**MESLEK YÜKSEK OKULU**

**2024 – 2025 EĞİTİM – ÖĞRETİM YILI** / 2023-2024 ACADEMIC YEAR

**GÜZ DÖNEMİ** / SEMESTER

**Öğrencinin** / *Student’s*

**Adı Soyadı** / *Name, Surname* :Erhan Dağhan

**Numarası** / *Number* :202207012005

**Bölüm - Program** / *Department-Programme* :Bilgisayar Programcılığı

**İmzası** / *Signature* :

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **DERS**  *Course* | **Adı** */ Name* | : Mobil Uygulamalar |  | |  |  |  |  |  | **SINAV**  *Exam* |  | **Tarih** */ Date :* 16.01.2025 |
| **Kodu** */ Code* | *:* BLP 273 |  | |  |  |  |  |  | **Süresi** */ Duration : 9*0 DK. |
| **Sorumlusu** */ Lecturer* | *:* Öğr. Gör. Tayfun SOY | | | | | | | | **Türü** */ Type :* Final |
| **Soru Numarası** */ Numbers of the Questions* | | | 1 |  | | | | | | | | |
| **Alınan Not** */ Scored Points* | | |  |
| **SINAV KURALLARI** */ Exam Rules : sınav proje olarak teslim edilecektir. Kırmızı ile belirlenen kuraları(notları) dikkate alarak projenizi oluşturunuz.* | | | | | | | | | | | | |

*\*Her sorunun puan değeri rakamsal olarak yanına belirtilmelidir. / The points for each question must be stated next to the question.*

**SORULAR** / *Questions*

**FİNAL PROJESİ**

Soru-) Aşağıda verilen konu başlıkları dikkate alınarak bir Android Studio kotlin uygulaması oluşturun. Oluşturulan projenin bütünlüğüne (tasarım, işleyiş ve kodlama) dikkat ediniz. Projenizde aşağıda belirtilen fonksiyonlardan farklı bir fonksiyon veya yapı kullanmak istediğinizde açıklama satırları ile kodlamanın çalışma mantığını anlatınız. Proje kapsamında bulunması gereken işlemler aşağıda sıralanmıştır:

* AKTİVİTELER ARASI GEÇİŞ
* KULLANICI GİRİŞ VE KAYIT İŞLEMLERİ
* RECYCLER VİEW İLE LİSTELEME İŞLEMLERİ
* SQLİTE İŞLEMLERİNİN KODLANMASI(Tercihen uzak sunucu kullanılabilir.)
* VERİ EKLEME
* VERİ LİSTELEME
* VERİ SİLME
* VERİ GÜNCELLEME
* BUTTON TIKLAMA OLAYLARININ GERÇEKLEŞTİRİLMESİ,
* TOAST MESAJININ KULLANIMI.

**Not: Her öğrencinin kendine özgü projesi olmalıdır. Özgün olmayan projeler kabul edilmeyecek olup notu 0 olarak girilecektir.**

**Not: Yukarıda belirtilen işlemler uygulama içeriği ile uyumlu şekilde kullanılmalıdır.**

**Not: Yapay zeka ile oluşturulan özgün olmayan kodlamadan puan kırılacaktır.**

* **Uygulama, tanıtımı ve işleyişi**

Uygulamada 4 aktivite 4 fragment bulunmaktadır. Bu aktivite ve fragmentler uygulama içinde kayıt olma, giriş yapma, diğer kullanıcılar ile mesajlaşma, profil düzenleme ve herkesin görebileceği mesajlar yayınlamak için kullanılıyor. Bu aktivite ve fragmentlerin içerisinde imageview, recyclerview, button, imagebutton, bottom navigation, edittext ve textview bulunmakta. Bu objeler ile birlikte kullanıcılar kendi aralarında ve ya herkese açık bir şekilde yazı paylaşabilmekte.

MainActivity'de giriş yapma ekranı bulunmakta.

MainActivity2'de kayıt olabileceğiniz yer mevcut.

MainActivity4'de herkese açık yazıları görebileceğiniz, diğer kullanıcıları aratıp onlarla iletişime geçebileceğiniz, kendi profilinizi düzenleyebileceğiniz ve herkese açık yazı paylaşabileceğiniz fragmentler bulunmakta. Bu fragmentler sırasıyla:HomeFragment, AramaFragment, EditProfileFragment, EklemeFragment.

