УТВЕРЖДЕН
RU.17701729.04.05-01 12 01–2

«HSE COFFEE» - КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ДЛЯ ЗНАКОМСТВ НА ANDROID & IOS

МОБИЛЬНЫЙ КЛИЕНТ

Текст программы

RU.17701729.04.05-01 12 01–2

Листов 83

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	RU.17701729.04.05-01 12 01-2

Москва 2021

ОГЛАВЛЕНИЕ

1. ТЕКСТ ПРОГРАММЫ 4
1.1. contact.dart 4
1.2. contact.g.dart 4
1.3. degree.dart 5
1.4. faculty.dart 5
1.5. gender.dart 5
1.6. meet.dart 5
1.7. meet.g.dart 6
1.8. meet_status.dart 7
1.9. search_params.dart 7
1.10. search_params.g.dart 8
1.11. user.dart 10
1.12. user.g.dart 11
1.13. api.dart 13
1.14. auth.dart 19
1.15. event_wrapper.dart 20
1.16. user_storage.dart 20
1.17. main.dart 21
1.18. router_auth.dart 22
1.19. auth_code.dart 22
1.20. auth_contacts.dart 26
1.21. auth_email.dart 29
1.22. auth_faculty.dart 32
1.23. auth_gender.dart 35
1.24. auth_name.dart 39
1.25. auth_photo.dart 42
1.26. header.dart 45
1.27. home.dart 46
1.28. home_cabinet.dart 49
1.29. home_cabinet_header.dart 50
1.30. home_cabinet_settings.dart 53

1.31. home_find_start.dart....57
1.32. home_loading.dart....63
1.33. home_meets.dart....66
1.34. home_person_info.dart....69
1.35. splash.dart....72
1.36. splash_background.dart....73
1.37. auth_faculty_items.dart....75
1.38. button_continue.dart....76
1.39. capsule_widget.dart....77
1.40. dialog_loading.dart....78
1.41. edu_fields.dart....78
1.42. text_field_wrapper.dart....80
1.43. toggle_button_gender.dart....81

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.05-01 12 01-2				
Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

1. ТЕКСТ ПРОГРАММЫ

1.1. contact.dart

```
import 'package:json_annotation/json_annotation.dart';

part 'contact.g.dart';

@JsonSerializable()
class Contact {
  String name;
  String value;

  Contact(this.name, this.value);

  factory Contact.fromJson(Map<String, dynamic> json) =>
      _$ContactFromJson(json);

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

  static Contact createVk(String login) {
    return Contact("vk", login);
  }

  static Contact createTelegram(String login) {
    return Contact("tg", login);
  }

  static Contact createInstagram(String login) {
    return Contact("inst", login);
  }
}
```

1.2. contact.g.dart

```
part of 'contact.dart';

// **************************************************************************
// JsonSerializableGenerator
// **************************************************************************

Contact _$ContactFromJson(Map<String, dynamic> json) {
  return Contact(
    json['name'] as String,
    json['value'] as String,
  );
}

Map<String, dynamic> _$ContactToJson(Contact instance) => <String, dynamic>{
      'name': instance.name,
      'value': instance.value,
    };
```

1.3. degree.dart

```
enum Degree { NONE, BACHELOR, MAGISTRACY, SPECIALTY, POSTGRADUATE }
```

1.4. faculty.dart

```
enum Faculty {
  LYCEUM,
  MATH,
  ECONOMY,
  ELECTRONIC,
  COMPUTER,
  BUSINESS,
  LAWYER,
  JURISPRUDENCE,
  HUMANITARIAN,
  SOCIAL,
  MEDIA,
  WORLD_ECONOMY,
  MIEF,
  PHYSICS,
  CITY,
  CHEMICAL,
  BIOLOGY,
  GEOGRAPHY,
  LANGUAGE,
  STATISTIC,
  BANK,
  NONE
}
```

1.5. gender.dart

```
enum Gender { MALE, FEMALE, NONE }
```

1.6. meet.dart

```
import 'package:hse_coffee/data/meet_status.dart';
import 'package:hse_coffee/data/user.dart';
import 'package:json_annotation/json_annotation.dart';

part 'meet.g.dart';

@JsonSerializable()
class Meet {
  User user1;
  User user2;

  MeetStatus meetStatus;
  DateTime createdDate;
  DateTime expiresDate;

  Meet(this.user1, this.user2, this.meetStatus, this.createdDate,
      this.expiresDate);
  factory Meet.fromJson(Map<String, dynamic> json) => _$MeetFromJson(json);

  Map<String, dynamic> toJson() => _$MeetToJson(this);
}
```

1.7. meet.g.dart

```
part of 'meet.dart';

// **************************************************************************
// JsonSerializableGenerator
// **************************************************************************

Meet _$MeetFromJson(Map<String, dynamic> json) {
  return Meet(
    json['user1'] == null
        ? null
        : User.fromJson(json['user1'] as Map<String, dynamic>),
    json['user2'] == null
        ? null
        : User.fromJson(json['user2'] as Map<String, dynamic>),
    _$enumDecodeNullable(_$MeetStatusEnumMap, json['meetStatus']),
    json['createdDate'] == null
        ? null
        : DateTime.fromMillisecondsSinceEpoch(json['createdDate'] as int),
    json['expiresDate'] == null
        ? null
        : DateTime.fromMillisecondsSinceEpoch(json['expiresDate'] as int),
  );
}

Map<String, dynamic> _$MeetToJson(Meet instance) => <String, dynamic>{
      'user1': instance.user1,
      'user2': instance.user2,
      'meetStatus': _$MeetStatusEnumMap[instance.meetStatus],
      'createdDate': instance.createdDate?.toIso8601String(),
      'expiresDate': instance.expiresDate?.toIso8601String(),
    };

T _$enumDecode<T>(
  Map<T, dynamic> enumValues,
  dynamic source, {
  T unknownValue,
}) {
  if (source == null) {
    throw ArgumentError('A value must be provided. Supported values: '
        '${enumValues.values.join(', ')}');
  }

  final value = enumValues.entries
      .singleWhere((e) => e.value == source, orElse: () => null)
      ?.key;
  if (value == null && unknownValue == null) {
    throw ArgumentError('`$source` is not one of the supported values: '
        '${enumValues.values.join(', ')}');
  }
  return value ?? unknownValue;
}

T _$enumDecodeNullable<T>(
  Map<T, dynamic> enumValues,
  dynamic source, {
  T unknownValue,
}) {
  if (source == null) {
    return null;
  }
  return _$enumDecode<T>(enumValues, source, unknownValue: unknownValue);
}

const _$MeetStatusEnumMap = {
  MeetStatus.ACTIVE: 'ACTIVE',
  MeetStatus.FINISHED: 'FINISHED',
  MeetStatus.SEARCH: 'SEARCH',
  MeetStatus.NONE: 'NONE',
  MeetStatus.ERROR: 'ERROR',
};
```

1.8. meet_status.dart

```
enum MeetStatus { ACTIVE, FINISHED, SEARCH, NONE, ERROR }
```

1.9. search_params.dart

```
import 'package:hse_coffee/data/faculty.dart';
import 'package:json_annotation/json_annotation.dart';

import 'degree.dart';
import 'gender.dart';

part 'search_params.g.dart';

@JsonSerializable()
class SearchParams {
  Set<Gender> genders;
  Set<Faculty> faculties;
  Set<Degree> degrees;

  int minCourse;
  int maxCourse;

  SearchParams(this.genders, this.faculties, this.degrees, this.minCourse,
      this.maxCourse);
  factory SearchParams.fromJson(Map<String, dynamic> json) =>
      _$SearchParamsFromJson(json);

  Map<String, dynamic> toJson() => _$SearchParamsToJson(this);
}
```

1.10. search_params.g.dart

```
part of 'search_params.dart';

// **************************************************************************
// JsonSerializableGenerator
// **************************************************************************

SearchParams _$SearchParamsFromJson(Map<String, dynamic> json) {
  return SearchParams(
    (json['genders'] as List)
        ?.map((e) => _$enumDecodeNullable(_$GenderEnumMap, e))
        ?.toSet(),
    (json['faculties'] as List)
        ?.map((e) => _$enumDecodeNullable(_$FacultyEnumMap, e))
        ?.toSet(),
    (json['degrees'] as List)
        ?.map((e) => _$enumDecodeNullable(_$DegreeEnumMap, e))
        ?.toSet(),
    json['minCourse'] as int,
    json['maxCourse'] as int,
  );
}

Map<String, dynamic> _$SearchParamsToJson(SearchParams instance) =>
    <String, dynamic>{
      'genders': instance.genders?.map((e) => _$GenderEnumMap[e])?.toList(),
      'faculties':
          instance.faculties?.map((e) => _$FacultyEnumMap[e])?.toList(),
      'degrees': instance.degrees?.map((e) => _$DegreeEnumMap[e])?.toList(),
      'minCourse': instance.minCourse,
      'maxCourse': instance.maxCourse,
    };

T _$enumDecode<T>(
  Map<T, dynamic> enumValues,
  dynamic source, {
  T unknownValue,
}) {
  if (source == null) {
    throw ArgumentError('A value must be provided. Supported values: '
        '${enumValues.values.join(', ')}');
  }

  final value = enumValues.entries
      .singleWhere((e) => e.value == source, orElse: () => null)
      ?.key;
  if (value == null && unknownValue == null) {
    throw ArgumentError('`$source` is not one of the supported values: '
        '${enumValues.values.join(', ')}');
  }
  return value ?? unknownValue;
}

T _$enumDecodeNullable<T>(
  Map<T, dynamic> enumValues,
  dynamic source, {
  T unknownValue,
}) {
  if (source == null) {
    return null;
  }
  return _$enumDecode<T>(enumValues, source, unknownValue: unknownValue);
}

const _$GenderEnumMap = {
  Gender.MALE: 'MALE',
  Gender.FEMALE: 'FEMALE',
  Gender.NONE: 'NONE',
};

const _$FacultyEnumMap = {
  Faculty.LYCEUM: 'LYCEUM',
  Faculty.MATH: 'MATH',
  Faculty.ECONOMY: 'ECONOMY',
  Faculty.ELECTRONIC: 'ELECTRONIC',
  Faculty.COMPUTER: 'COMPUTER',
  Faculty.BUSINESS: 'BUSINESS',
  Faculty.LAWYER: 'LAWYER',
  Faculty.JURISPRUDENCE: 'JURISPRUDENCE',
  Faculty.HUMANITARIAN: 'HUMANITARIAN',
  Faculty.SOCIAL: 'SOCIAL',
  Faculty.MEDIA: 'MEDIA',
  Faculty.WORLD_ECONOMY: 'WORLD_ECONOMY',
  Faculty.MIEF: 'MIEF',
  Faculty.PHYSICS: 'PHYSICS',
  Faculty.CITY: 'CITY',
  Faculty.CHEMICAL: 'CHEMICAL',
  Faculty.BIOLOGY: 'BIOLOGY',
  Faculty.GEOGRAPHY: 'GEOGRAPHY',
  Faculty.LANGUAGE: 'LANGUAGE',
  Faculty.STATISTIC: 'STATISTIC',
  Faculty.BANK: 'BANK',
  Faculty.NONE: 'NONE',
};

const _$DegreeEnumMap = {
  Degree.NONE: 'NONE',
  Degree.BACHELOR: 'BACHELOR',
  Degree.MAGISTRACY: 'MAGISTRACY',
  Degree.SPECIALTY: 'SPECIALTY',
  Degree.POSTGRADUATE: 'POSTGRADUATE',
};
```

1.11. user.dart

```
import 'package:json_annotation/json_annotation.dart';

import 'contact.dart';
import 'degree.dart';
import 'faculty.dart';
import 'gender.dart';

part 'user.g.dart';

@JsonSerializable(explicitToJson: true)
class User {
  String email;
  String firstName;
  String lastName;
  Gender gender;
  Degree degree;
  Faculty faculty;
  Set<Contact> contacts;
  int course;
  String photoUri;
  String aboutMe;

  User(
      {this.email,
      this.firstName,
      this.lastName,
      this.gender,
      this.degree,
      this.faculty,
      this.contacts,
      this.course,
      this.photoUri,
      this.aboutMe});

  factory User.fromJson(Map<String, dynamic> json) => _$UserFromJson(json);

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

  String getFullName() {
    return firstName + " " + lastName;
  }

  String getVk() {
    return _getValueContact("vk");
  }

  String getInst() {
    return _getValueContact("inst");
  }
  String getTelegram() {
    return "${_getValueContact("tg")}";
  }

  String _getValueContact(String name) {
    Contact contact;

    try {
      contact = contacts.firstWhere((element) => element.name == name);
    } catch (StateError) {
      return "";
    }

    return contact.value;
  }
}
```

