

ANKARA ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



BLM4537 - iOS ile Mobil Uygulama Geliştirme

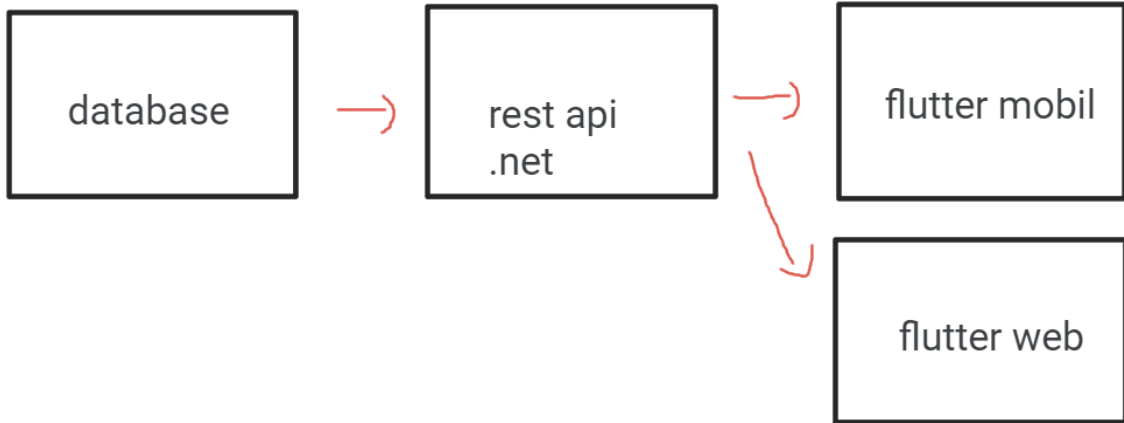
**Proje Raporu
OBS Uygulaması**

Refik Can ÖZTAŞ

19290266

12/01/2023

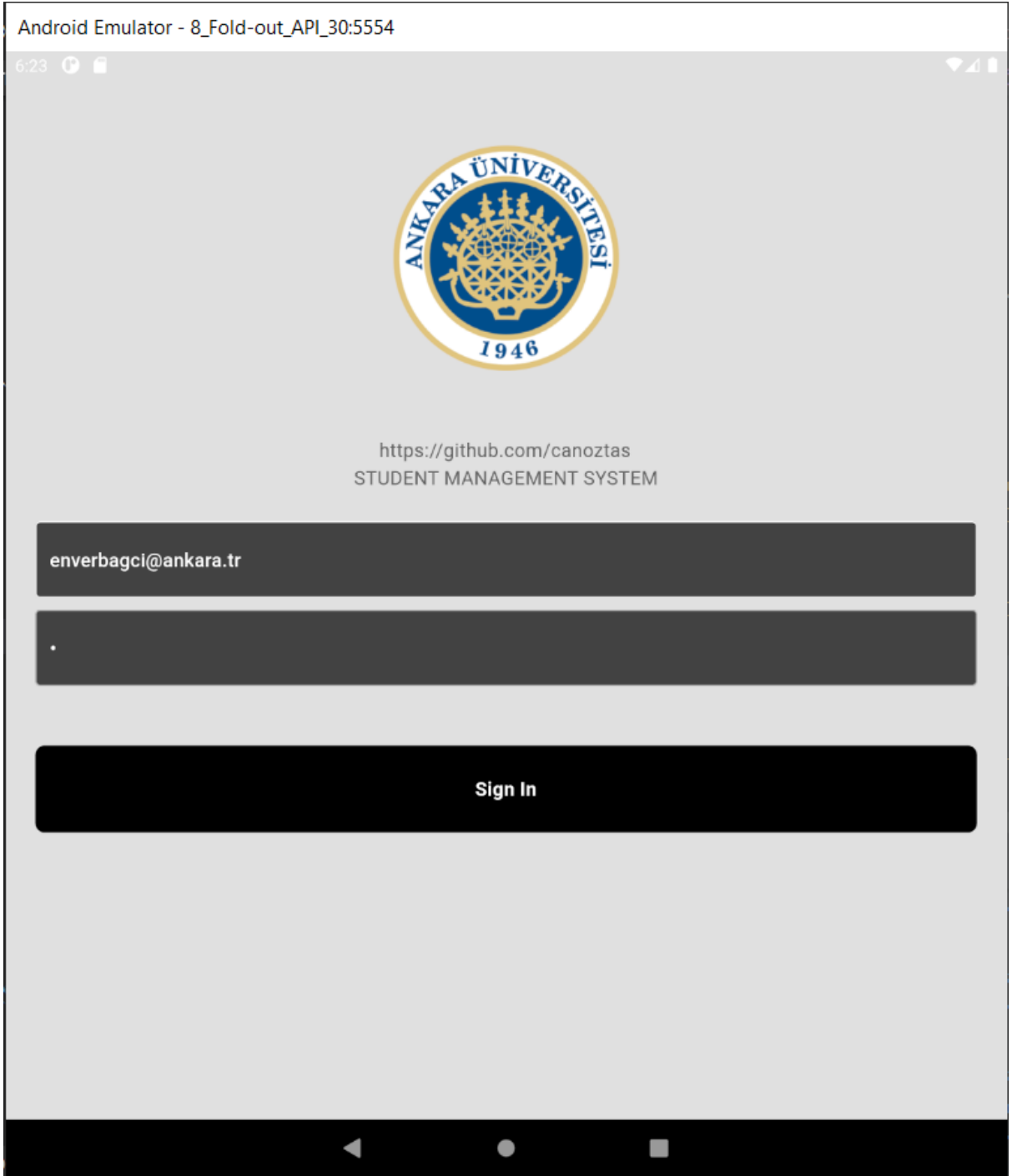
BLM4537 için bir OBS uygulaması yazılmıştır. Web ve mobil uygulamanın çalışma pipeline'ı kabaca aşağıdaki gibidir:



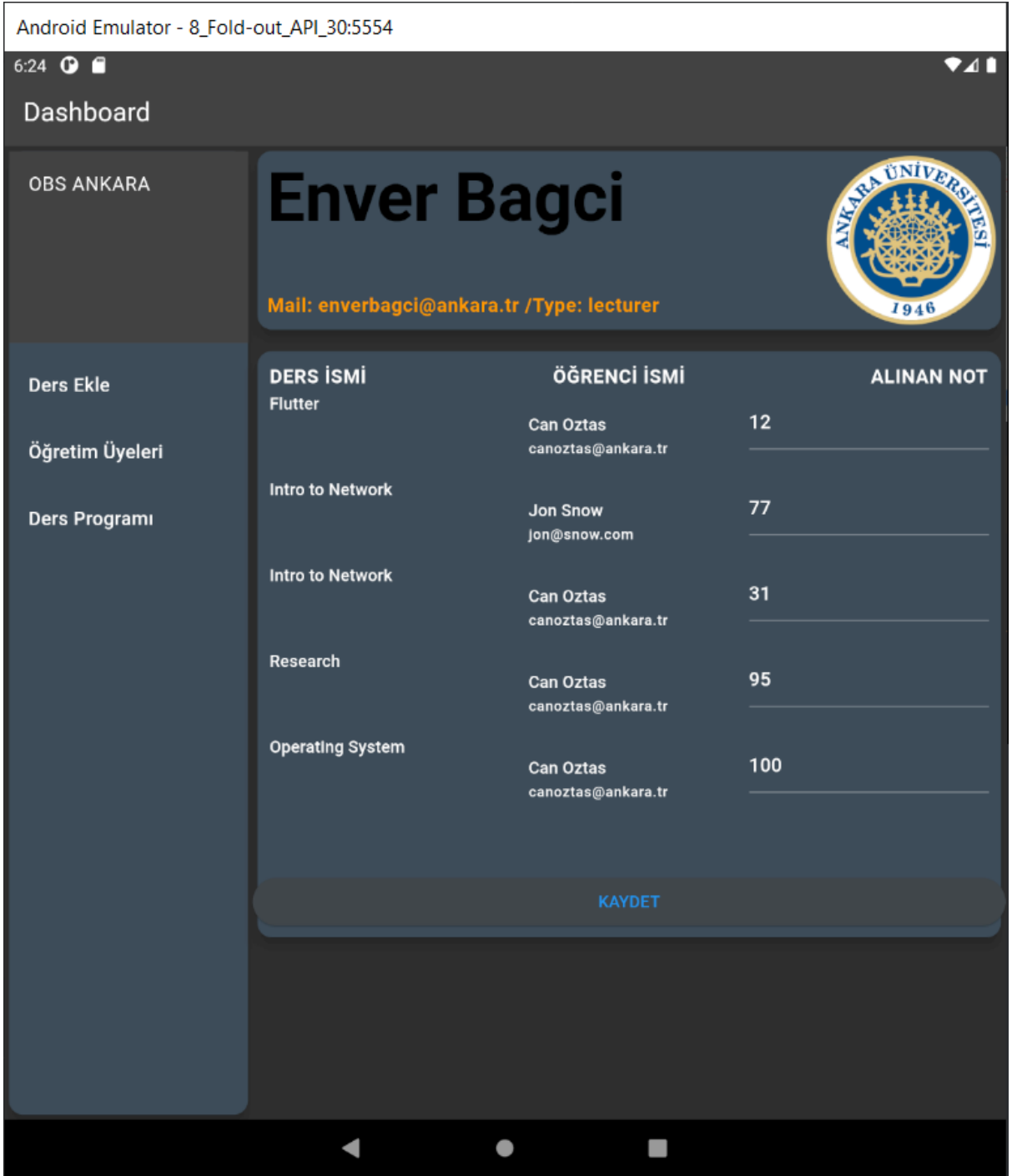
mobil uygulama : <https://github.com/canoztas/StudentManagementSystemMobile>
api : <https://github.com/canoztas/StudentManagementSystemAPI>