MainActivity5'de diğer kullanıcılar ile özel olarak mesajlaşabilirsiniz.

SplashScreen'de uygulamanın açılış ekranında ki animasyon mevcut.

**Uygulamanın videosu:** [**https://youtu.be/AGtask1ynZg**](https://youtu.be/AGtask1ynZg)

* **Tasarım**

Tasarım başlığı adı altında, uygulamada kullanılacak objeler ve işlevleri açıklanmalıdır.

**Örneğin**

Uygulamada kullanılacak objeler ve Activiteler

* RecyclerView (Listeleme için kullanılmıştır.)
* İmageView(Uygulama logosu ve tuş takımı butonları için kullanıldı.)
* Button(Tıklama olayları için kullanılmıştır.)
* İmageButton(Görsel butonlar için kullanıldı.)
* TextView(Veritabanında ki yazıların görüntülenmesi için kullanıldı)
* EditText(Veri ekleme ve güncelleme için kullanıldı.)

**3-Kodlama**

Bu başlık altında uygulamayı oluşturan kodlamanın tamamı eksiksiz olarak eklenmelidir. Not: kodlamanın yazılı olduğu dosyada CTRL+A(tümünü seç) yapılarak bu alana kopyalanmalıdır. Aynı işlemler tüm kodlaması yapılan dosyalar için yapılmalıdır.

**Örneğin**

**“SplashScreen.kt”**

package com.mychat.messageapp  
  
import android.annotation.SuppressLint  
import android.content.Intent  
import android.os.Bundle  
import android.os.Handler  
import androidx.activity.enableEdgeToEdge  
import androidx.appcompat.app.AppCompatActivity  
import com.mychat.messageapp.databinding.ActivitySplashScreenBinding  
  
@SuppressLint("CustomSplashScreen")  
@Suppress("DEPRECATION")  
class SplashScreen : AppCompatActivity() {  
 private val splashScreen = 3000  
 private lateinit var binding: ActivitySplashScreenBinding // Düzgün tanımlama  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 binding = ActivitySplashScreenBinding.inflate(*layoutInflater*) // Burada `binding`i doğru şekilde kullanıyoruz  
 *enableEdgeToEdge*()  
 setContentView(binding.*root*)  
 *supportActionBar*?.hide()  
  
 val animasyon1 = android.view.animation.AnimationUtils.loadAnimation(this, R.anim.*animasyon1*)  
 val imageView = binding.imageView  
 imageView.*animation* = animasyon1  
  
 Handler().postDelayed(**{** val intent = Intent(this, MainActivity::class.*java*)  
 startActivity(intent)  
 finish()  
 **}**, splashScreen.toLong())  
 }  
}

**“MainActivity.kt”**

package com.mychat.messageapp  
  
import android.annotation.SuppressLint  
import android.content.Intent  
import android.os.Bundle  
import android.text.method.HideReturnsTransformationMethod  
import android.text.method.PasswordTransformationMethod  
import android.view.MotionEvent  
import android.view.View  
import android.widget.Toast  
import androidx.activity.enableEdgeToEdge  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.Firebase  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.auth.auth  
import com.google.firebase.firestore.FirebaseFirestore  
import com.mychat.messageapp.databinding.ActivityMainBinding  
  
class MainActivity : AppCompatActivity() {  
  
 private lateinit var auth: FirebaseAuth  
 lateinit var binding: ActivityMainBinding  
 @SuppressLint("ClickableViewAccessibility")  
 override fun onCreate(savedInstanceState: Bundle?) {  
 binding=ActivityMainBinding.inflate(*layoutInflater*)  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 setContentView(binding.*root*)  
 auth = Firebase.*auth* //val etPass = findViewById<EditText>(R.id.etpass)  
 val etPass = binding.etpass  
  