1.12. user.g.dart

```
part of 'user.dart';

// **************************************************************************
// JsonSerializableGenerator
// **************************************************************************

User _$UserFromJson(Map<String, dynamic> json) {
  return User(
    email: json['email'] as String,
    firstName: json['firstName'] as String,
    lastName: json['lastName'] as String,
    gender: _$enumDecodeNullable(_$GenderEnumMap, json['gender']),
    degree: _$enumDecodeNullable(_$DegreeEnumMap, json['degree']),
    faculty: _$enumDecodeNullable(_$FacultyEnumMap, json['faculty']),
    contacts: (json['contacts'] as List)
        ?.map((e) =>
            e == null ? null : Contact.fromJson(e as Map<String, dynamic>))
        ?.toSet(),
    course: json['course'] as int,
    photoUri: json['photoUri'] as String,
    aboutMe: json['aboutMe'] as String,
  );
}

Map<String, dynamic> _$UserToJson(User instance) => <String, dynamic>{
      'email': instance.email,
      'firstName': instance.firstName,
      'lastName': instance.lastName,
      'gender': _$GenderEnumMap[instance.gender],
      'degree': _$DegreeEnumMap[instance.degree],
      'faculty': _$FacultyEnumMap[instance.faculty],
      'contacts': instance.contacts?.map((e) => e?.toJson())?.toList(),
      'course': instance.course,
      'photoUri': instance.photoUri,
      'aboutMe': instance.aboutMe
    };

T _$enumDecode<T>(
  Map<T, dynamic> enumValues,
  dynamic source, {
  T unknownValue,
}) {
  if (source == null) {
    throw ArgumentError('A value must be provided. Supported values: '
        '${enumValues.values.join(', ')}');
  }

  final value = enumValues.entries
      .singleWhere((e) => e.value == source, orElse: () => null)
      ?.key;

  if (value == null && unknownValue == null) {
    throw ArgumentError('`$source` is not one of the supported values: '
        '${enumValues.values.join(', ')}');
  }
  return value ?? unknownValue;
}

T _$enumDecodeNullable<T>(
  Map<T, dynamic> enumValues,
  dynamic source, {
  T unknownValue,
}) {
  if (source == null) {
    return null;
  }
  return _$enumDecode<T>(enumValues, source, unknownValue: unknownValue);
}

const _$GenderEnumMap = {
  Gender.MALE: 'MALE',
  Gender.FEMALE: 'FEMALE',
  Gender.NONE: 'NONE',
};

const _$DegreeEnumMap = {
  Degree.NONE: 'NONE',
  Degree.BACHELOR: 'BACHELOR',
  Degree.MAGISTRACY: 'MAGISTRACY',
  Degree.SPECIALTY: 'SPECIALTY',
  Degree.POSTGRADUATE: 'POSTGRADUATE',
};

const _$FacultyEnumMap = {
  Faculty.LYCEUM: 'LYCEUM',
  Faculty.MATH: 'MATH',
  Faculty.ECONOMY: 'ECONOMY',
  Faculty.ELECTRONIC: 'ELECTRONIC',
  Faculty.COMPUTER: 'COMPUTER',
  Faculty.BUSINESS: 'BUSINESS',
  Faculty.LAWYER: 'LAWYER',
  Faculty.JURISPRUDENCE: 'JURISPRUDENCE',
  Faculty.HUMANITARIAN: 'HUMANITARIAN',
  Faculty.SOCIAL: 'SOCIAL',
  Faculty.MEDIA: 'MEDIA',
  Faculty.WORLD_ECONOMY: 'WORLD_ECONOMY',
  Faculty.MIEF: 'MIEF',
  Faculty.PHYSICS: 'PHYSICS',
  Faculty.CITY: 'CITY',
  Faculty.CHEMICAL: 'CHEMICAL',
  Faculty.BIOLOGY: 'BIOLOGY',
  Faculty.GEOGRAPHY: 'GEOGRAPHY',
  Faculty.LANGUAGE: 'LANGUAGE',
  Faculty.STATISTIC: 'STATISTIC',
  Faculty.BANK: 'BANK',
  Faculty.NONE: 'NONE',
};
```

1.13. api.dart

```
import 'dart:convert';
import 'dart:io';
import 'dart:math';

import 'package:dio/dio.dart' as dio;
import 'package:hse_coffee/business_logic/auth.dart';
import 'package:hse_coffee/business_logic/event_wrapper.dart';
import 'package:hse_coffee/data/meet.dart';
import 'package:hse_coffee/data/meet_status.dart';
import 'package:hse_coffee/data/search_params.dart';
import 'package:hse_coffee/data/user.dart';
import 'package:http/http.dart' as http;

class Api {
  static const String _Ip = "http://188.120.233.197";

  static int _number = 0;

  static updateImage() {
    _number = Random().nextInt(99999);
  }

  static const Map<String, String> headers = {
    'Content-type': 'application/json',
    'Accept': 'application/json',
  };

  static String getImageUrlByUser(User user) {
    return _Ip + "/" + user.photoUri + "?$_number";
  }

  static Future<EventWrapper<bool>> sendCode(String email) async {
    print("/api/code?email=$email");
    final response = await http.post('$_Ip/api/code?email=$email');

    print("Код: ${response.statusCode}");

    if (response.statusCode == 200) {
      return EventWrapper(response.statusCode, true, "Удачно");
    }

    if (response.body != null)
      return EventWrapper(response.statusCode, null, response.body);

    return EventWrapper(
        response.statusCode, null, "Связь с сервером не была установлена");
  }

  static Future<EventWrapper<bool>> confirmCode(
      String code, String email) async {
    var fingerprint = await Auth.getFingerprint();

    print(
        "/api/confirm. Email: $email; fingerprint: $fingerprint; code: $code");

    final response = await http.post('$_Ip/api/confirm',
        body: {"email": email, "fingerprint": fingerprint, "code": code});

    print("Код: ${response.statusCode}");

    if (response.statusCode == 200) {
      await Auth.saveDataByJson(email, response.body);
      return EventWrapper(response.statusCode, true, "Удачно");
    }

    if (response.body != null)
      return EventWrapper(response.statusCode, null, response.body);

    return EventWrapper(
        response.statusCode, null, "Связь с сервером не была установлена");
  }

  static Future<EventWrapper<MeetStatus>> search(
      SearchParams searchParams) async {
    final accessToken = await Auth.getAccessToken();
    print("POST: /api/search/. accessToken = [$accessToken]");

    var jsonEnc = json.encode(searchParams.toJson());

    print(jsonEnc);

    final response = await http.post('$_Ip/api/search/$accessToken',
        body: jsonEnc, headers: headers);

    print("Код: ${response.statusCode}");

    if (response.statusCode == 200) {
      return EventWrapper(
          response.statusCode, _getMeetByResponse(response), "Удачно");
    }

    if ((response.statusCode == 403 || response.statusCode == 401) &&
        (await _updateTokens()) == true) {
      return search(searchParams);
    }

    if (response.body != null)
      return EventWrapper(response.statusCode, null, response.body);

    return EventWrapper(
        response.statusCode, null, "Связь с сервером не была установлена");
  }

  static Future<EventWrapper<MeetStatus>> cancelSearch() async {
    final accessToken = await Auth.getAccessToken();
    print("DELETE: /api/meet. accessToken = [$accessToken]");

    final response =
        await http.delete('$_Ip/api/meet/$accessToken', headers: headers);

    print("Код: ${response.statusCode}");

    if (response.statusCode == 200) {
      return EventWrapper(
          response.statusCode, _getMeetByResponse(response), "Удачно");
    }

    if ((response.statusCode == 403 || response.statusCode == 401) &&
        (await _updateTokens()) == true) {
      return cancelSearch();
    }

    if (response.body != null)
      return EventWrapper(response.statusCode, null, response.body);

    return EventWrapper(
        response.statusCode, null, "Связь с сервером не была установлена");
  }

  static MeetStatus _getMeetByResponse(http.Response response) {
    var meetStatus;

    try {
      meetStatus = MeetStatus.values.firstWhere((element) =>
          element.toString().toLowerCase().replaceAll("meetstatus.", "") ==
          response.body.toLowerCase());
    } catch (StateError) {
      print("Ошибка при получении meetStatus из ${response.body}");
      meetStatus = MeetStatus.NONE;
    }

    return meetStatus;
  }
```

```
static Future<EventWrapper<bool>> setUser(User user) async {
  final accessToken = await Auth.getAccessToken();
  print("PUT: /api/user/settings/. accessToken = [$accessToken]");

  var jsonEnc = json.encode(user.toJson());

  print(jsonEnc);

  final response = await http.put('$_Ip/api/user/settings/$accessToken',
      body: jsonEnc, headers: headers);

  print("Код: ${response.statusCode}");

  if (response.statusCode == 200) {
    return EventWrapper(response.statusCode, true, "Удачно");
  }

  if ((response.statusCode == 403 || response.statusCode == 401) &&
      (await _updateTokens()) == true) {
    return setUser(user);
  }

  if (response.body != null)
    return EventWrapper(response.statusCode, null, response.body);

  return EventWrapper(
      response.statusCode, null, "Связь с сервером не была установлена");
}

static Future<EventWrapper<bool>> setPhoto(User user, File file) async {
  final accessToken = await Auth.getAccessToken();
  print("POST: api/user/image/. accessToken = [$accessToken]");

  var jsonEnc = json.encode(user.toJson());

  print(jsonEnc);

  dio.FormData formData = new dio.FormData.fromMap(
      {"image": await dio.MultipartFile.fromFile(file.path)});

  final response = await dio.Dio()
      .post("$_Ip/api/user/image/$accessToken", data: formData);

  print("Код: ${response.statusCode}");

  if (response.statusCode == 200) {
    return EventWrapper(response.statusCode, true, "Удачно");
  }

  if ((response.statusCode == 403 || response.statusCode == 401) &&
      (await _updateTokens()) == true) {
    return setPhoto(user, file);
  }

  if (response.data != null)
    return EventWrapper(response.statusCode, null, response.data);
  return EventWrapper(
      response.statusCode, null, "Связь с сервером не была установлена");
}

static Future<EventWrapper<User>> getUser() async {
  final accessToken = await Auth.getAccessToken();
  print("GET: /api/user. accessToken = [$accessToken]");

  final response = await http.get('$_Ip/api/user/settings/$accessToken');

  print("Код: ${response.statusCode}");

  if (response.statusCode == 200) {
    var user = User.fromJson(jsonDecode(response.body));

    print("GET 200: ${user.toString()}");
    return EventWrapper(response.statusCode, user, "Удачно");
  }

  if ((response.statusCode == 403 || response.statusCode == 401) &&
      (await _updateTokens()) == true) {
    return getUser();
  }

  if (response.body != null)
    return EventWrapper(response.statusCode, null, response.body);

  return EventWrapper(
      response.statusCode, null, "Связь с сервером не была установлена");
}

static Future<EventWrapper<Meet>> getMeet() async {
  final accessToken = await Auth.getAccessToken();
  print("GET: /api/meet. accessToken = [$accessToken]");

  final response = await http.get('$_Ip/api/meet/$accessToken');

  print("Код: ${response.statusCode}");

  if (response.statusCode == 200) {
    Meet meet = Meet.fromJson(json.decode(response.body));

    print("GET 200: ${meet.toJson()}");
    return EventWrapper(response.statusCode, meet, "Удачно");
  }

  if ((response.statusCode == 403 || response.statusCode == 401) &&
      (await _updateTokens()) == true) {
    return getMeet();
  }

  if (response.body != null)
    return EventWrapper(response.statusCode, null, response.body);

  return EventWrapper(
      response.statusCode, null, "Связь с сервером не была установлена");
}

static Future<EventWrapper<List<Meet>>> getMeets() async {
  final accessToken = await Auth.getAccessToken();
  print("GET: /api/meets. accessToken = [$accessToken]");

  final response = await http.get('$_Ip/api/meets/$accessToken');

  print("Код: ${response.statusCode}");

  if (response.statusCode == 200) {
    Iterable l = json.decode(response.body);
    List<Meet> meets =
        List<Meet>.from(l.map((model) => Meet.fromJson(model)));

    print("GET 200: ${meets.toString()}");
    return EventWrapper(response.statusCode, meets, "Удачно");
  }

  if ((response.statusCode == 403 || response.statusCode == 401) &&
      (await _updateTokens()) == true) {
    return getMeets();
  }

  if (response.body != null)
    return EventWrapper(response.statusCode, null, response.body);

  return EventWrapper(
      response.statusCode, null, "Связь с сервером не была установлена");
}

static Future<bool> _updateTokens() async {
  var email = await Auth.getEmail();
  var refreshToken = await Auth.getRefreshToken();
  var fingerprint = await Auth.getFingerprint();

  print("/api/refresh. "
      "fingerprint = [$fingerprint], "
      "email = [$email], "
      "refreshToken = [$refreshToken]");

  final response = await http.post("$_Ip/api/refresh?", body: {
    "fingerprint": fingerprint,
    "email": email,
    "refreshToken": refreshToken
  });

  print("Код: ${response.statusCode}");

  if (response.statusCode == 200) {
    await Auth.saveDataByJson(email, response.body);

    return true;
  }
  return false;
}
}
```

1.14. auth.dart

```
import 'dart:convert';
import 'dart:io';

import 'package:device_info/device_info.dart';
import 'package:flutter_secure_storage/flutter_secure_storage.dart';

class Auth {
  static final String emailKey = "email";
  static final String accessTokenKey = "accessToken";
  static final String refreshTokenKey = "refreshToken";

  static Future<void> saveDataByJson(String email, String json) async {
    Map<String, dynamic> jsonMap = jsonDecode(json);

    print("Данные сохранены. email = [$email], "
        "accessToken = [${jsonMap["accessToken"]}], "
        "refreshToken = [${jsonMap["refreshToken"]}]");

    var storage = FlutterSecureStorage();

    storage.write(key: emailKey, value: email);
    storage.write(key: accessTokenKey, value: jsonMap["accessToken"]);
    storage.write(key: refreshTokenKey, value: jsonMap["refreshToken"]);
  }

  static Future<Map<String, String>> getData() async {
    return FlutterSecureStorage().readAll();
  }

  static Future<String> getFingerprint() async {
    var deviceInfo = DeviceInfoPlugin();
    if (Platform.isIOS) {
      var iosDeviceInfo = await deviceInfo.iosInfo;
      return iosDeviceInfo.identifierForVendor; // unique ID on iOS
    } else {
      var androidDeviceInfo = await deviceInfo.androidInfo;
      return androidDeviceInfo.androidId; // unique ID on Android
    }
  }

  static Future<String> getAccessToken() async {
    return (await Auth.getData())[Auth.accessTokenKey];
  }

  static Future<String> getEmail() async {
    return (await Auth.getData())[Auth.emailKey];
  }
  static Future<String> getRefreshToken() async {
    return (await Auth.getData())[Auth.refreshTokenKey];
  }
}
```

1.15. event_wrapper.dart

```
class EventWrapper<T> {
  final int statusCode;
  final T _data;
  final String message;

  EventWrapper(this.statusCode, this._data, this.message);

  bool isSuccess() {
    return _data != null;
  }

  T getData() {
    if (isSuccess()) {
      return _data;
    }

    throw Exception("Data is null");
  }
}
```

1.16. user_storage.dart

```
import 'package:hse_coffee/data/user.dart';

class UserStorage {
  User _user;

  User get user {
    if (_user == null) {
      throw Exception("Get null user");
    }
    return _user;
  }

  set user(User user) {
    if (user == null) {
      throw Exception("Set null user");
    }
    _user = user;
  }

  static final UserStorage instance = UserStorage._internal();

  factory UserStorage() {
    return instance;
  }

  UserStorage._internal();
}
```

1.17. main.dart

```
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:hse_coffee/ui/auth/auth_code.dart';
import 'package:hse_coffee/ui/auth/auth_contacts.dart';
import 'package:hse_coffee/ui/auth/auth_email.dart';
import 'package:hse_coffee/ui/auth/auth_faculty.dart';
import 'package:hse_coffee/ui/auth/auth_gender.dart';
import 'package:hse_coffee/ui/auth/auth_name.dart';
import 'package:hse_coffee/ui/auth/auth_photo.dart';
import 'package:hse_coffee/ui/home/home.dart';
import 'package:hse_coffee/ui/splash/splash.dart';

void main() {
  WidgetsFlutterBinding.ensureInitialized();
  SystemChrome.setPreferredOrientations([DeviceOrientation.portraitUp])
      .then((_) {
    runApp(new HseCoffeeApp());
  });
}

class HseCoffeeApp extends StatelessWidget {
  final routes = <String, WidgetBuilder>{
    AuthFacultyScreen.routeName: (BuildContext context) => AuthFacultyScreen(),
    AuthNameScreen.routeName: (BuildContext context) => AuthNameScreen(),
    AuthCodeScreen.routeName: (BuildContext context) => AuthCodeScreen(),
    AuthEmailScreen.routeName: (BuildContext context) => AuthEmailScreen(),
    AuthGenderScreen.routeName: (BuildContext context) => AuthGenderScreen(),
    AuthPhotoScreen.routeName: (BuildContext context) => AuthPhotoScreen(),
    AuthContactsScreen.routeName: (BuildContext context) =>
        AuthContactsScreen(),
    HomeScreen.routeName: (BuildContext context) =>
        HomeScreen(title: 'HSECoffee')
  };

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'HSEcoffee',
      home: SplashScreen(nextRoute: '/auth/email'),
      routes: routes,
    );
  }
}
```