Uygulama içi görüntüler:

UI kısmında web uygulamasında farklı olarak bazı değişiklikler yapılmıştır.



Login ekranı



Öğretmen not düzenleme ekranı, gelen jwt cookie'ye göre yetkilendirme ile bu ekrana ulaşılabilir, öğrenci tipi kullanıcı bu ekrana ulaşamıyor.

Ders ekle (not giriş ekranı)



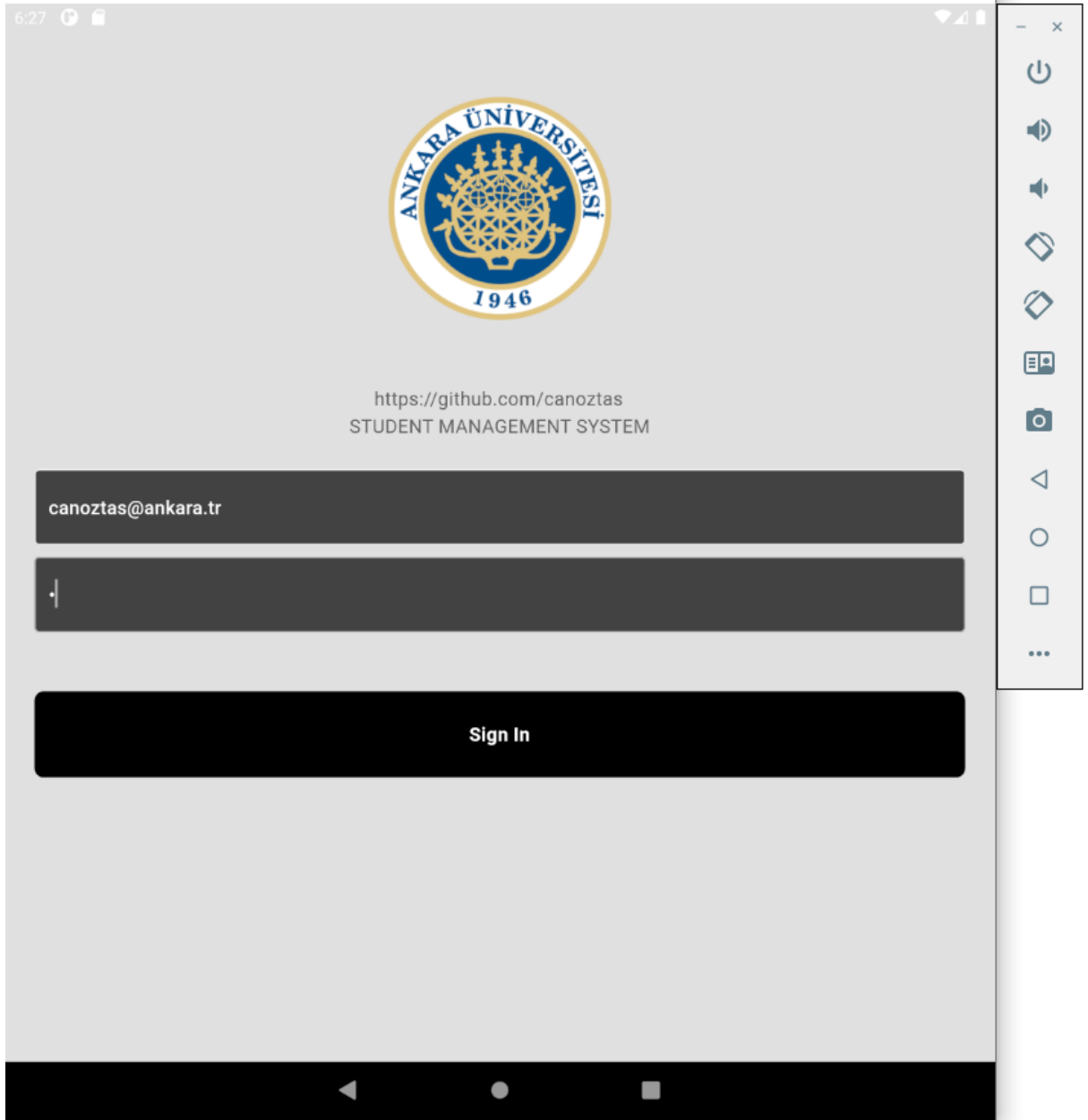
Yetkilendirme ile öğretmen tipi tüm derslere ait öğretmenleri görebiliyor.