// Başlangıç durumu  
 var isPasswordVisible = false  
  
 etPass.setOnTouchListener **{** \_, event **->** if (event.*action* == MotionEvent.*ACTION\_UP*) {  
 // Soldaki drawable'a dokunulup dokunulmadığını kontrol et  
 if (event.*rawX* <= (etPass.*left* + etPass.*compoundDrawables*[0].*bounds*.width())) {  
 isPasswordVisible = !isPasswordVisible  
 if (isPasswordVisible) {  
 // Şifreyi göster  
 etPass.*transformationMethod* = HideReturnsTransformationMethod.getInstance()  
 etPass.setCompoundDrawablesWithIntrinsicBounds(  
 R.drawable.*lockopen*, 0, 0, 0  
 )  
 } else {  
 // Şifreyi gizle  
 etPass.*transformationMethod* = PasswordTransformationMethod.getInstance()  
 etPass.setCompoundDrawablesWithIntrinsicBounds(  
 R.drawable.*lockofe*, 0, 0, 0  
 )  
 }  
 etPass.setSelection(etPass.*text*.length) // İmleci metnin sonuna taşı  
 true  
 } else {  
 false  
 }  
 } else {  
 false  
 }  
 **}** val kullanici=auth.*currentUser* if (kullanici!=null){  
 intent(MainActivity4::class.*java*)  
 }  
 }  
 fun kayitOl(view: View){  
 intent(MainActivity2::class.*java*)  
 }  
 fun girisYap(view: View) {  
 val mail = binding.etmail.*text*.toString()  
 val pass = binding.etpass.*text*.toString()  
  
 if (mail.*isNotEmpty*() && pass.*isNotEmpty*()) {  
 auth.signInWithEmailAndPassword(mail, pass).addOnCompleteListener **{** task **->** if (task.*isSuccessful*) {  
 val kullanici = auth.*currentUser* val firestore = FirebaseFirestore.getInstance()  
  
 // Firestore'dan kullanıcı bilgilerini al  
 firestore.collection("Kullanicilar")  
 .document(kullanici!!.*uid*) // Kullanıcının UID'si ile belgeyi bul  
 .get()  
 .addOnSuccessListener **{** document **->** if (document != null && document.exists()) {  
 val kullaniciAdi = document.getString("kullaniciAdi") ?: "Bilinmiyor"  
 Toast.makeText(  
 *applicationContext*,  
 "Hoşgeldin: $kullaniciAdi",  
 Toast.*LENGTH\_SHORT* ).show()  
  
 // Başarılı giriş sonrası MainActivity4'e yönlendirme  
 //val intent = Intent(this, MainActivity4::class.java)  
 //startActivity(intent)  
 //finish()  
 intent(MainActivity4::class.*java*)  
 } else {  
 Toast.makeText(  
 *applicationContext*,  
 "Kullanıcı bilgileri bulunamadı.",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 **}** .addOnFailureListener **{** exception **->** Toast.makeText(  
 *applicationContext*,  
 "Bilgiler alınırken hata: ${exception.*localizedMessage*}",  
 Toast.*LENGTH\_LONG* ).show()  
 **}** }  
 **}**.addOnFailureListener **{** exception **->** Toast.makeText(*applicationContext*, exception.*localizedMessage*, Toast.*LENGTH\_SHORT*)  
 .show()  
 **}** } else {  
 Toast.makeText(*applicationContext*, "Lütfen Boş Alanları Doldurunuz.", Toast.*LENGTH\_SHORT*).show()  
 }  
 }  
  
 fun intent(targetActivity: Class<\*>) {  
 val intent = Intent(this, targetActivity)  
 startActivity(intent)  
 finish()  
 }  
  
  
}

**“MainActivity2.kt”**

package com.mychat.messageapp  
  
import android.content.Intent  
import android.os.Bundle  
import android.view.View  
import android.widget.Toast  
import androidx.activity.enableEdgeToEdge  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.auth.ktx.auth  
import com.google.firebase.auth.userProfileChangeRequest  
import com.google.firebase.firestore.FirebaseFirestore  
import com.google.firebase.ktx.Firebase  
import com.mychat.messageapp.databinding.ActivityMain2Binding  
  
class MainActivity2 : AppCompatActivity() {  
 private lateinit var auth: FirebaseAuth  
 lateinit var binding: ActivityMain2Binding  
 override fun onCreate(savedInstanceState: Bundle?) {  
 binding=ActivityMain2Binding.inflate(*layoutInflater*)  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 setContentView(binding.*root*)  
 auth= Firebase.*auth* }  
fun kayitOl(view: View) {  
 val email = binding.etkayitMail.*text*.toString().*trim*()  
 val pass = binding.etkayitPass.*text*.toString().*trim*()  
 val kadi = binding.etkadi.*text*.toString().*trim*()  
  