1.18. router_auth.dart

```dart
import 'package:flutter/cupertino.dart';
import 'package:hse_coffee/data/faculty.dart';
import 'package:hse_coffee/data/gender.dart';
import 'package:hse_coffee/ui/auth/auth_contacts.dart';
import 'package:hse_coffee/ui/auth/auth_faculty.dart';
import 'package:hse_coffee/ui/auth/auth_gender.dart';
import 'package:hse_coffee/ui/auth/auth_name.dart';
import 'package:hse_coffee/ui/auth/auth_photo.dart';
import 'package:hse_coffee/ui/home/home.dart';

import 'data/user.dart';

class RouterHelper {
  static routeByUser(BuildContext context, User user) {
    // Если пусто имя
    if (user.firstName == null ||
        user.lastName == null ||
        user.firstName.isEmpty ||
        user.lastName.isEmpty) {
      Navigator.of(context).pushReplacementNamed(AuthNameScreen.routeName);
    }
    // Если не выбран пол
    else if (user.gender == null || user.gender == Gender.NONE) {
      Navigator.of(context).pushReplacementNamed(AuthGenderScreen.routeName);
    }
    // Если не выбран факультет
    else if (user.faculty == null ||
        user.faculty == Faculty.NONE ||
        user.course == null ||
        user.course == 0) {
      Navigator.of(context).pushReplacementNamed(AuthFacultyScreen.routeName);
    }
    // Если не указаны контактные данные
    else if (user.contacts.isEmpty) {
      Navigator.of(context).pushReplacementNamed(AuthContactsScreen.routeName);
    }
    // Если нет фотографии
    else if (!user.photoUri.contains(user.email)) {
      Navigator.of(context).pushReplacementNamed(AuthPhotoScreen.routeName);
    }
    // Иначе
    else {
      Navigator.of(context).pushReplacementNamed(HomeScreen.routeName);
    }
  }
}
```

1.19. auth_code.dart

```dart
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/router_auth.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'header.dart';

class AuthCodeScreen extends StatefulWidget {
  static const String routeName = "/auth/code";

  @override
  _AuthCodeScreen createState() => _AuthCodeScreen();
}

class _AuthCodeScreen extends State<AuthCodeScreen> {
  int count;
  bool _block;
  DateTime _blockTime;

  final globalKey = GlobalKey<ScaffoldState>();
  final textFieldKey = GlobalKey<FormState>();
  AnimationController rotationController;

  @override
  void initState() {
    count = 0;
    _block = false;
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    final dialogLoading = DialogLoading(context: this.context);

    String code;
    final ScreenAuthCodeArguments args =
        ModalRoute
            .of(context)
            .settings
            .arguments;

    void callSnackBar(String text) {
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
    }

    void errorSnackBar() {
      callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
    }

    void _onRetry() {
      if (count >= 1) {
        _blockTime = DateTime.now().add(Duration(minutes: 3));
        _block = true;
        count = 0;
      }
      if (_block) {
        if (DateTime.now().isAfter(_blockTime)) {
          _block = false;
        } else {
          Duration ostDate = _blockTime.difference(DateTime.now());
          globalKey.currentState.showSnackBar(SnackBar(
              content: Text(
                  "Не спешите! Подождите ${ostDate
                      .inMinutes} минут(ы) и ${ostDate.inSeconds %
                      60} секунд(ы).")));
        }
        return;
      }

      count++;
      dialogLoading.show();
      Api.sendCode(args.email)
          .then((value) =>
      {
        dialogLoading.stop(),
        if (value.statusCode == 200)
          {callSnackBar("Код был успешно выслан на почту повторно!")}
        else
          {
            count--,
            errorSnackBar(),
          }
      })
          .timeout(Duration(seconds: 15))
          .catchError((Object object) =>
      {count--, dialogLoading.stop(), errorSnackBar()});
    }

    Future<void> _onPressed() async {
      if (textFieldKey.currentState.validate()) {
        dialogLoading.show();
        Api.confirmCode(code, args.email)
            .then((value) =>
        {
          dialogLoading.stop(),
          if (value.isSuccess())
            {
              Api.getUser().then((value) =>
              {
                UserStorage.instance.user = value.getData(),
                RouterHelper.routeByUser(context, value.getData())
              })
            }
          else
            {
              callSnackBar(
                  "К сожалению, код неверный или произошла ошибка.")
            }
        })
            .timeout(Duration(seconds: 15))
```

```
            .catchError(
                (Object object) => {dialogLoading.stop(), errorSnackBar()});
      }
    }

    bool isDigit(String s, int idx) => (s.codeUnitAt(idx) ^ 0x30) <= 9;

    bool _isValidCodeForm(String code) {
      if (code == null || code.isEmpty) {
        return false;
      }

      if (code.length != 6) return false;

      for (int i = 0; i < code.length; i++) {
        if (!isDigit(code, i)) {
          return false;
        }
      }

      return true;
    }

    return Scaffold(
        key: globalKey,
        body: Builder(
            builder: (context) =>
                SingleChildScrollView(
                    reverse: true,
                    child:
                    Column(mainAxisSize: MainAxisSize.max, children: <Widget>[
                      Header(title: "Код"),
                      Padding(
                        padding: EdgeInsets.fromLTRB(30.0, 30.0, 30.0, 10.0),
                        child: Form(
                            key: textFieldKey,
                            child: TextFormField(
                              keyboardType: TextInputType.number,
                              cursorColor: Colors.blueAccent,
                              decoration: InputDecoration(
                                suffixIcon: IconButton(
                                  onPressed: _onRetry,
                                  icon: Icon(Icons.refresh),
                                ),
                                hintText: 'Введите код письма',
                                border: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(16.0),
                                  borderSide: BorderSide(
                                    width: 2,
                                    color: Colors.blueAccent,
                                  ),
                                ),
                                errorBorder: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(16.0),
                                  borderSide: BorderSide(
                                    width: 2,
                                        color: Colors.red,
                                      ),
                                    ),
                                    labelText: 'Код',
                                  ),
                                  validator: (input) =>
                                  _isValidCodeForm(input)
                                      ? null
                                      : "Код состоит только из 6-ти цифр.",
                                  onChanged: (String value) {
                                    code = value;
                                  },
                                  onEditingComplete: _onPressed,
                                )),
                          ),
                          Padding(
                            padding: EdgeInsets.all(15.0),
                            child: Text(
                                "Введите код, который пришёл на Ваш почтовый ящик ${args
                                    .email}",
                                textAlign: TextAlign.center,
                                style: TextStyle(
                                  fontSize: 12.0,
                                  color: Color.fromRGBO(81, 81, 81, 1),
                                )),
                          ),
                          ButtonContinue(onPressed: _onPressed)
                        ]))));
  }
}

class ScreenAuthCodeArguments {
  final String email;

  ScreenAuthCodeArguments(this.email);
}
```

1.20. auth_contacts.dart

```
import 'dart:collection';

import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/data/contact.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';

import '../../router_auth.dart';
import 'header.dart';

class AuthContactsScreen extends StatefulWidget {
  static const String routeName = "auth/contacts";

  @override
  _AuthContactsScreen createState() => _AuthContactsScreen();
}

class _AuthContactsScreen extends State<AuthContactsScreen> {
  final globalKey = GlobalKey<ScaffoldState>();

  final telegramFieldController = TextEditingController();

  @override
  void initState() {
    super.initState();
  }

  void callSnackBar(String text) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
  }

  @override
  void dispose() {
    telegramFieldController.dispose();

    super.dispose();
  }

  String tgClean(String tg) {
    return tg.trim().replaceAll("@", "");
  }

  @override
  Widget build(BuildContext context) {
    final dialogLoading = DialogLoading(context: this.context);

    void callSnackBar(String text) {
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
    }

    void errorSnackBar() {
      callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
    }

    final textFieldsKey = GlobalKey<FormState>();

    void _onButtonClick() {
      if (textFieldsKey.currentState.validate()) {
        UserStorage.instance.user.contacts = HashSet.of({
          Contact.createTelegram(telegramFieldController.text),
        });

        dialogLoading.show();
        Api.setUser(UserStorage.instance.user)
            .then((value) => {
                  if (value.isSuccess())
                    RouterHelper.routeByUser(context, UserStorage.instance.user)
                  else
                    callSnackBar("Произошла ошибка! Попробуйте позже.")
                })
            .timeout(Duration(seconds: 15))
            .catchError(
                (Object object) => {dialogLoading.stop(), errorSnackBar()});
        dialogLoading.stop();
      }
    }

    bool onTelegramValidate(String input) {
      return input != null && input.length > 2;
    }

    return Scaffold(
        key: globalKey,
        body: Builder(
            builder: (context) => SingleChildScrollView(
                reverse: true,
                child:
                    Column(mainAxisSize: MainAxisSize.max, children: <Widget>[
                  Header(title: "Мои контакты"),
                  Form(
                      key: textFieldsKey,
                      child: Column(
                        children: [
                          Padding(
                            padding:
                                EdgeInsets.fromLTRB(30.0, 15.0, 30.0, 10.0),
                            child: Text(
                              "Прежде чем продолжить, введите, пожалуйста, свой
реальный Telegram-логин",
                              textAlign: TextAlign.center,
                              style: TextStyle(
                                fontSize: 12.0,
                                color: Color.fromRGBO(81, 81, 81, 1),
                              ),
                            ),
                          ),
                          Padding(
                            padding: EdgeInsets.fromLTRB(30.0, 0.0, 30.0, 10.0),
                            child: TextField(
                              controller: telegramFieldController,
                              hintText: "@login",
                              labelText: "Логин Telegram",
                              validator: onTelegramValidate,
                              incorrectMessage:
                                  "Введите, пожалуйста, свой настоящий логин.",
                            ),
                          )
                        ],
                      )),
                  ButtonContinue(onPressed: _onButtonClick)
                ]))));
  }
}
```

```
class TextField extends StatelessWidget {
  static const int MinLengthName = 2;

  final String hintText;
  final String labelText;
  final TextEditingController controller;
  final Function(String value) validator;
  final String incorrectMessage;

  TextField(
      {Key key,
      this.hintText,
      this.labelText,
      this.controller,
      this.validator,
      this.incorrectMessage})
      : super(key: key);

  @override
  Widget build(BuildContext context) {
    return TextFormField(
      keyboardType: TextInputType.name,
      controller: this.controller,
      validator: (input) => validator(input) ? null : incorrectMessage,
      cursorColor: Colors.blueAccent,
      decoration: InputDecoration(
        hintText: hintText,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(16.0),
          borderSide: BorderSide(
            width: 2,
            color: Colors.blueAccent,
          ),
        ),
        errorBorder: OutlineInputBorder(
          borderRadius: BorderRadius.circular(16.0),
          borderSide: BorderSide(
            width: 2,
            color: Colors.red,
          ),
        ),
        labelText: labelText,
      ),
    );
  }
}
```

1.21. auth_email.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'auth_code.dart';
import 'header.dart';

class AuthEmailScreen extends StatefulWidget {
  static const String routeName = "/auth/email";

  @override
  _AuthEmailScreen createState() => _AuthEmailScreen();
}

class _AuthEmailScreen extends State<AuthEmailScreen> {
  final globalKey = GlobalKey<ScaffoldState>();
  final textFieldKey = GlobalKey<FormState>();

  bool _block;
  DateTime _blockTime;
  String email;
  int count;

  @override
  void initState() {
    _block = false;
    count = 0;
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    final dialogLoading = DialogLoading(context: this.context);

    void _nextPage() {
      Navigator.of(context).pushReplacementNamed(AuthCodeScreen.routeName,
          arguments: ScreenAuthCodeArguments(email));
    }

    void callSnackBar(String text) {
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
    }

    void errorSnackBar() {
      callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
    }

    void _onPressed() {
      if (_block) {
        if (DateTime.now().isAfter(_blockTime)) {
          _block = false;
        } else {
          Duration ostDate = _blockTime.difference(DateTime.now());
          globalKey.currentState.showSnackBar(SnackBar(
              content: Text(
                  "Не спешите! Подождите ${ostDate
                      .inMinutes} минут(ы) и ${ostDate.inSeconds %
                      60} секунд(ы).")));
          return;
        }
      }

      if (textFieldKey.currentState.validate()) {
        if (count >= 2) {
          _blockTime = DateTime.now().add(Duration(minutes: 5));
          _block = true;
          count = 0;
        }

        count++;
        dialogLoading.show();
        Api.sendCode(email.toLowerCase())
            .then((value) =>
        {
          dialogLoading.stop(),
          if (value.statusCode == 200)
            {_nextPage()}
          else
            {
              --count,
              globalKey.currentState.showSnackBar(
                  SnackBar(content: Text("Ошибка: ${value.message}")))
            }
        })
            .timeout(Duration(seconds: 15))
            .catchError((Object object) =>
        {
          print(object),
          --count,
          dialogLoading.stop(),
          errorSnackBar()
        });
      }
    }

    bool _isValidEmailForm(String email) {
      if (email == null || email
          .trim()
          .length < 4) {
        return false;
      }

      return (email.toLowerCase().endsWith("@edu.hse.ru") ||
          email.toLowerCase().endsWith("@hse.ru"));
    }