Öğretim Üyeleri Ekranı

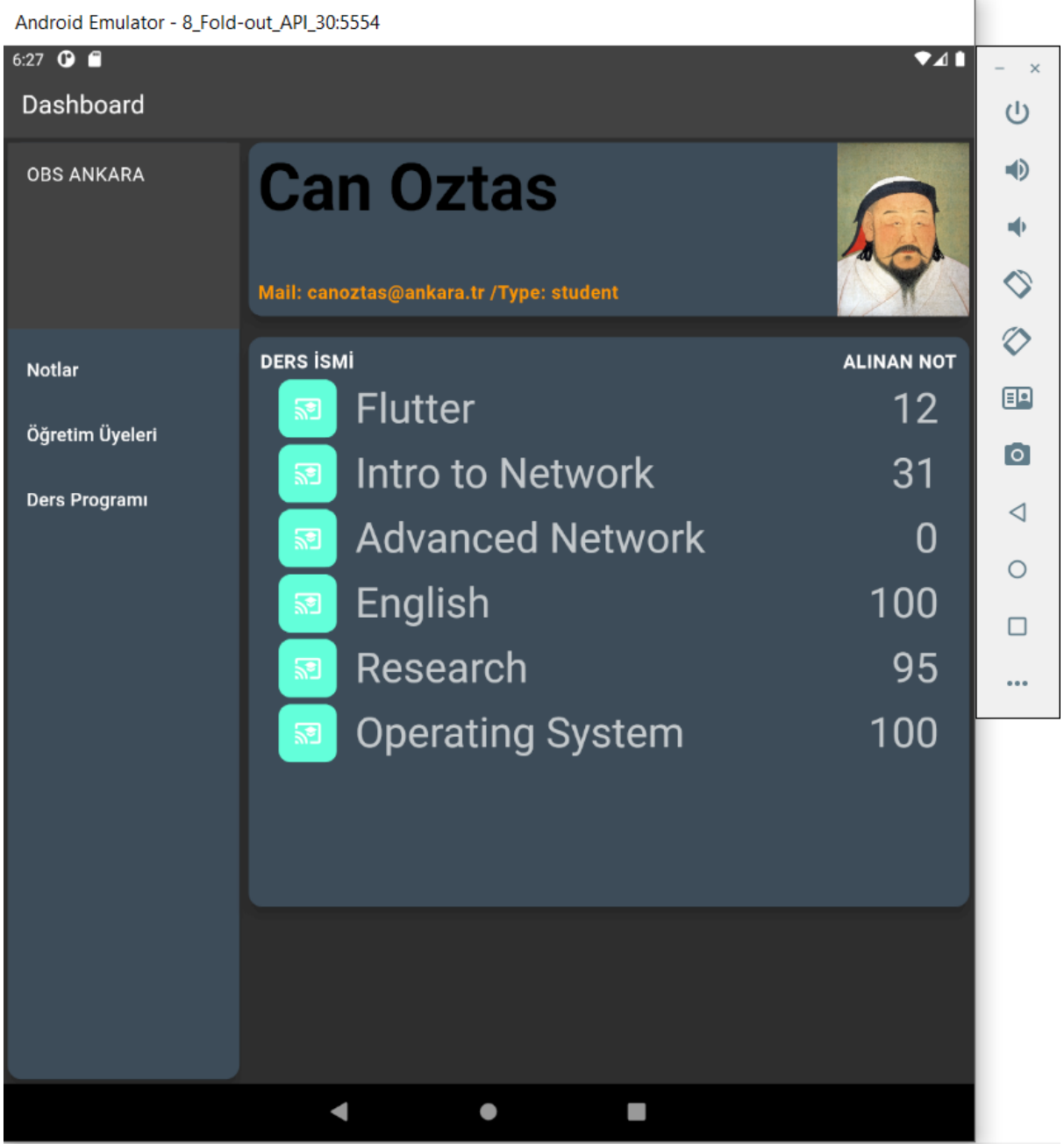


Öğretmen kendine ait dersleri görüntülüyor.

Ders Programı ekranı

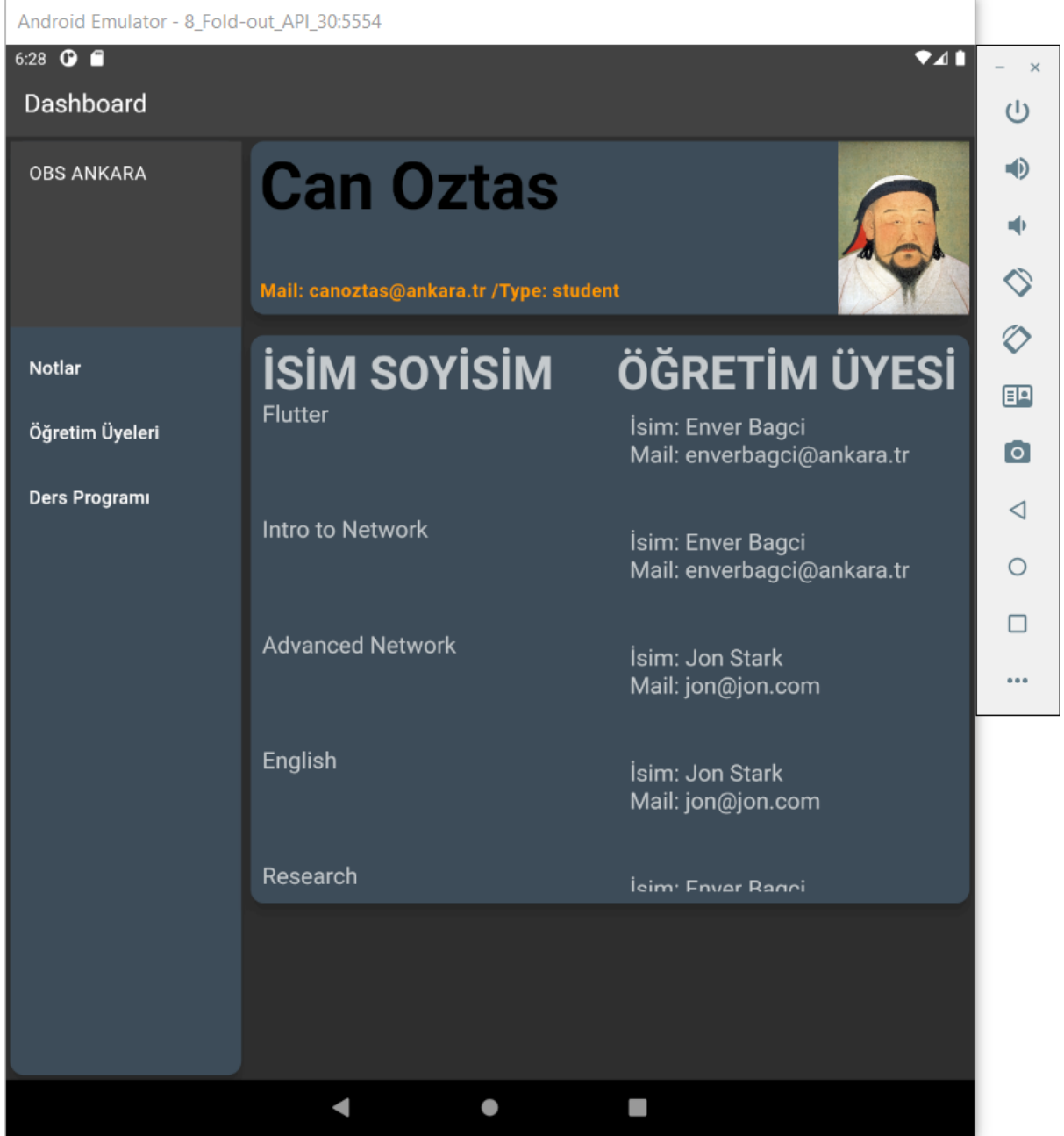


Öğrenci tipi kullanıcı giriş yapıyor.