 if (email.*isNotEmpty*() && pass.*isNotEmpty*() && kadi.*isNotEmpty*()) {  
 val intent = Intent(this, MainActivity4::class.*java*)  
  
 auth.createUserWithEmailAndPassword(email, pass).addOnCompleteListener **{** task **->** if (task.*isSuccessful*) {  
 val kullanici = auth.*currentUser* val profilGuncelleme = *userProfileChangeRequest* **{** *displayName* = kadi  
 **}** kullanici?.updateProfile(profilGuncelleme)?.addOnCompleteListener **{** task **->** if (task.*isSuccessful*) {  
 Toast.makeText(  
 *applicationContext*,  
 "Kullanıcı Adı Eklendi: $kadi",  
 Toast.*LENGTH\_SHORT* ).show()  
 }  
 **}** kullanici?.sendEmailVerification()?.addOnCompleteListener **{** task **->** if (task.*isSuccessful*) {  
 Toast.makeText(  
 *applicationContext*,  
 "Doğrulama Maili Gönderildi",  
 Toast.*LENGTH\_LONG* ).show()  
 }  
 **}** // Firestore'a kullanıcı bilgilerini ekleme  
 val firestore = FirebaseFirestore.getInstance()  
 val userMap = *hashMapOf*(  
 "email" *to* email,  
 "kullaniciAdi" *to* kadi.*lowercase*(),  
 "parola" *to* pass,  
 "isOnline" *to* "Offline",  
 "uid" *to* kullanici?.*uid* )  
  
 firestore.collection("Kullanicilar")  
 .document(kullanici!!.*uid*) // Kullanıcı UID'sini belge adı olarak kullanıyoruz  
 .set(userMap)  
 .addOnSuccessListener **{** Toast.makeText(*applicationContext*, "Kullanıcı Bilgileri Kaydedildi.", Toast.*LENGTH\_SHORT*).show()  
 startActivity(intent)  
 finish()  
 **}** .addOnFailureListener **{** exception **->** Toast.makeText(  
 *applicationContext*,  
 "Bilgiler Kaydedilirken Hata: ${exception.*localizedMessage*}",  
 Toast.*LENGTH\_LONG* ).show()  
 **}** Toast.makeText(*applicationContext*, "Başarıyla Kayıt Oldunuz.", Toast.*LENGTH\_LONG*).show()  
 }  
 **}**.addOnFailureListener **{** exception **->** Toast.makeText(*applicationContext*, exception.*localizedMessage*, Toast.*LENGTH\_LONG*).show()  
 **}** } else {  
 when {  
 email.*isEmpty*() -> Toast.makeText(*applicationContext*, "Lütfen Email Alanını Doldurunuz.", Toast.*LENGTH\_SHORT*).show()  
 pass.*isEmpty*() -> Toast.makeText(*applicationContext*, "Lütfen Parola Alanını Doldurunuz.", Toast.*LENGTH\_SHORT*).show()  
 kadi.*isEmpty*() -> Toast.makeText(*applicationContext*, "Lütfen Kullanıcı Adı Alanını Doldurunuz.", Toast.*LENGTH\_SHORT*).show()  
 }  
 }  
}  
 fun geriDon(view: View){  
 val intent = Intent (this, MainActivity::class.*java*)  
 startActivity(intent)  
 finish()  
 }  
}

**“MainActivity4.kt”**

@file:Suppress("DEPRECATION")  
  
package com.mychat.messageapp  
  
import android.os.Bundle  
import android.util.Log  
import androidx.activity.enableEdgeToEdge  
import androidx.appcompat.app.AppCompatActivity  
import androidx.fragment.app.Fragment  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.auth.FirebaseUser  
import com.google.firebase.firestore.FirebaseFirestore  
import com.mychat.messageapp.Fragment.AramaFragment  
import com.mychat.messageapp.Fragment.EditProfileFragment  
import com.mychat.messageapp.Fragment.EklemeFragment  
import com.mychat.messageapp.Fragment.HomeFragment  
import com.mychat.messageapp.databinding.ActivityMain4Binding  
  
class MainActivity4 : AppCompatActivity() {  
 lateinit var binding: ActivityMain4Binding  
 private lateinit var auth: FirebaseAuth  
 private lateinit var firestore: FirebaseFirestore  
 private lateinit var currentUserId: String  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *enableEdgeToEdge*()  
 binding = ActivityMain4Binding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
  