    return Scaffold(
        key: globalKey,
        body: Builder(
            builder: (context) =>
                SingleChildScrollView(
                    reverse: true,
                    child:
                    Column(mainAxisSize: MainAxisSize.max, children: <Widget>[
                      Header(title: "Моя почта"),
                      Padding(
                        padding: EdgeInsets.fromLTRB(30.0, 30.0, 30.0, 10.0),
                        child: Form(
                            key: textFieldKey,
                            child: TextFormField(
                              keyboardType: TextInputType.emailAddress,
                              cursorColor: Colors.blueAccent,
                              decoration: InputDecoration(
                                icon: Icon(Icons.email),
                                hintText: 'Введите свою корпоративную почту',
                                border: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(16.0),
                                  borderSide: BorderSide(
                                    width: 2,
                                    color: Colors.blueAccent,
                                  ),
                                ),
                                errorBorder: OutlineInputBorder(
                                  borderRadius: BorderRadius.circular(16.0),
                                  borderSide: BorderSide(
                                    width: 2,
                                    color: Colors.red,
                                  ),
                                ),
                                labelText: 'Моя почта',
                              ),
                              validator: (input) =>
                              _isValidEmailForm(input.trim())
                                  ? null
                                  : "Не забудьте @hse.ru или @edu.hse.ru",
                              onChanged: (String value) {
                                email = value.trim();
                              },
                              onEditingComplete: _onPressed,
                            )),
                      ),
                      Padding(
                        padding: EdgeInsets.all(15.0),
                        child: Text(
                            "Введите свой корпоративный почтовый ящик, на который
придёт письмо с кодом подтверждения.",
                            textAlign: TextAlign.center,
                            style: TextStyle(
                              fontSize: 12.0,
                              color: Color.fromRGBO(81, 81, 81, 1),
                            )),
                      ),
                      ButtonContinue(onPressed: _onPressed),
                    ]))));
  }
}
```

1.22. auth_faculty.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/data/degree.dart';
import 'package:hse_coffee/data/faculty.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'package:hse_coffee/ui/widgets/edu_fields.dart';

import '../../router_auth.dart';
import 'header.dart';

class AuthFacultyScreen extends StatefulWidget {
  static const String routeName = "/ui/auth/faculty";

  @override
  _AuthFacultyScreen createState() => _AuthFacultyScreen();
}

class _AuthFacultyScreen extends State<AuthFacultyScreen> {
  final globalKey = GlobalKey<ScaffoldState>();

  final _slider = SliderCourse(state: SliderState(currentSliderValue: 1.0));

  @override
  void initState() {
    super.initState();
  }

  void callSnackBar(String text) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
  }

  @override
  void dispose() {
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    EducationFields _educationFields = EducationFields();

    final dialogLoading = DialogLoading(context: this.context);
    void callSnackBar(String text) {
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
    }

    void errorSnackBar() {
      callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
    }

    void _onButtonClick() {
      Faculty currentFaculty = _educationFields.state.faculty;
      Degree currentDegree = _educationFields.state.degree;
      double course = _slider.state.currentSliderValue;

      if (currentFaculty == null || currentFaculty == Faculty.NONE) {
        callSnackBar("Пожалуйста, выберите свою образовательную программу!");
      } else if (currentDegree == null || currentDegree == Degree.NONE) {
        callSnackBar("Пожалуйста, выберите академическую степень!");
      } else {
        UserStorage.instance.user.faculty = currentFaculty;
        UserStorage.instance.user.degree = currentDegree;
        UserStorage.instance.user.course = course.toInt();

        dialogLoading.show();

        Api.setUser(UserStorage.instance.user)
            .then((value) => {
                  if (value.isSuccess())
                    RouterHelper.routeByUser(context, UserStorage.instance.user)
                  else
                    callSnackBar("Произошла ошибка! Попробуйте позже.")
                })
            .timeout(Duration(seconds: 15))
            .catchError(
                (Object object) => {dialogLoading.stop(), errorSnackBar()});

        dialogLoading.stop();
        return;
      }
    }

    return Scaffold(
        key: globalKey,
        body: Builder(
            builder: (context) => SingleChildScrollView(
                reverse: true,
                child:
                    Column(mainAxisSize: MainAxisSize.max, children: <Widget>[
                  Header(title: "Образовательная\nпрограмма"),
                  Padding(
                    padding: const EdgeInsets.all(8.0),
                    child: _educationFields,
                  ),
                  Padding(
                    padding: EdgeInsets.fromLTRB(45.0, 0.0, 45.0, 10.0),
                    child: Row(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: <Widget>[Text("Курс: "), _slider],
                    ),
                  ),
                  ButtonContinue(onPressed: _onButtonClick)
                ]))));
  }
}

class DropDown<T extends State<StatefulWidget>> extends StatefulWidget {
  final T state;
  DropDown({@required this.state});

  @override
  State<StatefulWidget> createState() {
    return state;
  }
}

class SliderCourse extends StatefulWidget {
  final SliderState state;

  SliderCourse({this.state, Key key}) : super(key: key);

  @override
  SliderState createState() => state;
}

class SliderState extends State<SliderCourse> {
  bool isInit = false;
  double currentSliderValue = 1;

  SliderState({this.currentSliderValue});

  @override
  Widget build(BuildContext context) {
    return Slider(
      value: currentSliderValue,
      min: 1,
      max: 6,
      divisions: 5,
      label: currentSliderValue.round().toString() + " курс",
      onChanged: (double value) {
        isInit = true;
        setState(() {
          currentSliderValue = value;
        });
      },
    );
  }
}
```

1.23. auth_gender.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/data/gender.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';

import '../../router_auth.dart';
import 'header.dart';
```

```
class AuthGenderScreen extends StatefulWidget {
  static const String routeName = "/auth/gender";

  @override
  _AuthGenderScreen createState() => _AuthGenderScreen();
}

class _AuthGenderScreen extends State<AuthGenderScreen> {
  @override
  void initState() {
    super.initState();
  }

  void callSnackBar(String text) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
  }

  @override
  void dispose() {
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
    final dialogLoading = DialogLoading(context: this.context);

    final _toggleButton = ToggleButton();

    bool _isValidInput() {
      if (_toggleButton.buttonState.isSelected.contains(true)) {
        return true;
      }

      callSnackBar("Выберите, пожалуйста, свой пол!");
      return false;
    }

    void errorSnackBar() {
      callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
    }

    void _onButtonClick() {
      if (_isValidInput()) {
        UserStorage.instance.user.gender =
            _toggleButton.buttonState.isSelected[0]
                ? Gender.MALE
                : Gender.FEMALE;

        dialogLoading.show();

        Api.setUser(UserStorage.instance.user)
            .then((value) => {
                  if (value.isSuccess())
                    RouterHelper.routeByUser(context, UserStorage.instance.user)
                  else
```

```
                    callSnackBar("Произошла ошибка! Попробуйте позже.")
                })
            .timeout(Duration(seconds: 15))
            .catchError(
                (Object object) => {dialogLoading.stop(), errorSnackBar()});

        dialogLoading.stop();
      }
    }

    return Scaffold(
        body: Builder(
            builder: (context) => SingleChildScrollView(
                reverse: true,
                child:
                    Column(mainAxisSize: MainAxisSize.max, children: <Widget>[
                  Header(title: "Мой пол"),
                  _toggleButton,
                  Padding(
                      padding: EdgeInsets.symmetric(
                          horizontal: 80.0, vertical: 10.0),
                      child: Text(
                        "Прежде чем продолжить, выберите, пожалуйста, свой пол.",
                        textAlign: TextAlign.center,
                        style: TextStyle(
                          fontSize: 12.0,
                          color: Color.fromRGBO(81, 81, 81, 1),
                        ),
                      )),
                  ButtonContinue(onPressed: _onButtonClick),
                ]))));
  }
}

class ToggleButton extends StatefulWidget {
  final _ToggleButtonState buttonState = _ToggleButtonState();
  final Function onPressed;

  ToggleButton({Key key, this.onPressed}) : super(key: key);

  @override
  State<StatefulWidget> createState() {
    return buttonState;
  }
}

class _ToggleButtonState extends State {
  List<bool> isSelected = <bool>[false, false];
  static const String FirstButtonText = "Мужской";
  static const String SecondButtonText = "Женский";

  _ToggleButtonState();

  void onClicked(int index) {
    if (isSelected[index]) return;
    setState(() {
      bool state = isSelected[index];
      isSelected[index] = !state;
      for (int i = 0; i < isSelected.length; i++) {
        if (i != index) {
          isSelected[i] = state;
        }
      }
    });
  }

  @override
  Widget build(BuildContext context) {
    return Column(children: <Widget>[
      Padding(
          padding: EdgeInsets.fromLTRB(30.0, 50.0, 30.0, 10.0),
          child: TButton(
            FirstButtonText,
            () => onClicked(0),
            isClicked: isSelected[0],
          )),
      Padding(
          padding: EdgeInsets.fromLTRB(30.0, 0.0, 30.0, 30.0),
          child: TButton(
            SecondButtonText,
            () => onClicked(1),
            isClicked: isSelected[1],
          )),
    ]);
  }
}

class TButton extends StatelessWidget {
  final String _text;
  final Function _onPressed;
  final bool isClicked;

  TButton(this._text, this._onPressed, {this.isClicked});

  @override
  Widget build(BuildContext context) {
    return new SizedBox(
      width: 160.0,
      height: 50.0,
      child: isClicked
          ? _ClickedButton(_text, _onPressed)
          : _NotClickedButton(_text, _onPressed),
    );
  }
}

class _ClickedButton extends StatelessWidget {
  final String _text;
  final Function _onPressed;

  _ClickedButton(this._text, this._onPressed);
  @override
  Widget build(BuildContext context) {
    return OutlinedButton(
      child: Text(
        _text,
        style: TextStyle(fontSize: 16.0),
      ),
      style: OutlinedButton.styleFrom(
        primary: Color.fromRGBO(20, 92, 212, 1),
        side: BorderSide(color: Color.fromRGBO(20, 102, 212, 1), width: 2),
        shape: const RoundedRectangleBorder(
            borderRadius: BorderRadius.all(Radius.circular(32))),
      ),
      onPressed: () {
        _onPressed();
      },
    );
  }
}

class _NotClickedButton extends StatelessWidget {
  final String _text;
  final Function _onPressed;

  _NotClickedButton(this._text, this._onPressed);

  @override
  Widget build(BuildContext context) {
    return OutlinedButton(
      child: Text(
        _text,
        style: TextStyle(fontSize: 16.0),
      ),
      style: OutlinedButton.styleFrom(
        primary: Color.fromRGBO(81, 81, 81, 1),
        backgroundColor: Color.fromRGBO(245, 245, 245, 1),
        shape: const RoundedRectangleBorder(
            borderRadius: BorderRadius.all(Radius.circular(32))),
      ),
      onPressed: () {
        _onPressed();
      },
    );
  }
}
```

1.24. auth_name.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'header.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import '../../router_auth.dart';

class AuthNameScreen extends StatefulWidget {
  static const String routeName = "/auth/name";

  @override
  _AuthNameScreen createState() => _AuthNameScreen();
}

class _AuthNameScreen extends State<AuthNameScreen> {
  final globalKey = GlobalKey<ScaffoldState>();

  final firstNameFieldController = TextEditingController();
  final secondNameFieldController = TextEditingController();

  @override
  void initState() {
    super.initState();
  }

  void callSnackBar(String text) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
  }

  @override
  void dispose() {
    firstNameFieldController.dispose();
    secondNameFieldController.dispose();

    super.dispose();
  }

  void errorSnackBar() {
    callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
  }

  @override
  Widget build(BuildContext context) {
    final dialogLoading = DialogLoading(context: this.context);

    final textFieldsKey = GlobalKey<FormState>();

    void _onButtonClick() {
      if (textFieldsKey.currentState.validate()) {
        UserStorage.instance.user.firstName = firstNameFieldController.text;
        UserStorage.instance.user.lastName = secondNameFieldController.text;

        dialogLoading.show();
        Api.setUser(UserStorage.instance.user)
            .then((value) => {
                  if (value.isSuccess())
                    RouterHelper.routeByUser(context, UserStorage.instance.user)
                  else
```

```
                    callSnackBar("Произошла ошибка! Попробуйте позже.")
                })
            .timeout(Duration(seconds: 15))
            .catchError(
                (Object object) => {dialogLoading.stop(), errorSnackBar()});
        dialogLoading.stop();
      }
    }

    return Scaffold(
        key: globalKey,
        body: Builder(
            builder: (context) => SingleChildScrollView(
                reverse: true,
                child:
                    Column(mainAxisSize: MainAxisSize.max, children: <Widget>[
                  Header(title: "Меня зовут"),
                  Form(
                      key: textFieldsKey,
                      child: Column(
                        children: [
                          Padding(
                            padding:
                                EdgeInsets.fromLTRB(30.0, 30.0, 30.0, 10.0),
                            child: TextField(
                              controller: firstNameFieldController,
                              hintText: "Введите своё имя",
                              labelText: "Имя",
                            ),
                          ),
                          Padding(
                            padding: EdgeInsets.fromLTRB(30.0, 0.0, 30.0, 10.0),
                            child: TextField(
                                controller: secondNameFieldController,
                                hintText: "Введите свою фамилию",
                                labelText: "Фамилия"),
                          )
                        ],
                      )),
                  ButtonContinue(onPressed: _onButtonClick)
                ]))));
  }
}

class TextField extends StatelessWidget {
  static const int MinLengthName = 2;

  final String hintText;
  final String labelText;
  final TextEditingController controller;

  TextField({Key key, this.hintText, this.labelText, this.controller})
      : super(key: key);

  bool _isValidName(String text) {
    return text != null && text.length > MinLengthName;
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.05-01 12 01-2				
Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
  }

  @override
  Widget build(BuildContext context) {
    return TextFormField(
      keyboardType: TextInputType.name,
      controller: this.controller,
      validator: (input) =>
          _isValidName(input) ? null : "Пожалуйста, заполните корректно поле!",
      cursorColor: Colors.blueAccent,
      decoration: InputDecoration(
        hintText: hintText,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(16.0),
          borderSide: BorderSide(
            width: 2,
            color: Colors.blueAccent,
          ),
        ),
        errorBorder: OutlineInputBorder(
          borderRadius: BorderRadius.circular(16.0),
          borderSide: BorderSide(
            width: 2,
            color: Colors.red,
          ),
        ),
        labelText: labelText,
      ),
    );
  }
}
```