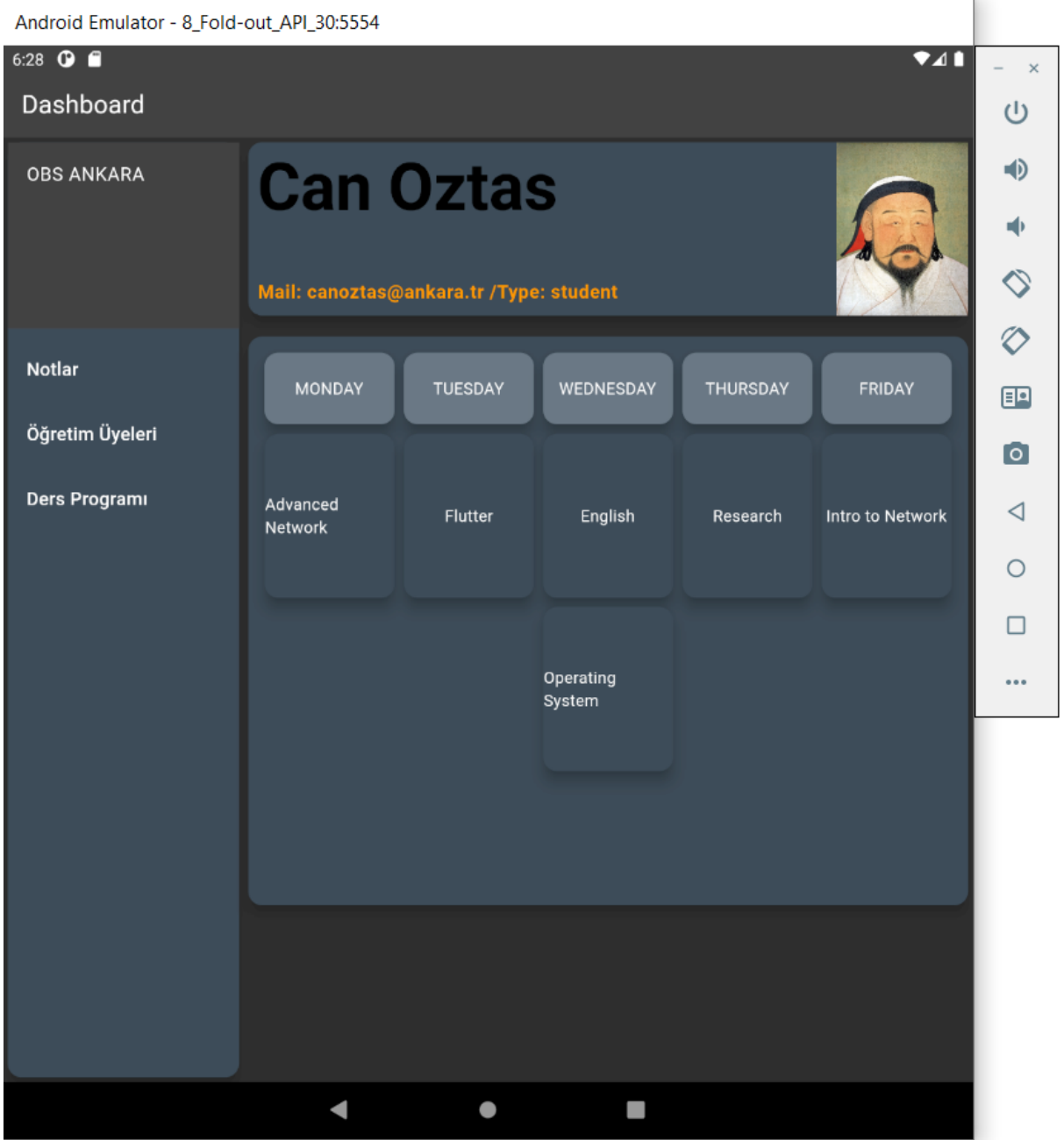
Öğrenci tipi kullanıcı sadece kendi derslerini görebiliyor.

Notlar Ekranı



Öğretim üyelerinde sadece öğrenciye ders verilen öğretmenler görüntüleniyor.

Öğretim Üyeleri Ekranı

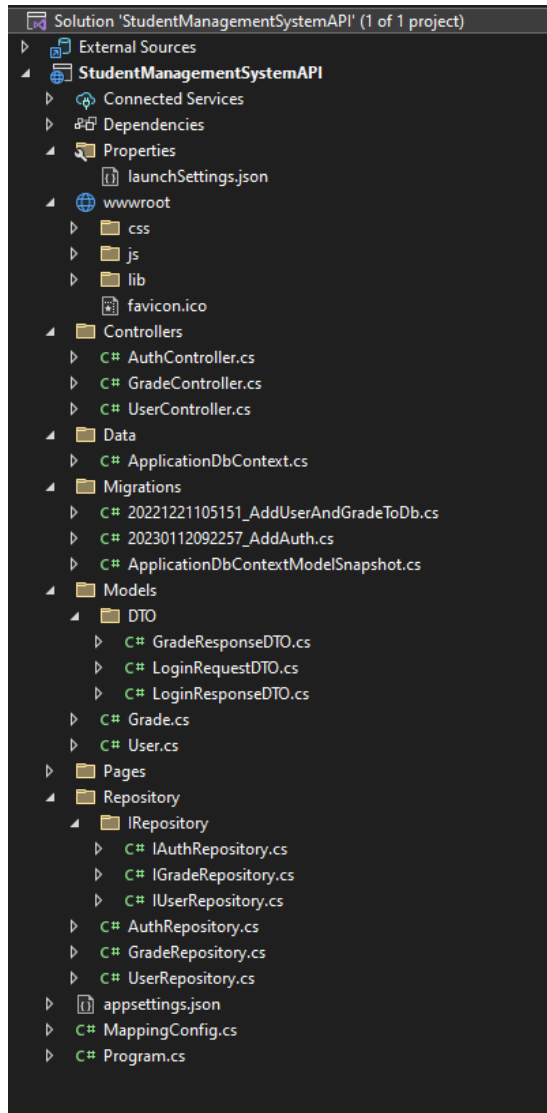


Öğrencinin kendi ders programı

Ders Programı Ekranı

Backend Kod Kısmı:

API kısmında:



Şeklinde bir yapı kurulmuştur.

Modeller iki tanedir:

```
namespace StudentManagementSystemAPI.Models
{
    26 references
    public class User
    {
        8 references
        public int UserId { get; set; }
        4 references
        public string FirstName { get; set; }
        4 references
        public string LastName { get; set; }
        7 references
        public string Email { get; set; }
        10 references
        public string PasswordHash { get; set; }
        4 references
        public string PhotoPath { get; set; }
        5 references
        public string UserType { get; set; }
    }
}
```

```
namespace StudentManagementSystemAPI.Models
{
    25 references
    public class Grade
    {
        5 references
        public int GradeId { get; set; }
        5 references
        public string LessonName { get; set; }
        6 references
        public int LecturerId { get; set; }
        12 references
        public int StudentId { get; set; }
        9 references
        public int Score { get; set; }
        9 references
        public string LessonDate { get; set; }
    }
}
```

İki modelin de IRepository'si ve Repository'leri vardır:

```
using StudentManagementSystemAPI.Models;

namespace StudentManagementSystemAPI.Repository.IRepository
{
    4 references
    public interface IUserRepository
    {
        2 references
        Task<User> GetAsync(int id);
        2 references
        Task CreateAsync(User user);
        2 references
        Task UpdateAsync(User user);
        2 references
        Task RemoveAsync(User user);
        1 reference
        Task SaveAsync();
    }
}
```

```
namespace StudentManagementSystemAPI.Repository
{
    2 references
    public class UserRepository : IUserRepository
    {
        private readonly ApplicationDbContext _context;

        0 references
        public UserRepository(ApplicationDbContext context)
        {
            _context = context;
        }

        2 references
        public async Task<User> GetAsync(int id)
        {
            return await _context.Users.FindAsync(id);
        }

        2 references
        public async Task CreateAsync(User user)
        {
            await _context.Users.AddAsync(user);
        }

        2 references
        public async Task UpdateAsync(User user)
        {
            _context.Users.Update(user);
        }

        2 references
        public async Task RemoveAsync(User user)
        {
            _context.Users.Remove(user);
        }
    }
}
```

```
using Microsoft.EntityFrameworkCore;
using StudentManagementSystemAPI.Models;
using StudentManagementSystemAPI.Models.DTO;

namespace StudentManagementSystemAPI.Repository.IRepository
{
    4 references
    public interface IGradeRepository
    {
        2 references
        Task<GradeResponseDTO> GetAsync(int id);
        2 references
        Task CreateAsync(Grade grade);
        2 references
        Task UpdateAsync(Grade grade);
        2 references
        Task RemoveAsync(Grade grade);
        2 references
        List<GradeResponseDTO> GetByStudentId(int id);
        2 references
        List<GradeResponseDTO> GetByLecturerId(int id);
        1 reference
        Task SaveAsync();
    }
}
```

```
namespace StudentManagementSystemAPI.Repository
{
    2 references
    public class GradeRepository : IGradeRepository
    {
        private readonly ApplicationDbContext _context;

        0 references
        public GradeRepository(ApplicationDbContext context)
        {
            _context = context;
        }

        2 references
        public async Task<GradeResponseDTO> GetAsync(int id)
        {
            Grade grade = new Grade();
            grade = await _context.Grades.FindAsync(id);
            List<User> query = _context.Users.ToList();
            GradeResponseDTO gradeResponseDTO = new GradeResponseDTO()
            {
                Grade = grade,
                Lecturer = query.SingleOrDefault(g => g.UserId == grade.LecturerId),
                Student = query.SingleOrDefault(g => g.UserId == grade.StudentId)
            };
            gradeResponseDTO.Student.PasswordHash = null;
            gradeResponseDTO.Lecturer.PasswordHash = null;
            return gradeResponseDTO;
        }
    }
}
```

```
12 references
public class ApplicationDbContext : DbContext
{
    0 references
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options)
    {
        Database.EnsureDeleted();
        Database.EnsureCreated();
    }

    8 references
    public DbSet<User> Users { get; set; }
    6 references
    public DbSet<Grade> Grades { get; set; }
}
```