 // FirebaseAuth ve Firestore başlat  
 auth = FirebaseAuth.getInstance()  
 firestore = FirebaseFirestore.getInstance()  
  
 val currentUser: FirebaseUser? = auth.*currentUser* currentUser?.*let* **{** currentUserId = **it**.*uid* **}** setupBottomNavigation()  
 }  
  
 private fun setupBottomNavigation() {  
 val bottomNavigationView = binding.bottomNav  
 var lastSelectedItemId: Int = R.id.*home* loadFragment(HomeFragment())  
  
 bottomNavigationView.setOnItemSelectedListener **{** item **->** if (item.*itemId* == lastSelectedItemId) return@setOnItemSelectedListener true  
  
 when (item.*itemId*) {  
 R.id.*home* -> {  
 loadFragment(HomeFragment())  
 true  
 }  
 R.id.*arama* -> {  
 loadFragment(AramaFragment())  
 true  
 }  
 R.id.*profil* -> {  
 loadFragment(EditProfileFragment())  
 true  
 }  
 R.id.*ekle* -> {  
 loadFragment(EklemeFragment())  
 true  
 }  
 else -> false  
 }.*also* **{** lastSelectedItemId = item.*itemId* **}  
 }** }  
  
 override fun onResume() {  
 super.onResume()  
 updateUserStatus(true)  
 }  
  
 override fun onPause() {  
 super.onPause()  
 updateUserStatus(false)  
 }  
  
 private fun updateUserStatus(isOnline: Boolean) {  
 val status = if (isOnline) "Online" else "Offline"  
 if (::currentUserId.*isInitialized*) {  
 firestore.collection("Kullanicilar").document(currentUserId)  
 .update("isOnline", status)  
 .addOnSuccessListener **{** Log.d("MainActivity4", "Kullanıcı durumu güncellendi: $status")  
 **}** .addOnFailureListener **{** Log.e("MainActivity4", "Kullanıcı durumu güncellenemedi: ${**it**.message}")  
 **}** }  
 }  
  
 private fun loadFragment(fragment: Fragment) {  
 *supportFragmentManager*.beginTransaction().*apply* **{** replace(R.id.*fragmentContainerView*, fragment)  
 addToBackStack(null)  
 commit()  
 **}** }  
}

**“AramaFragment.kt”**

package com.mychat.messageapp.Fragment  
  
import android.content.Context  
import android.content.Intent  
import android.graphics.Rect  
import android.os.Bundle  
import android.text.Editable  
import android.text.TextWatcher  
import android.util.Log  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.EditText  
import android.widget.ImageButton  
import android.widget.ImageView  
import android.widget.TextView  
import android.widget.Toast  
import androidx.core.content.ContextCompat  
import androidx.fragment.app.Fragment  
import androidx.recyclerview.widget.LinearLayoutManager  
import androidx.recyclerview.widget.RecyclerView  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.auth.FirebaseUser  
import com.google.firebase.database.DataSnapshot  
import com.google.firebase.database.DatabaseError  
import com.google.firebase.database.FirebaseDatabase  
import com.google.firebase.database.ValueEventListener  
import com.google.firebase.firestore.FirebaseFirestore  
import com.mychat.messageapp.MainActivity5  
import com.mychat.messageapp.Model.Kullanici  
import com.mychat.messageapp.R  
class AramaFragment : Fragment() {  
  
 private lateinit var firestore: FirebaseFirestore  
 private lateinit var rv: RecyclerView  
 private lateinit var adapter: KullaniciAdapter  
 private lateinit var aramaBar: EditText  
 private val kullanicilar = *mutableListOf*<Kullanici>()  
  