1.25. auth_photo.dart

```
import 'dart:io';

import 'package:cached_network_image/cached_network_image.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:gradient_widgets/gradient_widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'package:image_cropper/image_cropper.dart';
import 'package:image_picker/image_picker.dart';

import '../../router_auth.dart';
import 'header.dart';

class AuthPhotoScreen extends StatefulWidget {
  static const String routeName = "/auth/photo";
  @override
  _AuthPhotoScreen createState() => _AuthPhotoScreen();
}

class _AuthPhotoScreen extends State<AuthPhotoScreen> {
  File _image;
  final picker = ImagePicker();

  @override
  void initState() {
    super.initState();
  }

  @override
  void dispose() {
    super.dispose();
  }

  final globalKey = GlobalKey<ScaffoldState>();

  void callSnackBar(String text) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
  }

  Future _sendImage() {
    return Api.setPhoto(UserStorage.instance.user, _image)
        .then((value) => {
              if (value.isSuccess())
                {
                  Api.getUser()
                      .then((value) => {
                            UserStorage.instance.user = value.getData(),
                            setState(() {}),
                            RouterHelper.routeByUser(context, value.getData())
                          })
                      .timeout(Duration(seconds: 15))
                }
              else
                {callSnackBar("К сожалению, не удалось загрузить фотографию.")}
            })
        .timeout(Duration(seconds: 25));
  }

  Future getImage(ImageSource source) async {
    final pickedFile = await picker.getImage(source: source);
    if (pickedFile != null) {
      _image = await ImageCropper.cropImage(
          sourcePath: pickedFile.path,
          aspectRatioPresets: [
            CropAspectRatioPreset.square,
            CropAspectRatioPreset.ratio4x3,
            CropAspectRatioPreset.original
          ],
          maxWidth: 800);

      setState(() {
```

```
      //
    });
    var dialogLoading = DialogLoading(context: context);
    dialogLoading.show();
    _sendImage().then((value) => dialogLoading.stop());
  }
}

void _showPickOptionsDialog(BuildContext context) {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      content: Column(
        mainAxisSize: MainAxisSize.min,
        children: <Widget>[
          ListTile(
            title: Text("Выбрать из галереи"),
            onTap: () {
              getImage(ImageSource.gallery);
            },
          ),
          ListTile(
            title: Text("Сделать фотографию"),
            onTap: () {
              getImage(ImageSource.camera);
            },
          )
        ],
      ),
    ),
  );
}

@override
Widget build(BuildContext context) {
  void _onButtonClick() {
    var user = UserStorage.instance.user;
    if (user.photoUri.contains(user.email)) {
      RouterHelper.routeByUser(context, user);
    } else {
      callSnackBar("Загрузите свою фотографию, это важно!");
    }
  }

  return Scaffold(
      key: globalKey,
      body: Builder(
          builder: (context) =>
              Column(mainAxisSize: MainAxisSize.max, children: <Widget>[
                Header(
                    title: "Фотография",
                    image: CircleAvatar(
                      radius: 80,
                      child: _image == null
                          ? ClipRRect(
                              borderRadius: BorderRadius.circular(80),
                              child: CachedNetworkImage(
                                imageUrl: Api.getImageUrlByUser(
                                    UserStorage.instance.user),
                                placeholder: (context, url) =>
                                    new CircularProgressIndicator(),
                                errorWidget: (context, url, dynamic) =>
                                    new Icon(Icons.error),
                              ),
                            )
                          : null,
                      backgroundImage:
                          _image != null ? FileImage(_image) : null,
                    )),
                Padding(
                    padding: EdgeInsets.fromLTRB(15, 15, 15, 5),
                    child: Text(
                      "Загрузите, пожалуйста, свою фотографию. Вашему будущему
собеседнику будет приятно!",
                      style: TextStyle(color: Colors.grey),
                      textAlign: TextAlign.center,
                    )),
                GradientButton(
                  child: Text("Загрузить фотографию",
                      style: TextStyle(fontSize: 16.0)),
                  callback: () {
                    _showPickOptionsDialog(context);
                  },
                  increaseWidthBy: 150.0,
                  increaseHeightBy: 10,
                  gradient: LinearGradient(
                      begin: Alignment.topLeft,
                      end: Alignment.bottomRight,
                      colors: [
                        Color.fromRGBO(0, 82, 212, 1),
                        Color.fromRGBO(49, 94, 252, 1),
                        Color.fromRGBO(111, 177, 252, 1)
                      ]),
                  shadowColor:
                      Gradients.backToFuture.colors.last.withOpacity(0.25),
                ),
                Expanded(
                    child: Center(
                        child: ButtonContinue(onPressed: _onButtonClick)))
              ])));
  }
}
```

1.26. header.dart

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

class Header extends StatelessWidget {
  Header({Key key, this.title, this.image}) : super(key: key);
  final String title;
  final CircleAvatar image;

  @override
  Widget build(BuildContext context) {
    return Stack(
      alignment: Alignment.center,
      children: <Widget>[
        Image.asset("images/header/cloud.png"),
        Transform.rotate(
            angle: 0,
            child: Container(
                decoration: new BoxDecoration(
                    color: Color.fromRGBO(30, 51, 123, 0.5),
                    borderRadius: BorderRadius.circular(50)),
                child: SizedBox(height: 175, width: 175))),
        Transform.rotate(
            angle: 90,
            child: Container(
                decoration: new BoxDecoration(
                    color: Color.fromRGBO(30, 51, 123, 0.5),
                    borderRadius: BorderRadius.circular(50)),
                child: SizedBox(height: 125, width: 125))),
        Transform.rotate(
            angle: 90,
            child: Container(
                decoration: new BoxDecoration(
                    color: Color.fromRGBO(30, 51, 123, 0.5),
                    borderRadius: BorderRadius.circular(50)),
                child: SizedBox(height: 75, width: 75))),
        Padding(
          padding: EdgeInsets.symmetric(horizontal: 80),
          child: image == null
              ? Text(title,
                  textAlign: TextAlign.center,
                  style: TextStyle(
                      fontFamily: 'Nunito',
                      color: Colors.white,
                      fontSize: 24,
                      fontWeight: FontWeight.bold))
              : image,
        )
      ],
    );
  }
}
```

1.27. home.dart

```
import 'dart:async';
import 'dart:core';

import 'package:flutter/material.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/data/meet.dart';
import 'package:hse_coffee/data/meet_status.dart';
import 'package:hse_coffee/ui/home/home_meets.dart';
import 'package:hse_coffee/ui/home/home_person_info.dart';

import 'home_cabinet.dart';
import 'home_find_start.dart';
import 'home_loading.dart';

// Такое же виджет, как и SplashScreen, только передаём ему ещё и заголовок
class HomeScreen extends StatefulWidget {
  static const String routeName = "/home";

  HomeScreen({Key key, this.title}) : super(key: key);
  final String title;

  @override
  _HomeScreenState createState() => _HomeScreenState();
}

// Формирование состояния виджета
class _HomeScreenState extends State<HomeScreen> {
  int _selectedIndex = 0;

  _HomeScreenState() {
    Timer(Duration(seconds: 3), () {
      _updByMeet(null);
    });

    wait();
  }

  void wait() {
    Timer.periodic(Duration(seconds: 5), (timer) {
      _updByMeet(timer);
    });
  }

  void callSnackBar(String text) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
  }

  void callErrorSnackBar() {
    callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
  }

  void _updByMeet(Timer timer) {
    Api.getMeet()
        .then((value) => {
              if (value.isSuccess())
                {
                  _navigateByMeet(value.getData()),
                  if (value.getData().meetStatus == MeetStatus.ACTIVE)
                    {
                      timer.cancel(),
                      Timer(
                          Duration(
                              milliseconds:
                                  value.getData().expiresDate.microsecond -
                                      DateTime.now().millisecond), () {
                        wait();
                      })
                    }
                }
            })
        .timeout(Duration(seconds: 15))
        .catchError((obj) => print(obj.toString()));
  }

  final List<Widget> _widgetOptions = <Widget>[
    HomeLoadingScreen(),
    HomeMeetsScreen(),
    HomeCabinetScreen()
  ];

  Widget getScreenByMeet(Meet meet, MeetStatus meetStatus) {
    if (meetStatus == MeetStatus.NONE || meetStatus == MeetStatus.FINISHED) {
      return HomeFindScreen(_navigateByMeetStatus);
    } else if (meetStatus == MeetStatus.SEARCH) {
      return HomeLoadingScreen(
          withCancel: true, navigatorFunc: _navigateByMeetStatus);
    } else if (meetStatus == MeetStatus.ACTIVE && meet != null) {
      var user = UserStorage().user.email == meet.user1.email
          ? meet.user2
          : meet.user1;
      return HomePersonScreen(user, withBack: false);
    }

    return HomeLoadingScreen();
  }

  void _navigateByMeet(Meet meet) {
    _widgetOptions[0] = getScreenByMeet(meet, meet.meetStatus);

    if (_selectedIndex == 0) {
      _onItemTapped(0);
    }
  }

  void _navigateByMeetStatus(MeetStatus meetStatus) {
    _widgetOptions[0] = getScreenByMeet(null, meetStatus);

    if (_selectedIndex == 0) {
      _onItemTapped(0);
    }
  }

  void _onItemTapped(int index) {
    setState(() {
      _selectedIndex = index;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: _widgetOptions.elementAt(_selectedIndex),
      ),
      bottomNavigationBar: BottomNavigationBar(
        type: BottomNavigationBarType.fixed,
        items: const <BottomNavigationBarItem>[
          BottomNavigationBarItem(
            icon: Icon(Icons.find_replace),
            label: 'Поиск',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.history),
            label: 'История',
          ),
          BottomNavigationBarItem(
            icon: Icon(Icons.person),
            label: 'Кабинет',
          ),
        ],
        currentIndex: _selectedIndex,
        selectedItemColor: Color.fromRGBO(67, 100, 248, 0.98),
        onTap: _onItemTapped,
      ),
    );
  }
}
```

1.28. home_cabinet.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/rendering.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';

import 'home_cabinet_header.dart';
import 'home_cabinet_settings.dart';

class HomeCabinetScreen extends StatefulWidget {
  static const String routeName = "/home/lk";

  @override
  _HomeCabinetScreen createState() => _HomeCabinetScreen();
}

class _HomeCabinetScreen extends State<HomeCabinetScreen> {
  Header _header = Header(UserStorage.instance.user);
  Settings _settings = Settings(UserStorage.instance.user);
  @override
  void initState() {
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        floatingActionButton: Padding(
            padding: EdgeInsets.all(10),
            child: FloatingActionButton(
              backgroundColor: Colors.blueAccent,
              child: Icon(Icons.assignment_turned_in_outlined),
              onPressed: () {
                if (_settings.floatingButtonPress != null)
                  _settings.floatingButtonPress();
              },
            )),
        floatingActionButtonLocation: FloatingActionButtonLocation.endDocked,
        body: Builder(
            builder: (context) => SingleChildScrollView(
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.stretch,
                    mainAxisSize: MainAxisSize.min,
                    children: <Widget>[_header, _settings],
                  ),
                )));
  }
}
```

1.29. home_cabinet_header.dart

```
import 'dart:io';

import 'package:cached_network_image/cached_network_image.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/data/user.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'package:image_cropper/image_cropper.dart';
import 'package:image_picker/image_picker.dart';

class HeaderState extends State<Header> {
  User currentUser;

  HeaderState({this.currentUser});

  void callSnackBar(String text) {
    ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
  }

  void callErrorSnackBar() {
```

```
  callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
}

Future _sendImage() {
  return Api.setPhoto(UserStorage.instance.user, _image).then((value) => {
        if (value.isSuccess())
          {
            Api.getUser().then((value) => {
                  if (value.isSuccess())
                    {
                      UserStorage.instance.user = value.getData(),
                      currentUser = value.getData(),
                      Api.updateImage(),
                      setState(() {}),
                    }
                })
          }
        else
          {callSnackBar("К сожалению, не удалось загрузить фотографию.")}
      });
}

File _image;
final picker = ImagePicker();

Future getImage(ImageSource source) async {
  final pickedFile = await picker.getImage(source: source);
  if (pickedFile != null) {
    _image = await ImageCropper.cropImage(
        sourcePath: pickedFile.path,
        aspectRatioPresets: [
          CropAspectRatioPreset.square,
          CropAspectRatioPreset.ratio4x3,
          CropAspectRatioPreset.original
        ],
        maxWidth: 800);

    setState(() {
      //
    });
    var dialogLoading = DialogLoading(context: context);
    dialogLoading.show();
    _sendImage().then((value) => dialogLoading.stop());
  }
}

void showPickOptionsDialog(BuildContext context) {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      content: Column(
        mainAxisSize: MainAxisSize.min,
        children: <Widget>[
          ListTile(
            title: Text("Выбрать из галереи"),
            onTap: () {
                getImage(ImageSource.gallery);
              },
            ),
            ListTile(
              title: Text("Сделать фотографию"),
              onTap: () {
                getImage(ImageSource.camera);
              },
            )
          ],
        ),
      ),
    );
  }

  void onTap() {
    showPickOptionsDialog(context);
    setState(() {});
  }

  @override
  Widget build(BuildContext context) {
    return Stack(
      alignment: Alignment.center,
      children: <Widget>[
        Image.asset("images/header/cloud.png"),
        Padding(
          padding: EdgeInsets.symmetric(horizontal: 80),
          child: new GestureDetector(
              onTap: () => onTap(),
              child: Stack(
                children: <Widget>[
                  new Container(
                      margin: EdgeInsets.only(top: 30, bottom: 15),
                      width: 220.0,
                      height: 220.0,
                      decoration: new BoxDecoration(
                          shape: BoxShape.circle,
                          image: new DecorationImage(
                              fit: BoxFit.fill,
                              image: new CachedNetworkImageProvider(
                                Api.getImageUrlByUser(currentUser),
                              )))),
                  new Container(
                      margin: EdgeInsets.only(top: 30, bottom: 15),
                      width: 220.0,
                      height: 220.0,
                      child: Icon(Icons.add, color: Colors.white54, size: 64))
                ],
              )),
        )
      ],
    );
  }
}
```

```
class Header extends StatefulWidget {
  final User _currentUser;

  Header(this._currentUser, {Key key}) : super(key: key);

  @override
  State<StatefulWidget> createState() {
    return HeaderState(currentUser: _currentUser);
  }
}
```