Routing ise Controllerlar ile sağlanmıştır:

```
namespace StudentManagementSystemAPI.Controllers
{
    [Route("api/[controller]")]
    public class GradeController : Controller
    {
        private readonly IGradeRepository _gradeRepository;

        public GradeController(IGradeRepository gradeRepository)
        {
            _gradeRepository = gradeRepository;
        }

        [HttpGet]
        public async Task<ActionResult<Grade>> Get(int? gradeId, int? lecturerId, int? studentId)
        {
            if (gradeId != null)
            {
                var grade = await _gradeRepository.GetAsync(gradeId.Value);
                return Ok(grade);
            }
            if (studentId != null)
            {
                var grade = _gradeRepository.GetByStudentId(studentId.Value);
                return Ok(grade);
            }
            if (lecturerId != null)
            {
                var grade = _gradeRepository.GetByLecturerId(lecturerId.Value);
                return Ok(grade);
            }
        }
    }
}
```

```
[Route("api/[controller]")]
public class UserController : Controller
{
    private readonly IUserRepository _userRepository;

    public UserController(IUserRepository userRepository)
    {
        _userRepository = userRepository;
    }

    [HttpGet]
    public async Task<ActionResult<User>> Get(int id)
    {
        var user = await _userRepository.GetAsync(id);
        return Ok(user);
    }

    [HttpPost]
    public async Task<ActionResult> Create(User user)
    {
        await _userRepository.CreateAsync(user);
        return Ok();
    }

    [HttpPut]
    public async Task<ActionResult> Update(User user)
    {
        await _userRepository.UpdateAsync(user);
        return Ok();
    }
}
```

Uygulamanın önemli bir kısmını oluşturan

yetkilendirme için jwt kullanılmıştır.

```

4 references
public async Task<LoginResponseDTO> Login(LoginRequestDTO loginRequestDTO)
{
    var user = _db.Users.SingleOrDefault(x => x.Email == loginRequestDTO.Email && x.PasswordHash == loginRequestDTO.Password);
    //bool isValid = await _userManager.CheckPasswordAsync(user, loginRequestDTO.Password);

    //user not found
    if (user == null)
    {
        return null;
    }

    //var roles = await _userManager.GetRolesAsync(user);

    var tokenHandler = new JwtSecurityTokenHandler();
    var key = Encoding.ASCII.GetBytes(secretKey);
    var tokenDescriptor = new SecurityTokenDescriptor
    {
        Subject = new ClaimsIdentity(new Claim[]
        {
            new Claim(ClaimTypes.Email, user.Email),
            new Claim(ClaimTypes.Role, user.UserType),
        }),
        Expires = DateTime.UtcNow.AddDays(7),
        SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(key), SecurityAlgorithms.HmacSha256Signature)
    };

    var token = tokenHandler.CreateToken(tokenDescriptor);
    LoginResponseDTO loginResponseDTO = new()
    {
        User = _mapper.Map<User>(user),
        Token = new JwtSecurityTokenHandler().WriteToken(token)
    };
    loginResponseDTO.User.PasswordHash = null;
    return loginResponseDTO;
}

```

jwt için AuthRepository'si oluşturulmuştur. Bu repository User objesinin rolüne göre yetkilendirme sağlayıp jwt döner. Bu repository AuthController ile çağrılır.

```

4 references
public interface IAuthRepository
{
    2 references
    Task<LoginResponseDTO> Login(LoginRequestDTO loginRequestDTO);
}

```

```

1 reference
[Route("api/login")]
public class AuthController : Controller
{
    private readonly IAuthRepository _authRepository;

    0 references
    public AuthController(IAuthRepository authRepository)
    {
        _authRepository = authRepository;
    }

    [HttpPost]
    [HttpOptions]
    [DisableCors]
    0 references
    public async Task<ActionResult> Login([FromBody] LoginRequestDTO model)
    {
        var loginResponse = await _authRepository.Login(model);
        if (loginResponse == null)
        {
            return BadRequest();
        }
        return Ok(loginResponse);
    }
}

```

Bu repository için Request ve

Response olarak iki adet DTO oluşturulmuştur. Bu objeler aracılığı ile e-mail pass alınıp, jwt döner.

```
namespace StudentManagementSystemAPI.Models.DTO
{
    4 references
    public class LoginRequestDTO
    {
        1 reference
        public string Email { get; set; }
        1 reference
        public string Password { get; set; }
    }
}
```

```
namespace StudentManagementSystemAPI.Models.DTO
{
    5 references
    public class LoginResponseDTO
    {
        2 references
        public User User { get; set; }

        1 reference
        public string Token { get; set; }
    }
}
```

Api endpointlerindeki en önemli sorgu kısmı grade endpointidir.

```
2 references
public async Task<GradeResponseDTO> GetAsync(int id)
{
    Grade grade = new Grade();
    grade = await _context.Grades.FindAsync(id);
    List<User> query = _context.Users.ToList();
    GradeResponseDTO gradeResponseDTO = new GradeResponseDTO()
    {
        Grade = grade,
        Lecturer = query.SingleOrDefault(g => g.UserId == grade.LecturerId),
        Student = query.SingleOrDefault(g => g.UserId == grade.StudentId)
    };
    gradeResponseDTO.Student.PasswordHash = null;
    gradeResponseDTO.Lecturer.PasswordHash = null;
    return gradeResponseDTO;
}

2 references
public List<GradeResponseDTO> GradeListToDTO(List<Grade> grades)
{
    List<User> query = _context.Users.ToList();
    List<GradeResponseDTO> gradeList = new List<GradeResponseDTO>();
    foreach (var grade in grades) {
        GradeResponseDTO gradeResponseDTO = new GradeResponseDTO()
        {
            Grade = grade,
            Lecturer = query.SingleOrDefault(g => g.UserId == grade.LecturerId),
            Student = query.SingleOrDefault(g => g.UserId == grade.StudentId)
        };
        gradeResponseDTO.Student.PasswordHash = null;
        gradeResponseDTO.Lecturer.PasswordHash = null;
        gradeList.Add(gradeResponseDTO);
    }
    return gradeList;
}
```

```

2 references
public List<GradeResponseDTO> GetByStudentId(int id)
{
    List<Grade> query = _context.Grades.ToList();

    List<Grade> grades = query.Where(g => g.StudentId == id).ToList();

    List<GradeResponseDTO> gradeList = new List<GradeResponseDTO>();

    gradeList = GradeListToDTO(grades);

    return (gradeList);
}

2 references
public List<GradeResponseDTO> GetByLecturerId(int id)
{
    List<Grade> query = _context.Grades.ToList();

    List<Grade> grades = query.Where(g => g.LecturerId == id).ToList();

    List<GradeResponseDTO> gradeList = new List<GradeResponseDTO>();

    gradeList = GradeListToDTO(grades);

    return (gradeList);
}

```

Buraya sorgu atılırken kullanıcı tipine göre kendi id'si ile sorgu atar. Ona göre

```

namespace StudentManagementSystemAPI.Models.DTO
{
    17 references
    public class GradeResponseDTO
    {
        2 references
        public Grade Grade { get; set; }

        4 references
        public User Lecturer { get; set; }

        4 references
        public User Student { get; set; }
    }
}