  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? = inflater.inflate(R.layout.*fragment\_arama*, container, false)  
  
 override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  
 super.onViewCreated(view, savedInstanceState)  
  
 firestore = FirebaseFirestore.getInstance()  
 rv = view.findViewById(R.id.*arama\_rec*)  
 aramaBar = view.findViewById(R.id.*aramabar*)  
  
 rv.*layoutManager* = LinearLayoutManager(*context*)  
 adapter = KullaniciAdapter(requireContext(), kullanicilar)  
 rv.*adapter* = adapter  
  
  
 aramaBar.addTextChangedListener(object : TextWatcher {  
 override fun beforeTextChanged(s: CharSequence?, start: Int, count: Int, after: Int) {}  
  
 override fun onTextChanged(s: CharSequence?, start: Int, before: Int, count: Int) {  
 val query = s.*toString*().*trim*()  
 if (query.*isNotEmpty*()) {  
 fetchKullanicilar(query)  
 } else {  
 kullanicilar.clear()  
 adapter.notifyDataSetChanged()  
 }  
 }  
  
 override fun afterTextChanged(s: Editable?) {}  
 })  
 }  
  
 private fun fetchKullanicilar(query: String) {  
 val normalizedQuery = query.*lowercase*()  
 firestore.collection("Kullanicilar")  
 .orderBy("kullaniciAdi")  
 .startAt(normalizedQuery)  
 .endAt("$normalizedQuery\uf8ff")  
 .get()  
 .addOnSuccessListener **{** documents **->** kullanicilar.clear()  
 for (document in documents) {  
 val displayName = document.getString("kullaniciAdi")  
 val email = document.getString("email")  
 val isOnline = document.getString("isOnline") ?: "Offline"  
 val uid = document.getString("uid") ?: ""  
  
 if (!displayName.*isNullOrEmpty*() && !email.*isNullOrEmpty*()) {  
 kullanicilar.add(Kullanici(email, displayName, isOnline, uid))  
 }  
 }  
 adapter.notifyDataSetChanged()  
 **}** .addOnFailureListener **{** Toast.makeText(*context*, "Kullanıcılar alınamadı", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
}  
  
  
class KullaniciAdapter(  
 private val mContext: Context,  
 private var Kullanicilar: MutableList<Kullanici>  
) : RecyclerView.Adapter<KullaniciAdapter.ViewHolder>() {  
  
 private val firebaseUser: FirebaseUser? = FirebaseAuth.getInstance().*currentUser* override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {  
 val view = LayoutInflater.from(mContext).inflate(R.layout.*kullanici\_ogesi*, parent, false)  
 return ViewHolder(view)  
 }  
  
 override fun onBindViewHolder(holder: ViewHolder, position: Int) {  
 val kullanici = Kullanicilar[position]  
 holder.kullaniciAd.*text* = kullanici.kadi  
 holder.itemView.findViewById<ImageButton>(R.id.*kmesaj*).setOnClickListener **{** val intent = Intent(mContext, MainActivity5::class.*java*)  
 intent.putExtra("kullaniciAdi", kullanici.kadi) // Kullanıcı adını gönder  
 intent.putExtra("uid", kullanici.uid) // Kullanıcı ID'sini gönder  
 mContext.startActivity(intent)  
 **}** // Kullanıcının online durumunu kontrol et ve UI'yi güncelle  
 updateOnlineStatus(holder, kullanici.isOnline)  
 }  
  
 override fun getItemCount(): Int = Kullanicilar.size  
  
 // Kullanıcının online durumunu kontrol etme fonksiyonu  
 private fun updateOnlineStatus(holder: ViewHolder, status: String?) {  
 when (status) {  
 "Online" -> {  
 holder.onlineDurum.*text* = "Online"  
 holder.onlineDurum.setTextColor(ContextCompat.getColor(mContext, R.color.*green*)) // Yeşil renk  
 }  
 "Offline" -> {  
 holder.onlineDurum.*text* = "Offline"  
 holder.onlineDurum.setTextColor(ContextCompat.getColor(mContext, R.color.*red*)) // Kırmızı renk  
 }  
 else -> {  
 holder.onlineDurum.*text* = "Bilinmiyor" // Durum gelmediğinde  
 holder.onlineDurum.setTextColor(ContextCompat.getColor(mContext, R.color.*gri*)) // Gri renk  
 }  
 }  
 }  
  
 inner class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 val profilresim: ImageView = itemView.findViewById(R.id.*resimKullanici*)  
 val kullaniciAd: TextView = itemView.findViewById(R.id.*kullaniciAdi*)  
 val onlineDurum: TextView = itemView.findViewById(R.id.*onlineDurum*)  
 }  
  