1.30. home_cabinet_settings.dart

```
import 'dart:async';

import 'package:flutter/material.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/data/contact.dart';
import 'package:hse_coffee/data/gender.dart';
import 'package:hse_coffee/data/user.dart';
import 'package:hse_coffee/ui/auth/auth_faculty.dart';
import 'package:hse_coffee/ui/widgets/capsule_widget.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'package:hse_coffee/ui/widgets/edu_fields.dart';
import 'package:hse_coffee/ui/widgets/text_field_wrapper.dart';
import 'package:hse_coffee/ui/widgets/toggle_button_gender.dart';

class Settings extends StatelessWidget {
  final nameKey = GlobalKey<FormState>();

  Function floatingButtonPress;
  TextEditingController _firstNameFieldController;
  TextEditingController _aboutFieldController;
  TextEditingController _lastNameFieldController;
  TextEditingController _vkFieldController;
  TextEditingController _tgFieldController;
  TextEditingController _instFieldController;

  EducationFields _educationFields = EducationFields();

  SliderCourse _slider;
  ToggleButtonGender _toggleButton;

  final User _currentUser;

  Settings(this._currentUser) {
    // Init elements by user instance
    _slider = SliderCourse(
        state: SliderState(currentSliderValue: _currentUser.course.toDouble()));

    _educationFields.state.degree = _currentUser.degree;
    _educationFields.state.faculty = _currentUser.faculty;
    _tgFieldController =
        TextEditingController(text: _currentUser.getTelegram());
    _vkFieldController = TextEditingController(text: _currentUser.getVk());
    _instFieldController = TextEditingController(text: _currentUser.getInst());

    _lastNameFieldController =
        TextEditingController(text: _currentUser.lastName);

    _firstNameFieldController =
        TextEditingController(text: _currentUser.firstName);

    _aboutFieldController = TextEditingController(text: _currentUser.aboutMe);

    var male = _currentUser.gender == Gender.MALE;
    var female = _currentUser.gender == Gender.FEMALE;

    _toggleButton = ToggleButtonGender(
        buttonState: ToggleButtonGenderState(List.of({male, female})));
  }

  @override
  Widget build(BuildContext context) {
    void callSnackBar(String text) {
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text(text)));
    }

    void callErrorSnackBar() {
      callSnackBar('Ошибка! Попробуйте повторить запрос позже.');
    }

    bool isContinue = true;

    void _onPressed() {
      if (!isContinue) {
        callSnackBar("Не спешите!");
        return;
      }

      isContinue = false;
      Timer(Duration(seconds: 3), () {
        isContinue = true;
      });

      if (nameKey.currentState.validate()) {
        User newUser = getUserByUi();

        DialogLoading dialogLoading = DialogLoading(context: context);

        dialogLoading.show();
        Api.setUser(newUser)
            .then((value) => {
                  dialogLoading.stop(),
                  if (value.isSuccess())
                    UserStorage.instance.user = newUser
                  else
                    callErrorSnackBar()
            })
          .catchError(
              (object) => {callErrorSnackBar(), dialogLoading.stop()});
      }
    }

    floatingButtonPress = _onPressed;

    return Column(children: <Widget>[
      Form(
          key: nameKey,
          child: Column(
            children: [
              CapsuleWidget(
                label: 'О себе'.toUpperCase(),
                ribbonHeight: 12,
              ),
              Padding(
                padding: EdgeInsets.fromLTRB(30.0, 0.0, 30.0, 0.0),
                child: TextFieldWrapper(
                  textInputType: TextInputType.multiline,
                  controller: _aboutFieldController,
                  hintText: "Укажите информацию о себе",
                  labelText: "Мои хобби, интересы, увлечения",
                  maxLines: 4,
                  minLengthName: -1,
                  maxLengthName: 100,
                ),
              ),
              const Divider(indent: 20),
              CapsuleWidget(
                label: 'Мои данные'.toUpperCase(),
                ribbonHeight: 12,
              ),
              Padding(
                padding: EdgeInsets.fromLTRB(30.0, 0.0, 30.0, 0.0),
                child: TextFieldWrapper(
                  controller: _firstNameFieldController,
                  hintText: "Введите своё имя",
                  labelText: "Имя",
                ),
              ),
              Padding(
                padding: EdgeInsets.fromLTRB(30.0, 5.0, 30.0, 0.0),
                child: TextFieldWrapper(
                    controller: _lastNameFieldController,
                    hintText: "Введите свою фамилию",
                    labelText: "Фамилия"),
              ),
              Padding(padding: EdgeInsets.all(7.5), child: _toggleButton),
              const Divider(indent: 20),
              CapsuleWidget(
                label: 'Мои контакты'.toUpperCase(),
                ribbonHeight: 12,
              ),
              Column(
```

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.05-01 12 01-2				
Инв. № подл.	Подп. и	Взам. инв. №	Инв. № дубл.	Подп. и дата

```
                children: [
                  Padding(
                    padding: EdgeInsets.fromLTRB(30.0, 0.0, 30.0, 0.0),
                    child: TextFieldWrapper(
                        controller: _tgFieldController,
                        hintText: "Введите свой Telegram-логин",
                        iconPath: "images/icons/tg.png",
                        labelText: "Telegram"),
                  ),
                  Padding(
                    padding: EdgeInsets.fromLTRB(30.0, 7.0, 30.0, 0.0),
                    child: TextFieldWrapper(
                        controller: _vkFieldController,
                        ignoreValidate: true,
                        hintText: "Введите свой VK-логин",
                        labelText: "VK",
                        iconPath: "images/icons/vk.png"),
                  ),
                  Padding(
                    padding: EdgeInsets.fromLTRB(30.0, 7.0, 30.0, 0.0),
                    child: TextFieldWrapper(
                        controller: _instFieldController,
                        ignoreValidate: true,
                        hintText: "Введите свой Instagram",
                        iconPath: "images/icons/inst.png",
                        labelText: "Instagram"),
                  )
                ],
              ),
            ],
          )),
      const Divider(indent: 20),
      CapsuleWidget(
        label: 'Обучение'.toUpperCase(),
        ribbonHeight: 12,
      ),
      _educationFields,
      Padding(
        padding: EdgeInsets.fromLTRB(45.0, 0.0, 45.0, 10.0),
        child: Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[Text("Курс: "), _slider],
        ),
      ),
    ]);
  }

  User getUserByUi() {
    User newUser = UserStorage.instance.user;

    newUser.firstName = _firstNameFieldController.text;
    newUser.lastName = _lastNameFieldController.text;
    newUser.aboutMe = _aboutFieldController.text;

    newUser.contacts = Set.of({
      Contact.createVk(_vkFieldController.text),
      Contact.createInstagram(_instFieldController.text),
      Contact.createTelegram(_tgFieldController.text)
    });

    List<bool> isSelected = _toggleButton.buttonState.isSelected;
    if (isSelected.length == 2) {
      isSelected[0]
          ? newUser.gender = Gender.MALE
          : newUser.gender = Gender.FEMALE;
    } else {
      newUser.gender = Gender.NONE;
    }

    newUser.degree = _educationFields.state.degree;
    newUser.faculty = _educationFields.state.faculty;
    newUser.course = _slider.state.currentSliderValue.toInt();
    return newUser;
  }
}
```

1.31. home_find_start.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/data/degree.dart';
import 'package:hse_coffee/data/faculty.dart';
import 'package:hse_coffee/data/meet_status.dart';
import 'package:hse_coffee/data/search_params.dart';
import 'package:hse_coffee/ui/auth/header.dart';
import 'package:hse_coffee/ui/widgets/auth_faculty_items.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'package:hse_coffee/ui/widgets/toggle_button_gender.dart';
import 'package:smart_select/smart_select.dart';

class HomeFindScreen extends StatefulWidget {
  final Function(MeetStatus meetStatus) navigatorFunc;

  HomeFindScreen(this.navigatorFunc);

  static const String routeName = "/home/find";

  @override
  _HomeFindScreen createState() => _HomeFindScreen(navigatorFunc);
}

class _HomeFindScreen extends State<HomeFindScreen> {
  final Function(MeetStatus meetStatus) navigatorFunc;

  _HomeFindScreen(this.navigatorFunc);

  @override
```

```
void initState() {
  super.initState();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
      body: Builder(
          builder: (context) => Column(
                children: [
                  Stack(
                    children: [
                      Header(title: "Поиск собеседника"),
                    ],
                  ),
                  Expanded(
                      child: ButtonContinue(
                          textButton: "Найти встречу",
                          onPressed: () => _settingModalBottomSheet(context)))
                ],
              )));
}

void _settingModalBottomSheet(context) {
  final featuresModalWidget = FeaturesModalConfirm();
  final rangeWidget = RangeWidget(_RangeWidgetState(new RangeValues(1, 6)),
      min: 1, max: 6, divisions: 5);
  final _toggleButtonWidget = ToggleButtonGender(
      buttonState: ToggleButtonGenderState([true, true]),
      onlySingleChose: false);

  SearchParams getSearchParams() {
    var genders = _toggleButtonWidget.buttonState.getGenders();
    var faculties = featuresModalWidget.state.faculties;
    var degrees = featuresModalWidget.state.degrees;

    var minCourse =
        rangeWidget.rangeWidgetState.currentRangeValues.start.toInt();
    var maxCourse =
        rangeWidget.rangeWidgetState.currentRangeValues.end.toInt();

    return SearchParams(genders.toSet(), faculties.toSet(), degrees.toSet(),
        minCourse, maxCourse);
  }

  void callSnackBar(String text, BuildContext context) {
    ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(behavior: SnackBarBehavior.floating, content: Text(text)));
  }

  void callErrorSnackBar(BuildContext context) {
    callSnackBar('Ошибка! Попробуйте повторить запрос позже.', context);
  }

  bool validate(SearchParams searchParams) {
    print(searchParams.toJson());
    if (searchParams.faculties.isEmpty ||
        searchParams.genders.isEmpty ||
        searchParams.degrees.isEmpty) return false;

    return true;
  }

  void startFind(BuildContext context) {
    var searchParams = getSearchParams();

    if (validate(searchParams)) {
      DialogLoading dialogLoading = DialogLoading(context: context);
      dialogLoading.show();
      Api.search(searchParams)
          .then((value) => {
                dialogLoading.stop(),
                if (value.isSuccess())
                  {navigatorFunc(value.getData())}
                else
                  {
                    callErrorSnackBar(context),
                  }
              })
          .timeout(Duration(seconds: 10))
          .catchError(
              (obj) => {dialogLoading.stop(), callErrorSnackBar(context)});
    } else {
      callSnackBar('Пожалуйста, выберите все параметры поиска!', context);
    }
  }

  showModalBottomSheet(
      context: context,
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.vertical(top: Radius.circular(25.0)),
      ),
      clipBehavior: Clip.antiAliasWithSaveLayer,
      isScrollControlled: true,
      builder: (BuildContext bc) {
        return Container(
          child: new Wrap(
            children: <Widget>[
              Padding(
                padding: const EdgeInsets.all(8.0),
                child: Center(
                  child: Text(
                    "Критерии поиска",
                    style: TextStyle(fontSize: 18, fontFamily: "Nunito"),
                    textAlign: TextAlign.center,
                  ),
                ),
              ),
              const Divider(indent: 20),
              SeparatorText("Пол"),
              Padding(
                          padding: EdgeInsets.all(7.5), child: _toggleButtonWidget),
                      const Divider(indent: 20),
                      SeparatorText("Курс"),
                      Padding(
                        padding: const EdgeInsets.symmetric(horizontal: 32),
                        child: rangeWidget,
                      ),
                      featuresModalWidget,
                      Padding(
                        padding: const EdgeInsets.all(8.0),
                        child: ButtonContinue(
                            textButton: "Поиск",
                            onPressed: () =>
                                {Navigator.pop(context), startFind(context)}),
                      )
                    ],
                  ),
                );
              });
  }
}

class SeparatorText extends StatelessWidget {
  final String text;

  SeparatorText(this.text);

  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text(
        text,
        style: TextStyle(fontSize: 14, fontFamily: "Nunito"),
        textAlign: TextAlign.center,
      ),
    );
  }
}

class RangeWidget extends StatefulWidget {
  final double min;
  final double max;
  final int divisions;

  final _RangeWidgetState rangeWidgetState;

  RangeWidget(this.rangeWidgetState,
      {Key key, this.min: 1, this.max: 6, this.divisions: 5})
      : super(key: key);

  @override
  _RangeWidgetState createState() => rangeWidgetState;
}

class _RangeWidgetState extends State<RangeWidget> {
  RangeValues currentRangeValues;
  _RangeWidgetState(this.currentRangeValues);

  @override
  Widget build(BuildContext context) {
    return RangeSlider(
      values: currentRangeValues,
      min: widget.min,
      max: widget.max,
      divisions: widget.divisions,
      labels: RangeLabels(
        currentRangeValues.start.round().toString(),
        currentRangeValues.end.round().toString(),
      ),
      onChanged: (RangeValues values) {
        setState(() {
          currentRangeValues = values;
        });
      },
    );
  }
}

class FeaturesModalConfirm extends StatefulWidget {
  final state = _FeaturesModalConfirmState();

  @override
  _FeaturesModalConfirmState createState() => state;
}

class _FeaturesModalConfirmState extends State<FeaturesModalConfirm> {
  var faculties = <Faculty>[];
  var degrees = <Degree>[];