```

yine bir DTO

döner.

```

SqlServerModelBuilderExtensions.UseIdentityColumns(modelBuilder);

modelBuilder.Entity("StudentManagementSystemAPI.Models.Grade", b =>
{
    b.Property<int>("GradeId")
        .ValueGeneratedOnAdd()
        .HasColumnType("int");

    SqlServerPropertyBuilderExtensions.UseIdentityColumn(b.Property<int>("GradeId"));

    b.Property<int>("LecturerId")
        .HasColumnType("int");

    b.Property<string>("LessonDate")
        .HasColumnType("nvarchar(max)");

    b.Property<string>("LessonName")
        .HasColumnType("nvarchar(max)");

    b.Property<int>("Score")
        .HasColumnType("int");

    b.Property<int>("StudentId")
        .HasColumnType("int");

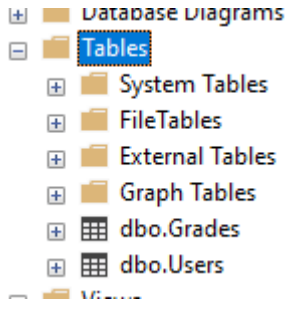
    b.HasKey("GradeId");

    b.ToTable("Grades");

    b.HasData(

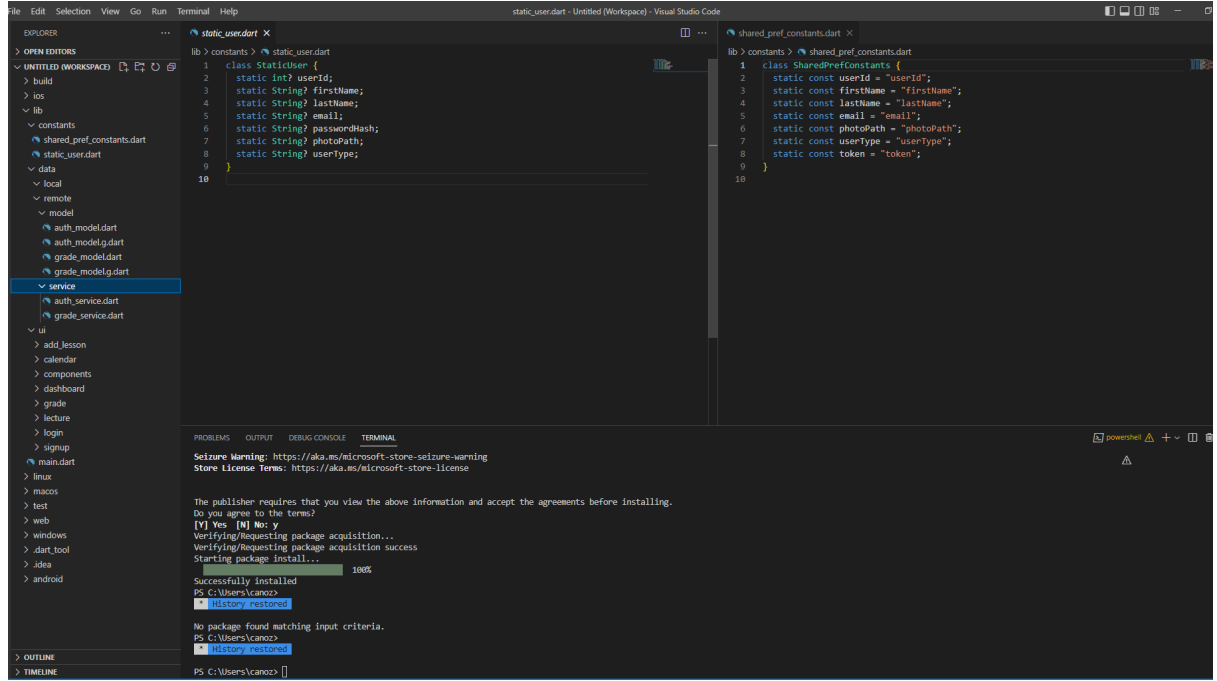
```

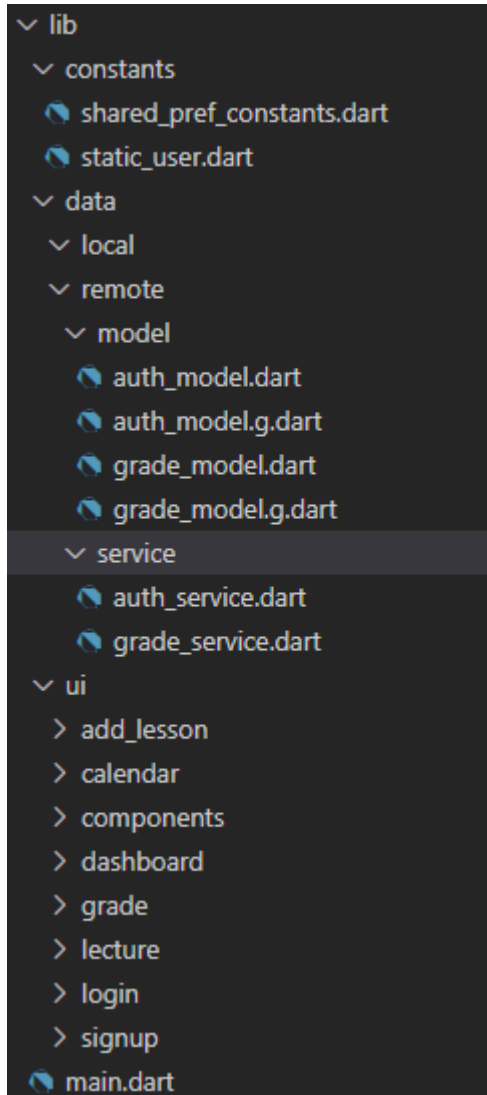
Database oluşturulurken Code-First kullanıldığı için migration yapılmıştır. Bu sayede



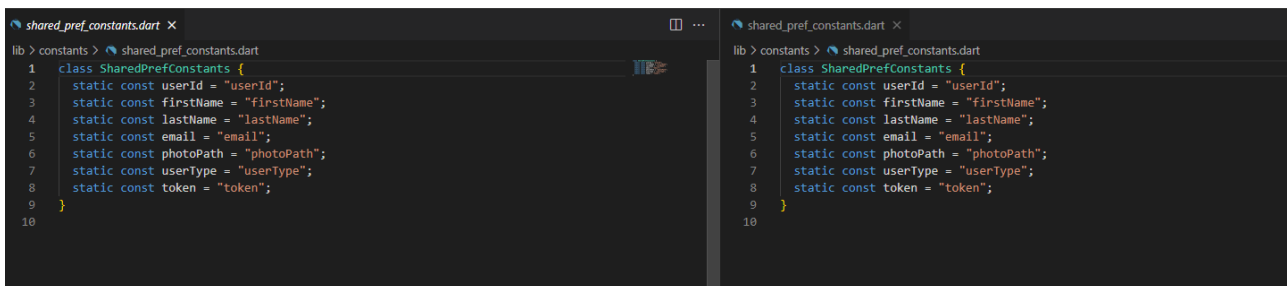
Tabloları oluşturulmuştur.

Web Uygulamasının Front Kısımında flutter kullanılmıştır.





API'den veri çekmek için servisler yazılmıştır. Bunları modellere mapleyip ui kısmında gösterilmiştir.