 // Kullanıcı listesi güncellenirse, RecyclerView'i yenile  
 fun updateKullaniciList(newList: MutableList<Kullanici>) {  
 val filteredList = newList.*filter* **{ it**.uid != firebaseUser?.*uid* **}**.*toMutableList*()  
 Kullanicilar = newList  
 notifyDataSetChanged()  
 }  
  
 // Firebase'den kullanıcı verilerini almak ve durumu güncellemek için ek bir fonksiyon:  
 fun updateFirebaseStatus(userId: String) {  
 val userRef = FirebaseDatabase.getInstance().getReference("Kullanicilar").child(userId)  
 userRef.addValueEventListener(object : ValueEventListener {  
 override fun onDataChange(snapshot: DataSnapshot) {  
 val kullanici = snapshot.getValue(Kullanici::class.*java*)  
 kullanici?.*let* **{** // Eğer kullanıcı verisi alınmışsa, giriş yapan kullanıcıyı hariç tutarak listeyi güncelle  
 if (kullanici.uid != firebaseUser?.*uid*) {  
 val position = Kullanicilar.*indexOfFirst* **{ it**.uid == kullanici.uid **}** if (position != -1) {  
 Kullanicilar[position] = kullanici  
 notifyItemChanged(position)  
 }  
 }  
 **}** }  
  
 override fun onCancelled(error: DatabaseError) {  
 Log.e("KullaniciAdapter", "Veri alınamadı: ${error.*message*}")  
 }  
 })  
 }  
}

**“EditProfileFragment.kt”**

package com.mychat.messageapp.Fragment  
  
import android.content.Intent  
import android.os.Bundle  
import androidx.fragment.app.Fragment  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.firestore.FirebaseFirestore  
import com.mychat.messageapp.MainActivity  
import com.mychat.messageapp.R  
  
  
class EditProfileFragment : Fragment() {  
  
 private lateinit var firestore: FirebaseFirestore  
 private lateinit var auth: FirebaseAuth  
  
 private lateinit var usernameEditText: EditText  
 private lateinit var emailEditText: EditText  
 private lateinit var saveButton: Button  
 private lateinit var logoutButton: Button  
  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 val view = inflater.inflate(R.layout.*fragment\_edit\_profile*, container, false)  
   
 firestore = FirebaseFirestore.getInstance()  
 auth = FirebaseAuth.getInstance()  
   
 usernameEditText = view.findViewById(R.id.*editTextUsername*)  
 emailEditText = view.findViewById(R.id.*editTextEmail*)  
 saveButton = view.findViewById(R.id.*buttonSave*)  
 logoutButton = view.findViewById(R.id.*buttonLogout*)  
  
 loadUserData()  
  
 saveButton.setOnClickListener **{** saveUserData()  
 **}** logoutButton.setOnClickListener **{** logoutUser()  
 **}** return view  
 }  
 private fun logoutUser() {  
 auth.signOut() // Firebase oturumu kapat  
 Toast.makeText(*context*, "Çıkış yapıldı", Toast.*LENGTH\_SHORT*).show()  
  
 // Giriş ekranına yönlendirme (örnek olarak LoginActivity'ye gidiliyor)  
 val intent = Intent(requireContext(), MainActivity::class.*java*)  
 intent.*flags* = Intent.*FLAG\_ACTIVITY\_NEW\_TASK* or Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* startActivity(intent)  
 }  
  
  
 private fun loadUserData() {  
 val currentUser = auth.*currentUser* currentUser?.*let* **{** val userId = **it**.*uid* firestore.collection("Kullanicilar").document(userId).get()  
 .addOnSuccessListener **{** document **->** if (document != null && document.exists()) {  
 usernameEditText.setText(document.getString("kullaniciAdi"))  
 emailEditText.setText(document.getString("email"))  
 }  
 **}** .addOnFailureListener **{** exception **->** Toast.makeText(*context*, "Kullanıcı bilgileri alınamadı: ${exception.message}", Toast.*LENGTH\_SHORT*).show()  
 **}  
 }** }  
  