  @override
  Widget build(BuildContext context) {
    return Column(children: <Widget>[
      SmartSelect<Faculty>.multiple(
        title: 'Образовательные программы',
        value: faculties,
        onChange: (selected) => setState(() => faculties = selected.value),
        choiceItems: DataRes.choicesFaculties,
        modalType: S2ModalType.bottomSheet,
        modalConfirm: true,
        modalValidation: (value) {
          return value.length > 0 ? null : 'Выберите образовательные программы';
        },
        modalHeaderStyle: S2ModalHeaderStyle(
          backgroundColor: Theme.of(context).cardColor,
        ),
        modalActionsBuilder: (context, state) {
          return <Widget>[
            Padding(
              padding: const EdgeInsets.only(right: 13),
              child: state.choiceSelector,
            )
          ];
        },
        modalDividerBuilder: (context, state) {
          return const Divider(height: 1);
        },
        modalFooterBuilder: (context, state) {
          return Container(
            padding: const EdgeInsets.symmetric(
              horizontal: 12.0,
              vertical: 7.0,
            ),
            child: Row(
              children: <Widget>[
                const Spacer(),
                FlatButton(
                  child: const Text('Cancel'),
                  onPressed: () => state.closeModal(confirmed: false),
                ),
                const SizedBox(width: 5),
                FlatButton(
                  child: Text('OK (${state.changes.value.length})'),
                  color: Theme.of(context).primaryColor,
                  textColor: Colors.white,
                  onPressed: state.changes.value.isNotEmpty
                      ? () => state.closeModal(confirmed: true)
                      : null,
                ),
              ],
            ),
          );
        },
      ),
      const Divider(indent: 20),
      SmartSelect<Degree>.multiple(
        title: 'Степень образования',
        value: degrees,
        onChange: (selected) => setState(() => degrees = selected.value),
        choiceItems: DataRes.choicesDegrees,
        modalType: S2ModalType.bottomSheet,
        modalConfirm: true,
        modalValidation: (value) {
          return value.length > 0 ? null : 'Выберите степень образования';
        },
        modalHeaderStyle: S2ModalHeaderStyle(
          backgroundColor: Theme.of(context).cardColor,
        ),
        modalActionsBuilder: (context, state) {
          return <Widget>[
            Padding(
              padding: const EdgeInsets.only(right: 13),
              child: state.choiceSelector,
            )
          ];
        },
        modalDividerBuilder: (context, state) {
          return const Divider(height: 1);
        },
        modalFooterBuilder: (context, state) {
          return Container(
            padding: const EdgeInsets.symmetric(
              horizontal: 12.0,
              vertical: 7.0,
            ),
            child: Row(
              children: <Widget>[
                const Spacer(),
                FlatButton(
                  child: const Text('Cancel'),
                  onPressed: () => state.closeModal(confirmed: false),
                ),
                const SizedBox(width: 5),
                FlatButton(
                  child: Text('OK (${state.changes.value.length})'),
                  color: Theme.of(context).primaryColor,
                  textColor: Colors.white,
                  onPressed: state.changes.value.isNotEmpty
                      ? () => state.closeModal(confirmed: true)
                      : null,
                ),
              ],
            ),
          );
        },
      ),
    ]);
  }
}
```

1.32. home_loading.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/rendering.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/data/meet_status.dart';
import 'package:hse_coffee/ui/widgets/dialog_loading.dart';
import 'package:progress_indicators/progress_indicators.dart';

class HomeLoadingScreen extends StatefulWidget {
  final bool withCancel;
  final Function(MeetStatus meetStatus) navigatorFunc;

  static const String routeName = "/home/loading";

  HomeLoadingScreen({this.withCancel: false, this.navigatorFunc});

  @override
  _HomeLoadingScreen createState() =>
      _HomeLoadingScreen(withCancel, navigatorFunc);
```

```
}

class _HomeLoadingScreen extends State<HomeLoadingScreen> {
  final bool withCancel;
  final Function(MeetStatus meetStatus) navigatorFunc;

  _HomeLoadingScreen(this.withCancel, this.navigatorFunc);

  void callSnackBar(String text, BuildContext context) {
    ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(behavior: SnackBarBehavior.floating, content: Text(text)));
  }

  void callErrorSnackBar(BuildContext context) {
    callSnackBar('Ошибка! Попробуйте повторить запрос позже.', context);
  }

  @override
  void initState() {
    super.initState();
  }

  _cancelMeet() {
    if (withCancel) {
      var dialogLoading = DialogLoading(context: context);
      dialogLoading.show();
      Api.cancelSearch()
          .then((value) => {
                dialogLoading.stop(),
                if (value.isSuccess())
                  {navigatorFunc(value.getData())}
                else
                  {
                    callErrorSnackBar(context),
                  }
              })
          .timeout(Duration(seconds: 10))
          .catchError(
              (obj) => {dialogLoading.stop(), callErrorSnackBar(context)});
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        floatingActionButton: withCancel
            ? FloatingActionButton(
                onPressed: _cancelMeet,
                child: Icon(Icons.cancel),
                backgroundColor: Colors.white,
                foregroundColor: Colors.blueAccent,
              )
            : null,
        backgroundColor: Color.fromRGBO(67, 100, 248, 0.98),
        body: CustomPaint(
            painter: Painter(),
            child: Center(
              child: Stack(children: [
                Center(
                  child: Image.asset(
                    "images/bird_with_coffee.png",
                    scale: 4,
                  ),
                ),
                Positioned(
                  right: MediaQuery.of(context).size.width / 2 - 43,
                  bottom: MediaQuery.of(context).size.height / 2 - 130,
                  child: JumpingDotsProgressIndicator(
                    color: Colors.white,
                    numberOfDots: 5,
                    fontSize: 36.0,
                  ),
                ),
              ]),
            )));
  }
}

class Painter extends CustomPainter {
  @override
  void paint(Canvas canvas, Size size) {
    final height = size.height;
    final width = size.width;

    Paint paint = Paint();
    var path = Path();

    paint.color = Color.fromRGBO(255, 255, 255, 1.0);

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.25, height * 0.1), radius: height * 0.2));

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.9, height * 0.9), radius: height * 0.3));

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.9, height * 0.9), radius: height * 0.25));

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.9, height * 0.9), radius: height * 0.2));

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.05, height * 0.6), radius: height * 0.16));

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.95, height * 0.3), radius: height * 0.15));

    canvas.drawPath(path, paint);
    path.close();

    var path2 = Path();
    paint.color = Color.fromRGBO(67, 100, 248, 0.98);
    path2.addOval(Rect.fromCircle(
        center: Offset(width * 0.95, height * 0.95), radius: height * 0.16));
    canvas.drawPath(path2, paint);

    canvas.drawPath(path2, paint);
  }

  @override
  bool shouldRepaint(CustomPainter oldDelegate) {
    return oldDelegate != this;
  }
}
```

1.33. home_meets.dart

```
import 'package:cached_network_image/cached_network_image.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/rendering.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/data/meet.dart';
import 'package:hse_coffee/ui/auth/header.dart';

import 'home_person_info.dart';

class HomeMeetsScreen extends StatefulWidget {
  static const String routeName = "/home/meets";

  @override
  _HomeMeetsScreen createState() => _HomeMeetsScreen();
}

class _HomeMeetsScreen extends State<HomeMeetsScreen> {
  List<Meet> _meets;

  @override
  void initState() {
    super.initState();

    loadMeets();
  }

  void loadMeets() {
    Api.getMeets().then((value) => {
          if (value.isSuccess()) {_meets = value.getData(), setState(() {})}
        });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        body: Builder(
      builder: (context) => Column(mainAxisSize: MainAxisSize.max, children: [
        Header(title: "История встреч"),
        if (_meets == null)
          Center(child: CircularProgressIndicator())
        else if (_meets.isNotEmpty)
          SingleChildScrollView(
              child: GridView.count(
            physics: NeverScrollableScrollPhysics(),
            shrinkWrap: true,
            scrollDirection: Axis.vertical,
            padding: const EdgeInsets.all(20.0),
            crossAxisSpacing: 10.0,
            crossAxisCount: 2,
            children: List.generate(_meets.length, (index) {
              return Center(child: MeetCard(_meets[index]));
            }),
          ))
        else
          Expanded(
            child: Center(
              child: Text(
                "У Вас ещё не было встреч.\n"
                " Самое время начать поиск!",
                textAlign: TextAlign.center,
                style: TextStyle(fontSize: 12.0, fontFamily: "Nunito"),
              ),
            ),
          ),
      ]),
    ));
  }
}

class MeetCard extends StatelessWidget {
  final Meet meet;

  MeetCard(this.meet, {Key key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    var user =
        UserStorage().user.email == meet.user1.email ? meet.user2 : meet.user1;
    var date = meet.createdDate.toLocal();

    void onTap() {
      Navigator.push(
        context,
        MaterialPageRoute(builder: (context) => HomePersonScreen(user)),
      );
    }

    return GestureDetector(
      onTap: onTap,
      child: new Container(
          width: double.infinity,
          decoration: new BoxDecoration(
            boxShadow: [
              new BoxShadow(
                color: Colors.grey.withOpacity(.5),
                blurRadius: 5.0, // soften the shadow
                spreadRadius: 5.0, //extend the shadow
                offset: Offset(
                  5.0, // Move to right 10  horizontally
                  5.0, // Move to bottom 10 Vertically
                ),
              ),
            ],
          ),
          child: Card(
              clipBehavior: Clip.antiAliasWithSaveLayer,
              shadowColor: Colors.black,
              shape: RoundedRectangleBorder(
                borderRadius: BorderRadius.circular(10),
              ),
              elevation: 5,
              child: Column(
                mainAxisSize: MainAxisSize.min,
                crossAxisAlignment: CrossAxisAlignment.stretch,
                children: <Widget>[
                  Container(
                    height: 110,
                    child: CachedNetworkImage(
                        imageUrl: Api.getImageUrlByUser(user),
                        fit: BoxFit.cover),
                  ),
                  Padding(
                      padding: EdgeInsets.only(
                          top: 2, bottom: 1, right: 12, left: 12),
                      child: Center(
                          child: Text(
                        user.firstName + " " + user.lastName,
                        textAlign: TextAlign.center,
                        style: TextStyle(fontSize: 12.0, fontFamily: "Nunito"),
                      ))),
                  Padding(
                      padding: EdgeInsets.symmetric(vertical: 1.0),
                      child: Row(
                        mainAxisAlignment: MainAxisAlignment.center,
                        //Center Row contents horizontally,
                        crossAxisAlignment: CrossAxisAlignment.center,
                        children: <Widget>[
                          Padding(
                              padding: EdgeInsets.symmetric(horizontal: 3),
                              child: Icon(
                                Icons.person,
                                color: Colors.blueAccent,
                                size: 12,
                              )),
                          Text(
                            "${date.day}.${date.month.toString().length == 1 ? "0" +
date.month.toString() : date.month}.${date.year}",
                          style: TextStyle(fontSize: 10.0),
                        )
                      ],
                    ))
                ],
              ))),
    );
  }
}
```

1.34. home_person_info.dart

```
import 'package:cached_network_image/cached_network_image.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/data/user.dart';
import 'package:hse_coffee/ui/widgets/auth_faculty_items.dart';
import 'package:hse_coffee/ui/widgets/button_continue.dart';

class HomePersonScreen extends StatefulWidget {
  final bool withBack;

  static const String routeName = "/home/person";
  final User user;

  HomePersonScreen(this.user, {this.withBack: true});

  @override
  _HomePersonScreen createState() => _HomePersonScreen(user);
}

class _HomePersonScreen extends State<HomePersonScreen> {
  final User user;

  _HomePersonScreen(this.user);

  @override
  void initState() {
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
        body: Builder(
            builder: (context) => SingleChildScrollView(
                  child: Column(
                    mainAxisSize: MainAxisSize.min,
                    children: [
                      Container(
                        width: double.infinity,
                        child: CachedNetworkImage(
                          alignment: Alignment.bottomLeft,
                          fit: BoxFit.cover,
                          imageUrl: Api.getImageUrlByUser(user),
                          errorWidget: (context, url, dynamic) =>
                              new Icon(Icons.error),
                        ),
                      ),
                      Padding(
                        padding: const EdgeInsets.only(
                            top: 10, right: 12, left: 12, bottom: 0),
                        child: Text(
                          user.getFullName(),
                          style: TextStyle(
                              fontSize: 22,
                              fontFamily: "Nunito",
                              fontWeight: FontWeight.bold),
                          textAlign: TextAlign.center,
                        ),
                      ),
                      Padding(
                        padding: const EdgeInsets.all(4.0),
                        child: Text(
                          DataRes.getFacultyName(user.faculty) +
                              ", ${user.course} курс",
                          style: TextStyle(
                              fontSize: 16,
                              fontFamily: "Nunito",
                              fontWeight: FontWeight.bold),
                          textAlign: TextAlign.center,
                        ),
                      ),
                      user.aboutMe.isEmpty
                          ? Container()
                          : Padding(
                              padding:
                                  const EdgeInsets.symmetric(horizontal: 8.0),
                              child: Text(
                                "\"${user.aboutMe.replaceAll("\n\n", "\n")}\"",
                                style: TextStyle(
                                  fontSize: 14,
                                  fontStyle: FontStyle.italic,
                                  fontFamily: "Nunito",
                                ),
                                textAlign: TextAlign.center,
                              ),
                            ),
                      const Divider(height: 24, thickness: 2),
                      Padding(
                        padding: const EdgeInsets.all(8.0),
                        child: Column(
                          mainAxisSize: MainAxisSize.min,
                          crossAxisAlignment: CrossAxisAlignment.center,
                          children: [
                            DataCard("images/icons/tg.png", user.getTelegram()),
                            DataCard("images/icons/vk.png", user.getVk()),
                            DataCard("images/icons/inst.png", user.getInst()),
                            ],
                          ),
                        ),
                        widget.withBack
                            ? Padding(
                                padding: EdgeInsets.symmetric(
                                    horizontal: 16, vertical: 32),
                                child: ButtonContinue(
                                    textButton: "Вернуться назад",
                                    onPressed: () => Navigator.pop(context)))
                            : Container()
                      ],
                    ),
                  )));
  }
}

class DataCard extends StatelessWidget {
  final String _text;
  final String _assetPath;
  final double sizeIco;
  final Function onTap;

  DataCard(this._assetPath, this._text, {this.sizeIco: 32.0, this.onTap});

  @override
  Widget build(BuildContext context) {
    if (_text.isEmpty) return Container();
    return GestureDetector(
      onTap: onTap == null ? () => print("null func") : onTap(),
      child: Padding(
        padding: const EdgeInsets.symmetric(horizontal: 16.0),
        child: Card(
          child: Center(
            child: Row(
                mainAxisSize: MainAxisSize.max,
                mainAxisAlignment: MainAxisAlignment.center,
                crossAxisAlignment: CrossAxisAlignment.center,
                children: [
                  Padding(
                    padding: const EdgeInsets.all(8.0),
                    child: Image.asset(_assetPath,
                        width: sizeIco, height: sizeIco),
                  ),
                  Text(
                    _text,
                    textAlign: TextAlign.left,
                  )
                ]),
          ),
        ),
      ),
    );
  }
}
```

1.35. splash.dart

```
import 'dart:async';
import 'dart:core';

import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:hse_coffee/business_logic/api.dart';
import 'package:hse_coffee/business_logic/auth.dart';
import 'package:hse_coffee/business_logic/user_storage.dart';
import 'package:hse_coffee/router_auth.dart';
import 'package:hse_coffee/ui/splash/splash_background.dart';

class SplashScreen extends StatefulWidget {
  final String nextRoute;

  SplashScreen({this.nextRoute});

  // все подобные виджеты должны создавать своё состояние,
  // нужно переопределять данный метод
  @override
  State<StatefulWidget> createState() => _SplashScreenState();
}

// Создаём состояние виджета
class _SplashScreenState extends State<SplashScreen> {
  // Инициализация состояния
  @override
  void initState() {
    super.initState();
    Timer(Duration(seconds: 3), () {
      Auth.getData()
          .then((value) => {
                if (value.containsKey(Auth.refreshTokenKey))
                  {
                    Api.getUser().then((value) => {
                          if (value.isSuccess())
                            {
                              UserStorage.instance.user = value.getData(),
                              RouterHelper.routeByUser(context, value.getData())
                            }
                          else
                            {
                              Navigator.of(context)
                                  .pushReplacementNamed(widget.nextRoute)
                            }
                        })
                  }
                else
                  {Navigator.of(context).pushReplacementNamed(widget.nextRoute)}
              })
          .timeout(Duration(seconds: 5))
          .catchError((obj) =>
              {Navigator.of(context).pushReplacementNamed(widget.nextRoute)});
    });
  }

  // Формирование виджета
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        body: CustomPaint(
            painter: SplashPainter(),
            child: Column(children: <Widget>[
              Container(
                  width: double.infinity,
                  child: Padding(
                      padding:
                          EdgeInsets.symmetric(vertical: 25, horizontal: 25),
                      child: RichText(
                          text: TextSpan(children: <TextSpan>[
                        TextSpan(
                            text: "Добро\nпожаловать\nв ",
                            style: TextStyle(
                                fontFamily: 'Nunito',
                                color: Colors.black,
                                fontSize: 48,
                                fontWeight: FontWeight.bold)),
                        TextSpan(
                            text: "HSEcoffee",
                            style: TextStyle(
                                fontFamily: 'Nunito',
                                color: Colors.blueAccent,
                                fontSize: 52,
                                fontWeight: FontWeight.bold))
                      ])))),
              Transform.rotate(
                  angle: 76,
                  child: Image(image: AssetImage('images/bird.png'))),
              Expanded(
                  child: Padding(
                padding: EdgeInsets.all(15),
                child: Align(
                    alignment: Alignment.bottomRight,
                    child: CircularProgressIndicator(
                        strokeWidth: 4.0, backgroundColor: Colors.white)),
              ))
            ])));
  }
}
```