Giriş yapıldıktan sonra gelen user sessionu shared_pref ve static_user olarak tutulmuştur.

```
auth_service.dart X
lib > data > remote > service > auth_service.dart
1 import 'package:dio/dio.dart';
2 import 'package:student_management_system_app/data/remote/model/auth_model.dart';
3
4 class AuthService {
5   static AuthService service = AuthService._();
6
7   AuthService._();
8
9   final String BASE_URL = "http://4665-88-244-38-68.ngrok.io/api/";
10  final String LOGIN = "login";
11
12  Dio dio = Dio();
13
14  Future<AuthModel> login(Map<String, String> loginModel) async {
15    final url = "$BASE_URL$LOGIN";
16
17    try {
18      Response response = await dio.post(url, data: loginModel);
19      AuthModel model = AuthModel.fromJson(response.data);
20      return model;
21    } on DioError catch (e) {}
22    print(e);
23  }
24
25  return null;
26
27 }
28

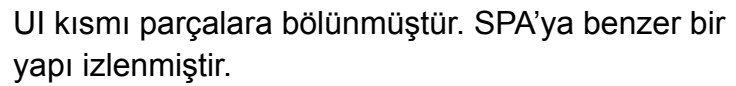
grade_service.dart
lib > data > remote > service > grade_service.dart
1 import 'package:dio/dio.dart';
2 import 'package:student_management_system_app/constants/static_user.dart';
3 import 'package:student_management_system_app/data/remote/model/grade_model.dart';
4
5 class GradeService {
6   static GradeService service = GradeService._();
7
8   GradeService._();
9
10  Dio dio = Dio();
11
12  final String BASE_URL = "http://4665-88-244-38-68.ngrok.io/api/";
13  final String GRADE = "grade";
14  final String USER = "user";
15
16  Future<List<GradeModel>> fetchAllGrades() async {
17    var url = "$BASE_URL$GRADE";
18    var id = StaticUser.userId == "student" ? "studentId" : "lecturerId";
19
20    try {
21      Response response = await dio.get(url, queryParameters: {id: StaticUser.userId});
22      List<GradeModel> grades = (response.data as List).map((e) => GradeModel.fromJson(e));
23      return grades;
24    } on DioError catch (e) {
25      print(e);
26      return List.empty();
27    }
28  }
29 }
```

Servisler şekildeki gibidir. Static base url olarak dışarıda test yapma amaçlı ngrok ile tünel açılmıştır. Kullanımı free version olduğu için uygulama tekrar çalıştığında endpoint değişecektir.

```
grade_model.dart X
lib > data > remote > model > grade_model.dart
1 // ignore_for_file: must_be_immutable
2
3 import 'package:equatable/equatable.dart';
4
5 part 'grade_model.g.dart';
6
7 class GradeModel extends Equatable {
8   Grade? grade;
9   Lecturer? lecturer;
10  Lecturer? student;
11
12  GradeModel(this.grade, this.lecturer, this.student);
13
14  GradeModel.fromJson(Map<String, dynamic> json) {
15    grade = json['grade'] != null ? Grade.fromJson(json['grade']) : null;
16    lecturer = json['lecturer'] != null ? Lecturer.fromJson(json['lecturer']) : null;
17    student = json['student'] != null ? Lecturer.fromJson(json['student']) : null;
18  }
19
20  @override
21  List<Object>? get props => [grade, lecturer, student];
22
23 }
24
25 class Grade extends Equatable {
26   int? gradeId;
27   String? lessonName;
28   int? lecturerId;
29   int? studentId;
30   int? score;
31 }
32

auth_model.dart X
lib > data > remote > model > auth_model.dart
1 import 'package:equatable/equatable.dart';
2 import 'package:json_annotation/json_annotation.dart';
3
4 part 'auth_model.g.dart';
5
6 @JsonSerializable(createToJson: false)
7 class AuthModel extends Equatable {
8   User? user;
9   String? token;
10
11  AuthModel(this.user, this.token);
12
13  factory AuthModel.fromJson(Map<String, dynamic> json) {
14    return _AuthModelFromJson(json);
15  }
16
17  @override
18  List<Object>? get props => [user, token];
19
20 }
21
22 @JsonSerializable(createToJson: false)
23 class User extends Equatable {
24   int? userId;
25   String? firstName;
26   String? lastName;
27   String? email;
28   String? passwordHash;
29   String? photoPath;
30   String? userType;
31 }
```

