
691
692                   //
693                   if (_selectedImage != null) ...[
694                     Container(
695                       height: 150,
696                       width: double.infinity,
697                       decoration: BoxDecoration(
698                         borderRadius: BorderRadius.circular(8),
699                         image: DecorationImage(
700                           image: FileImage(_selectedImage!),
701                           fit: BoxFit.cover,
702                         ),
703                       ),
704                     ),
705                     const SizedBox(height: 8),
706                     Row(
707                       children: [
708                         Expanded(
709                           child: OutlinedButton.icon(
710                             icon: const Icon(Icons.delete),
711                             label: const Text('
                    '),
712                             onPressed: _removeImage,
713                           ),
714                         ),
715                       ],
716                     ),
717                     const SizedBox(height: 8),
718                   ] else ...[
719                     ElevatedButton.icon(
720                       icon: const Icon(Icons.photo_library),
721                       label: const Text('
                        '),
722                       onPressed: _pickImage,
723                       style: ElevatedButton.styleFrom(
724                         backgroundColor: Colors.blue,
```

```
725                       foregroundColor: Colors.white,
726                       minimumSize: const Size(double.infinity, 50),
727                     ),
728                   ),
729                 ],
730
731                 const SizedBox(height: 8),
732                 TextField(
733                   controller: _imageAltController,
734                   decoration: const InputDecoration(
735                     labelText: '                                    ',
736                     border: OutlineInputBorder(),
737                     hintText: '                          ALT        ',
738                   ),
739                 ),
740                 const SizedBox(height: 8),
741                 const Text(
742                   '
                                                          ',
743                   style: TextStyle(fontSize: 12, color: Colors.grey),
744                   textAlign: TextAlign.center,
745                 ),
746               ],
747             ],
748           ),
749         ),
750       ),
751       const SizedBox(height: 12),
752
753       //                                                    -
                                                          Expanded
754       Container(
755         height: 200, //
756         child: TextFormField(
757           controller: _bodyController,
758           decoration: _dec('
                     HTML'),
759           maxLines: null,
760           expands: true,
761           validator: (v) => (v == null || v.isEmpty) ? '
               ' : null,
762           enabled: !_isSending,
763         ),
764       ),
765     ],
766   ),
```

```
767     ),
768   ),
769
770                 const SizedBox(height: 16),
771
772                 if (_isSending) ...[
773                   LinearProgressIndicator(
774                     value: _totalProgress > 0 ? _currentProgress /
    _totalProgress : 0,
775                     backgroundColor: Colors.grey[300],
776                     valueColor: AlwaysStoppedAnimation<Color>(Colors.
    blue.shade400),
777                   ),
778                   const SizedBox(height: 8),
779                   Text(
780                     '                    $_currentProgress
    $_totalProgress          ',
781                     style: const TextStyle(fontSize: 14),
782                   ),
783                   const SizedBox(height: 16),
784                 ],
785
786                 SizedBox(
787                   width: double.infinity,
788                   child: ElevatedButton(
789                     onPressed: _isSending ? null : _sendEmails,
790                     style: ElevatedButton.styleFrom(
791                       backgroundColor: _recipients.isEmpty ? Colors.grey
     : Colors.blue,
792                       foregroundColor: Colors.white,
793                       padding: const EdgeInsets.symmetric(vertical: 16),
794                     ),
795                     child: _isSending
796                         ? const Row(
797                             mainAxisSize: MainAxisSize.min,
798                             children: [
799                               SizedBox(
800                                 height: 20,
801                                 width: 20,
802                                 child: CircularProgressIndicator(
803                                   strokeWidth: 2,
804                                   valueColor: AlwaysStoppedAnimation<
    Color>(Colors.white),
805                                 ),
806                               ),
807                               SizedBox(width: 12),
```

```
                                    Text('                    ...'),
                                  ],
                                )
                              : Text(
                                  _recipients.isEmpty
                                      ? '
                                    ,
                                      : '                                    (${
    _recipients.length})',
                                ),
                          ),
                      ),
                      const SizedBox(height: 16),
                    ],
                  ),
                ),
              ),
            ),
          ),
        ),
      );
  }
}

void main() {
  runApp(const SmtpApp());
}
```

Результат запуска представлен на рисунке 1.

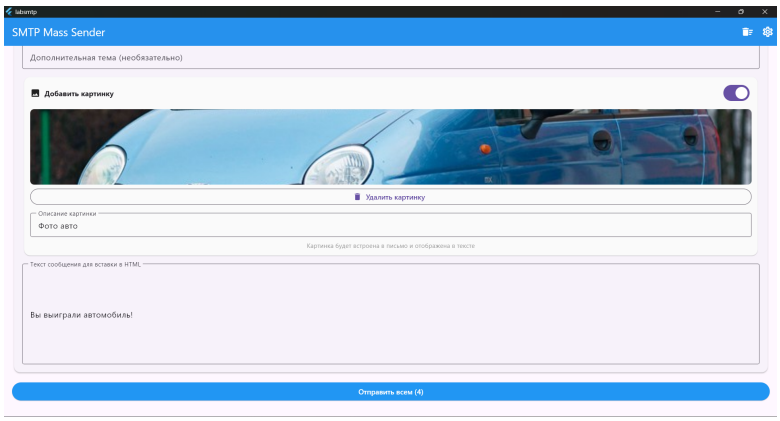

Рис. 1 — Результат

.