1.36. splash_background.dart

```
import 'dart:math';

import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';

class SplashPainter extends CustomPainter {
  @override
  void paint(Canvas canvas, Size size) {
    final height = size.height;
    final width = size.width;
    Paint paint = Paint();

    Path mainBackground = Path();
    mainBackground.addRect(Rect.fromLTRB(0, 0, width, height));
    paint.color = Color.fromRGBO(60, 106, 252, 1);
    canvas.drawPath(mainBackground, paint);
    mainBackground.close();

    var path = Path();

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.5, height * 0.5), radius: height * 0.2));

    paint.color = Color.fromRGBO(112, 144, 253, 100);
    canvas.drawPath(path, paint);

    path.addOval(Rect.fromCircle(
        center: Offset(width * 0.1, height), radius: height * 0.2));

    path.addOval(Rect.fromCircle(
        center: Offset(width - width * 0.1, height * 0.8),
        radius: height * 0.3));

    paint.color = Color.fromRGBO(255, 255, 255, 0.2);
    canvas.drawPath(path, paint);

    path.close();

    var parabola = Path();
    parabola.addOval(Rect.fromCircle(
        center: Offset(width * 0.4, -height * 0.3), radius: height * 0.75));

    paint.color = Colors.white;
    canvas.drawPath(parabola, paint);
    parabola.close();
  }

  void drawRandomOvals(
      Canvas canvas, Paint paint, int n, double width, double height) {
    paint.color = Colors.blueAccent;

    for (int i = 0; i < n; i++) {
      var oval = Path();
      oval.addOval(Rect.fromCircle(
          center: Offset(
              width * Random().nextDouble(), height * Random().nextDouble()),
          radius: height * Random().nextDouble() / 15));
      canvas.drawPath(oval, paint);
    }
  }

  @override
  bool shouldRepaint(CustomPainter oldDelegate) {
    return oldDelegate != this;
  }
}
```

1.37. auth_faculty_items.dart

```
import 'package:hse_coffee/data/degree.dart';
import 'package:hse_coffee/data/faculty.dart';
import 'package:smart_select/smart_select.dart';

class DataRes {
  static String getFacultyName(Faculty faculty) {
    if (choicesFaculties.firstWhere((element) => element.value == faculty) ==
        null) {
      return "";
    }

    return choicesFaculties
        .firstWhere((element) => element.value == faculty)
        .title;
  }

  static final choicesDegrees = <S2Choice<Degree>>[
    S2Choice<Degree>(value: Degree.BACHELOR, title: "Бакалавриат"),
    S2Choice<Degree>(value: Degree.MAGISTRACY, title: "Магистратура"),
    S2Choice<Degree>(value: Degree.POSTGRADUATE, title: "Аспирантура"),
    S2Choice<Degree>(value: Degree.SPECIALTY, title: "Специалитет"),
  ];

  static final choicesFaculties = <S2Choice<Faculty>>[
    S2Choice<Faculty>(value: Faculty.BANK, title: "Банковский институт"),
    S2Choice<Faculty>(
        value: Faculty.JURISPRUDENCE,
        title: "Высшая школа юрисиспруденции и администрирования"),
    S2Choice<Faculty>(value: Faculty.BUSINESS, title: "Высшая школа бизнеса"),
    S2Choice<Faculty>(
        value: Faculty.STATISTIC,
        title: "Институт статистических исследований и экономики знаний"),
    S2Choice<Faculty>(value: Faculty.LYCEUM, title: "Лицей НИУ ВШЭ"),
    S2Choice<Faculty>(value: Faculty.ELECTRONIC, title: "МИЭМ"),
    S2Choice<Faculty>(
        value: Faculty.MIEF,
        title: "Международный институт экономики и финансов"),
    S2Choice<Faculty>(
        value: Faculty.BIOLOGY, title: "Факультет биологии и биотехнологии"),
    S2Choice<Faculty>(
        value: Faculty.HUMANITARIAN, title: "Факультет гуманитарных наук"),
    S2Choice<Faculty>(
        value: Faculty.CITY,
        title: "Факультет городского и регионального развития"),
    S2Choice<Faculty>(
        value: Faculty.GEOGRAPHY,
        title: "Факультет географии и геоинформационных технологий"),
    S2Choice<Faculty>(
        value: Faculty.MEDIA, title: "Факультет коммуникация, медиа и дизайна"),
    S2Choice<Faculty>(
        value: Faculty.COMPUTER, title: "Факультет компьютерных наук"),
    S2Choice<Faculty>(value: Faculty.MATH, title: "Факультет математики"),
    S2Choice<Faculty>(
        value: Faculty.WORLD_ECONOMY,
        title: "Факультет мировой экономики и мировой политики"),
    S2Choice<Faculty>(value: Faculty.LAWYER, title: "Факультет права"),
    S2Choice<Faculty>(
        value: Faculty.SOCIAL, title: "Факультет социальных наук"),
    S2Choice<Faculty>(value: Faculty.PHYSICS, title: "Факультет физики"),
    S2Choice<Faculty>(value: Faculty.CHEMICAL, title: "Факультет химии"),
    S2Choice<Faculty>(
        value: Faculty.ECONOMY, title: "Факультет экономических наук"),
    S2Choice<Faculty>(
        value: Faculty.LANGUAGE, title: "Школа иностранных языков"),
  ];
}
```

1.38. button_continue.dart

```
import 'package:flutter/cupertino.dart';
import 'package:gradient_widgets/gradient_widgets.dart';

class ButtonContinue extends StatelessWidget {
  String textButton = "Продолжить";

  ButtonContinue({Key key, this.onPressed, this.textButton}) : super(key: key);
  final Function onPressed;

  @override
  Widget build(BuildContext context) {
    return GradientButton(
      child: Text(textButton == null ? "Продолжить" : textButton,
          style: TextStyle(fontSize: 16.0)),
      callback: () {
        onPressed();
      },
      increaseWidthBy: 150.0,
      increaseHeightBy: 10,
      gradient: LinearGradient(
          begin: Alignment.topLeft,
          end: Alignment.bottomRight,
          colors: [
            Color.fromRGBO(0, 82, 212, 1),
            Color.fromRGBO(49, 94, 252, 1),
            Color.fromRGBO(111, 177, 252, 1)
          ]),
      shadowColor: Gradients.backToFuture.colors.last.withOpacity(0.25),
    );
  }
}
```

1.39. capsule_widget.dart

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

class CapsuleWidget extends StatelessWidget {
  final Color fillColor;
  final Color textColor;
  final String label;
  final double ribbonHeight;
  final double ribbonRadius;

  const CapsuleWidget({
    Key key,
    this.fillColor = Colors.white24,
    this.textColor = Colors.black,
    @required this.label,
    @required this.ribbonHeight,
    this.ribbonRadius = 1000,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Row(
      children: <Widget>[
        Expanded(
          child: Container(
            height: ribbonHeight,
            color: fillColor,
          ),
        ),
        Container(
          decoration: BoxDecoration(
              color: fillColor,
              borderRadius: BorderRadius.circular(ribbonRadius)),
          child: Padding(
            padding: const EdgeInsets.all(6.0),
            child: Text(
              label,
              style: TextStyle(color: textColor, fontWeight: FontWeight.w500),
            ),
          ),
        ),
        Expanded(
          child: Container(
            height: ribbonHeight,
            color: fillColor,
          ),
        ),
      ],
    );
  }
}
```

1.40. dialog_loading.dart

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

class DialogLoading {
  static const String _Default_Text = "Загружаю...";
  final BuildContext context;
  final String text;

  DialogLoading({@required this.context, this.text});

  void show() {
    AlertDialog alert = AlertDialog(
      content: new Row(
        children: [
          CircularProgressIndicator(strokeWidth: 3.0),
          Container(
              margin: EdgeInsets.symmetric(horizontal: 15.0),
              child: Text(text == null ? _Default_Text : text)),
        ],
      ),
    );
    showDialog(
      barrierDismissible: false,
      context: context,
      builder: (BuildContext context) {
        return alert;
      },
    );
  }

  void stop() {
    Navigator.pop(context);
  }
}
```

1.41. edu_fields.dart

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:hse_coffee/data/degree.dart';
import 'package:hse_coffee/data/faculty.dart';
import 'package:smart_select/smart_select.dart';

import 'auth_faculty_items.dart';

class EducationFields extends StatefulWidget {
  final state = _EducationFieldsState();

  @override
  _EducationFieldsState createState() => state;
}
```

```
class _EducationFieldsState extends State<EducationFields> {
  Faculty faculty;
  Degree degree;

  @override
  Widget build(BuildContext context) {
    return Column(children: <Widget>[
      SmartSelect<Faculty>.single(
        title: 'Образовательные программы',
        value: faculty,
        onChange: (selected) => setState(() => faculty = selected.value),
        choiceItems: DataRes.choicesFaculties,
        modalType: S2ModalType.fullPage,
        modalHeaderStyle: S2ModalHeaderStyle(
          backgroundColor: Theme.of(context).cardColor,
        ),
        modalActionsBuilder: (context, state) {
          return <Widget>[
            Padding(
              padding: const EdgeInsets.only(right: 13),
              child: state.choiceSelector,
            )
          ];
        },
        modalDividerBuilder: (context, state) {
          return const Divider(height: 1);
        },
      ),
      const Divider(indent: 20),
      SmartSelect<Degree>.single(
        title: 'Степень образования',
        value: degree,
        onChange: (selected) => setState(() => degree = selected.value),
        choiceItems: DataRes.choicesDegrees,
        modalType: S2ModalType.fullPage,
        modalHeaderStyle: S2ModalHeaderStyle(
          backgroundColor: Theme.of(context).cardColor,
        ),
        modalActionsBuilder: (context, state) {
          return <Widget>[
            Padding(
              padding: const EdgeInsets.only(right: 13),
              child: state.choiceSelector,
            )
          ];
        },
        modalDividerBuilder: (context, state) {
          return const Divider(height: 1);
        },
      ),
    ]);
  }
}
```

1.42. text_field_wrapper.dart

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

class TextFieldWrapper extends StatelessWidget {
  final int minLengthName;
  final int maxLengthName;
  final TextInputType textInputType;
  final String hintText;
  final String labelText;
  final String iconPath;
  final int maxLines;
  final bool ignoreValidate;
  final TextEditingController controller;

  TextFieldWrapper(
      {Key key,
      this.hintText,
      this.labelText,
      this.controller,
      this.minLengthName: 2,
      this.maxLengthName: 15,
      this.textInputType: TextInputType.name,
      this.maxLines: 1,
      this.iconPath,
      this.ignoreValidate: false})
      : super(key: key);

  bool _isValidName(String text) {
    return text != null &&
        text.length > minLengthName &&
        text.length < maxLengthName;
  }

  @override
  Widget build(BuildContext context) {
    return TextFormField(
      maxLines: maxLines,
      keyboardType: textInputType,
      controller: this.controller,
      validator: (input) => ignoreValidate
          ? null
          : _isValidName(input)
              ? null
              : "Пожалуйста, заполните корректно поле!",
      cursorColor: Colors.blue,
      decoration: InputDecoration(
        prefixIcon:
            iconPath != null ? Image.asset(iconPath, scale: 2.25) : null,
        hintText: hintText,
        border: OutlineInputBorder(
          borderRadius: BorderRadius.circular(16.0),
          borderSide: BorderSide(
            width: 2,
            color: Colors.blue,
          ),
        ),
        errorBorder: OutlineInputBorder(
          borderRadius: BorderRadius.circular(16.0),
          borderSide: BorderSide(
            width: 2,
            color: Colors.red,
          ),
        ),
        labelText: labelText,
      ),
    );
  }
}
```

1.43. toggle_button_gender.dart

```
import 'package:flutter/cupertino.dart';
import 'package:hse_coffee/data/gender.dart';
import 'package:hse_coffee/ui/auth/auth_gender.dart';

class ToggleButtonGender extends StatefulWidget {
  final bool onlySingleChose;
  final ToggleButtonGenderState buttonState;
  final Function onPressed;

  ToggleButtonGender(
      {Key key, this.onPressed, this.buttonState, this.onlySingleChose: true})
      : super(key: key);

  @override
  State<StatefulWidget> createState() {
    return buttonState;
  }
}

class ToggleButtonGenderState extends State<ToggleButtonGender> {
  var isSelected = <bool>[false, false];
  static const String FirstButtonText = "Мужской";
  static const String SecondButtonText = "Женский";

  List<Gender> getGenders() {
    List<Gender> genders = List.of({}, growable: true);
    if (isSelected.length >= 1) {
      if (isSelected[0]) {
        genders.add(Gender.MALE);
      }
      if (isSelected.length >= 2 && isSelected[1]) {
        genders.add(Gender.FEMALE);
      }
    }

    return genders;
```

```
}

ToggleButtonGenderState(this.isSelected);

void onClicked(int index) {
  if (!widget.onlySingleChose) {
    isSelected[index] = !isSelected[index];

    bool allOff = true;
    for (int i = 0; i < isSelected.length; i++) {
      if (isSelected[i]) {
        allOff = false;
        break;
      }
    }

    if (allOff) {
      isSelected[index] = !isSelected[index];
      return;
    }
  } else {
    if (isSelected[index]) return;

    bool state = isSelected[index];
    isSelected[index] = !state;
    for (int i = 0; i < isSelected.length; i++) {
      if (i != index) {
        isSelected[i] = state;
      }
    }
  }
  setState(() {});
}

@override
Widget build(BuildContext context) {
  return Center(
      child: Row(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: <Widget>[
        Padding(
          padding: const EdgeInsets.all(4.0),
          child: TButton(
            FirstButtonText,
            () => onClicked(0),
            isClicked: isSelected[0],
          ),
        ),
        Padding(
          padding: const EdgeInsets.all(4.0),
          child: TButton(
            SecondButtonText,
            () => onClicked(1),
            isClicked: isSelected[1],
          ),
          ),
        ]));
  }
}
```

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц в докум.)	№ документа	Входящий № сопроводительного докум. и дата	Подп.	Дата
Изм.	Измененных	Замененных	Новых	Аннулированных					