API'den dönen jsonlar modellere parse edilmiştir.



```
lib > ui > add_lesson.dart
80 mainAxisAlignment: MainAxisAlignment.spaceBetween,
81 crossAxisAlignment: CrossAxisAlignment.start,
82 children: [
83   Expanded(
84     child: Text(
85       model.grade?.lessonName ?? "",
86       style: Theme.of(context).textTheme.labelLarge,
87     ),
88   ),
89   Expanded(
90     child: ListTile(
91       title: Text(
92         "${model.student?.firstName} ${model.student?.lastName}",
93         style: Theme.of(context).textTheme.labelLarge,
94       ),
95       subtitle: Text(
96         "${model.student?.email}",
97         style: Theme.of(context).textTheme.labelMedium,
98       ),
99     ),
100   ),
101   Expanded(
102     child: TextField(controller: controller),
103   ),
104 ],
105 ),
106 ],
107 ),
108 }

lib > ui > calendar.dart
41 buildLesson(model: model, grade: grade, elevation: 10, title:
42   ),
43   Expanded(
44     child: SizedBox(
45       height: 70,
46       child: Card(color: Colors.grey[600], elevation: 10, child:
47   ),
48   Expanded(
49     child: SizedBox(
50       height: 70,
51       child: Card(color: Colors.grey[600], elevation: 10, child:
52   ),
53   Expanded(
54     child: SizedBox(
55       height: 70,
56       child: Card(color: Colors.grey[600], elevation: 10, child:
57   ),
58   Expanded(
59     child: SizedBox(
60       height: 70,
61       child: Card(color: Colors.grey[600], elevation: 10, child:
62   ),
63   ),
64   Row(
65     children: [
66       Expanded(
67         child: SizedBox(height: 400, child: buildLesson(model, "
68   ),
69       ),
70     ],
71   ),
72   ),
73   ),
74   ),
75   ),
76   ),
77   ),
78   ),
79   ),
80   ),
81   ),
82   ),
83   ),
84   ),
85   ),
86   ),
87   ),
88   ),
89   ),
90   ),
91   ),
92   ),
93   ),
94   ),
95   ),
96   ),
97   ),
98   ),
99   ),
100   ),
101   ),
102   ),
103   ),
104   ),
105   ),
106   ),
107   ),
108   ),
109   ),
110   ),
111   ),
112   ),
113   ),
114   ),
115   ),
116   ),
117   ),
118   ),
119   ),
120   ),
121   ),
122   ),
123   ),
124   ),
125   ),
126   ),
127   ),
128   ),
129   ),
130   ),
131   ),
132   ),
133   ),
134   ),
135   ),
136   ),
137   ),
138   ),
139   ),
140   ),
141   ),
142   ),
143   ),
144   ),
145   ),
146   ),
147   ),
148   ),
149   ),
150   ),
151   ),
152   ),
153   ),
154   ),
155   ),
156   ),
157   ),
158   ),
159   ),
160   ),
161   ),
162   ),
163   ),
164   ),
165   ),
166   ),
167   ),
168   ),
169   ),
170   ),
171   ),
172   ),
173   ),
174   ),
175   ),
176   ),
177   ),
178   ),
179   ),
180   ),
181   ),
182   ),
183   ),
184   ),
185   ),
186   ),
187   ),
188   ),
189   ),
190   ),
191   ),
192   ),
193   ),
194   ),
195   ),
196   ),
197   ),
198   ),
199   ),
200   ),
201   ),
202   ),
203   ),
204   ),
205   ),
206   ),
207   ),
208   ),
209   ),
210   ),
211   ),
212   ),
213   ),
214   ),
215   ),
216   ),
217   ),
218   ),
219   ),
220   ),
221   ),
222   ),
223   ),
224   ),
225   ),
226   ),
227   ),
228   ),
229   ),
230   ),
231   ),
232   ),
233   ),
234   ),
235   ),
236   ),
237   ),
238   ),
239   ),
240   ),
241   ),
242   ),
243   ),
244   ),
245   ),
246   ),
247   ),
248   ),
249   ),
250   ),
251   ),
252   ),
253   ),
254   ),
255   ),
256   ),
257   ),
258   ),
259   ),
260   ),
261   ),
262   ),
263   ),
264   ),
265   ),
266   ),
267   ),
268   ),
269   ),
270   ),
271   ),
272   ),
273   ),
274   ),
275   ),
276   ),
277   ),
278   ),
279   ),
280   ),
281   ),
282   ),
283   ),
284   ),
285   ),
286   ),
287   ),
288   ),
289   ),
290   ),
291   ),
292   ),
293   ),
294   ),
295   ),
296   ),
297   ),
298   ),
299   ),
300   ),
301   ),
302   ),
303   ),
304   ),
305   ),
306   ),
307   ),
308   ),
309   ),
310   ),
311   ),
312   ),
313   ),
314   ),
315   ),
316   ),
317   ),
318   ),
319   ),
320   ),
321   ),
322   ),
323   ),
324   ),
325   ),
326   ),
327   ),
328   ),
329   ),
330   ),
331   ),
332   ),
333   ),
334   ),
335   ),
336   ),
337   ),
338   ),
339   ),
340   ),
341   ),
342   ),
343   ),
344   ),
345   ),
346   ),
347   ),
348   ),
349   ),
350   ),
351   ),
352   ),
353   ),
354   ),
355   ),
356   ),
357   ),
358   ),
359   ),
360   ),
361   ),
362   ),
363   ),
364   ),
365   ),
366   ),
367   ),
368   ),
369   ),
370   ),
371   ),
372   ),
373   ),
374   ),
375   ),
376   ),
377   ),
378   ),
379   ),
380   ),
381   ),
382   ),
383   ),
384   ),
385   ),
386   ),
387   ),
388   ),
389   ),
390   ),
391   ),
392   ),
393   ),
394   ),
395   ),
396   ),
397   ),
398   ),
399   ),
400   ),
401   ),
402   ),
403   ),
404   ),
405   ),
406   ),
407   ),
408   ),
409   ),
410   ),
411   ),
412   ),
413   ),
414   ),
415   ),
416   ),
417   ),
418   ),
419   ),
420   ),
421   ),
422   ),
423   ),
424   ),
425   ),
426   ),
427   ),
428   ),
429   ),
430   ),
431   ),
432   ),
433   ),
434   ),
435   ),
436   ),
437   ),
438   ),
439   ),
440   ),
441   ),
442   ),
443   ),
444   ),
445   ),
446   ),
447   ),
448   ),
449   ),
450   ),
451   ),
452   ),
453   ),
454   ),
455   ),
456   ),
457   ),
458   ),
459   ),
460   ),
461   ),
462   ),
463   ),
464   ),
465   ),
466   ),
467   ),
468   ),
469   ),
470   ),
471   ),
472   ),
473   ),
474   ),
475   ),
476   ),
477   ),
478   ),
479   ),
480   ),
481   ),
482   ),
483   ),
484   ),
485   ),
486   ),
487   ),
488   ),
489   ),
490   ),
491   ),
492   ),
493   ),
494   ),
495   ),
496   ),
497   ),
498   ),
499   ),
500   ),
501   ),
502   ),
503   ),
504   ),
505   ),
506   ),
507   ),
508   ),
509   ),
510   ),
511   ),
512   ),
513   ),
514   ),
515   ),
516   ),
517   ),
518   ),
519   ),
520   ),
521   ),
522   ),
523   ),
524   ),
525   ),
526   ),
527   ),
528   ),
529   ),
530   ),
531   ),
532   ),
533   ),
534   ),
535   ),
536   ),
537   ),
538   ),
539   ),
540   ),
541   ),
542   ),
543   ),
544   ),
545   ),
546   ),
547   ),
548   ),
549   ),
550   ),
551   ),
552   ),
553   ),
554   ),
555   ),
556   ),
557   ),
558   ),
559   ),
560   ),
561   ),
562   ),
563   ),
564   ),
565   ),
566   ),
567   ),
568   ),
569   ),
570   ),
571   ),
572   ),
573   ),
574   ),
575   ),
576   ),
577   ),
578   ),
579   ),
580   ),
581   ),
582   ),
583   ),
584   ),
585   ),
586   ),
587   ),
588   ),
589   ),
590   ),
591   ),
592   ),
593   ),
594   ),
595   ),
596   ),
597   ),
598   ),
599   ),
600   ),
601   ),
602   ),
603   ),
604   ),
605   ),
606   ),
607   ),
608   ),
609   ),
610   ),
611   ),
612   ),
613   ),
614   ),
615   ),
616   ),
617   ),
618   ),
619   ),
620   ),
621   ),
622   ),
623   ),
624   ),
625   ),
626   ),
627   ),
628   ),
629   ),
630   ),
631   ),
632   ),
633   ),
634   ),
635   ),
636   ),
637   ),
638   ),
639   ),
640   ),
641   ),
642   ),
643   ),
644   ),
645   ),
646   ),
647   ),
648   ),
649   ),
650   ),
651   ),
652   ),
653   ),
654   ),
655   ),
656   ),
657   ),
658   ),
659   ),
660   ),
661   ),

```

```
lib > ui > dashboard > dashboard_page.dart
122
123
124
125 Visibility(
126   child: ListTile(
127     title: const Text('Ders Ekle'),
128     onTap: () {
129       setState(() {
130         currentScreen = const AddLessonPage();
131       });
132     },
133   ),
134   ListTile(
135     title: const Text('Öğretim Üyeleri'),
136     onTap: () {
137       setState(() {
138         currentScreen = const LecturerPage();
139       });
140     },
141   ),
142   ListTile(
143     title: const Text('Ders Programı'),
144     onTap: () {
145       setState(() {
146         currentScreen = const CalendarPage();
147       });
148     },
149   ),
150 ),

lib > ui > grade > grade_page.dart
62
63
64 Widget buildListItem(GradeModel model) {
65   return ListTile(
66     trailing: Text(model.grade?.score.toString() ?? "", style: Theme.of(context)
67   leading: Container(
68     width: 50,
69     height: 50,
70     decoration: BoxDecoration(
71       color: Theme.of(context).colorScheme.secondary,
72       borderRadius: BorderRadius.circular(10),
73     ),
74     child: Icon(
75       Icons.cast_for_education_outlined,
76       color: Colors.white,
77     ),
78   ),
79   title: Text(model.grade?.lessonName ?? "", style: Theme.of(context).textTheme
80 },
81 },
82 }
```

```
lib > ui > lecturer > lecturer_page.dart
67
68 width: 500,
69 child: Text(
70   model.grade?.lessonName ?? "",
71   style: Theme.of(context).textTheme.displaySmall,
72 ),
73 ),
74 SizedBox(
75   height: 100,
76   width: 500,
77   child: ListTile(
78     title: Text(
79       "İsim: ${model.lecturer?.firstName} ${model.lecturer?.lastName}",
80       style: Theme.of(context).textTheme.displaySmall,
81     ),
82     subtitle: Text(
83       "Mail: ${model.lecturer?.email}",
84       style: Theme.of(context).textTheme.displaySmall,
85     ),
86   ),
87 ),
88 ),
89 },
90 },
91 },

lib > ui > login > login_page.dart
69
70 child: Column(
71   children: [
72     SizedBox(height: 50),
73     Image.network(
74       "https://upload.wikimedia.org/wikipedia/tr/5/5f/Ankara_University_Logo.png",
75       height: 200,
76       width: 200,
77     ),
78     SizedBox(height: 50),
79     Text(
80       "https://github.com/canoztas/INSTUDENT_MANAGEMENT_SYSTEM",
81       style: TextStyle(color: Colors.grey[700], fontSize: 16),
82     ),
83     MyTextField(controller: userNameController, hintText: "User name",
84     MyTextField(controller: passwordController, hintText: "Password",
85     MyButton(onTap: () {
86       signIn(context);
87     },
88     SizedBox(height: 50),
89     Divider()
90   ],
91 ),
92 ),
93 ),
94 ),
95 ),
96 },
97 }
```