 private fun saveUserData() {  
 val currentUser = auth.*currentUser* currentUser?.*let* **{** val userId = **it**.*uid* val updatedData = *hashMapOf*(  
 "uid" *to* userId,  
 "kullaniciAdi" *to* usernameEditText.*text*.toString(),  
 "email" *to* emailEditText.*text*.toString()  
 )  
  
 firestore.collection("Kullanicilar").document(userId).set(updatedData)  
 .addOnSuccessListener **{** Toast.makeText(*context*, "Profil başarıyla güncellendi", Toast.*LENGTH\_SHORT*).show()  
 **}** .addOnFailureListener **{** exception **->** Toast.makeText(*context*, "Profil güncellenemedi: ${exception.message}", Toast.*LENGTH\_SHORT*).show()  
 **}  
 }** }  
}

**“EklemeFragment.kt”**

package com.mychat.messageapp.Fragment  
  
import android.os.Bundle  
import androidx.fragment.app.Fragment  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.firestore.FirebaseFirestore  
import com.mychat.messageapp.R  
  
class EklemeFragment : Fragment() {  
  
 private lateinit var firestore: FirebaseFirestore  
 private lateinit var auth: FirebaseAuth  
  
 private lateinit var titleEditText: EditText  
 private lateinit var contentEditText: EditText  
 private lateinit var shareButton: Button  
  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 val view = inflater.inflate(R.layout.*fragment\_ekleme*, container, false)  
   
 firestore = FirebaseFirestore.getInstance()  
 auth = FirebaseAuth.getInstance()  
   
 titleEditText = view.findViewById(R.id.*editTextTitle*)  
 contentEditText = view.findViewById(R.id.*editTextContent*)  
 shareButton = view.findViewById(R.id.*buttonShare*)  
  
 shareButton.setOnClickListener **{** sharePost()  
 **}** return view  
 }  
  
 private fun sharePost() {  
 val title = titleEditText.*text*.toString()  
 val content = contentEditText.*text*.toString()  
  
 if (title.*isBlank*() || content.*isBlank*()) {  
 Toast.makeText(*context*, "Başlık ve içerik boş bırakılamaz!", Toast.*LENGTH\_SHORT*).show()  
 return  
 }  
  
 val currentUser = auth.*currentUser* currentUser?.*let* **{** val userId = **it**.*uid* val post = *hashMapOf*(  
 "title" *to* title,  
 "content" *to* content,  
 "userId" *to* userId,  
 "timestamp" *to* System.currentTimeMillis(),  
 "kullaniciAdi" *to* currentUser.*displayName* )  
  
 firestore.collection("Yazilar").add(post)  
 .addOnSuccessListener **{** Toast.makeText(*context*, "Yazı başarıyla paylaşıldı!", Toast.*LENGTH\_SHORT*).show()  
 titleEditText.*text*.clear()  
 contentEditText.*text*.clear()  
 **}** .addOnFailureListener **{** exception **->** Toast.makeText(*context*, "Yazı paylaşımı başarısız: ${exception.message}", Toast.*LENGTH\_SHORT*).show()  
 **}  
 }** ?: *run* **{** Toast.makeText(*context*, "Kullanıcı oturumu açık değil!", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
}

**“HomeFragment.kt”**

package com.mychat.messageapp.Fragment  
  
import android.os.Bundle  
import android.util.Log  
import androidx.fragment.app.Fragment  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.Button  
import android.widget.EditText  
import android.widget.TextView  
import android.widget.Toast  
import androidx.recyclerview.widget.LinearLayoutManager  
import androidx.recyclerview.widget.RecyclerView  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.firestore.FirebaseFirestore  
import com.google.firebase.firestore.Query  
import com.google.type.Date  
import com.mychat.messageapp.R  
import java.text.SimpleDateFormat  
import java.util.Locale  
  
class HomeFragment : Fragment() {  
  
 private lateinit var firestore: FirebaseFirestore  
 private lateinit var recyclerView: RecyclerView  
 private lateinit var postsAdapter: PostsAdapter  
 private val postsList = *mutableListOf*<Post>()  
  
 override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 val view = inflater.inflate(R.layout.*fragment\_home*, container, false)  
  
 // Firebase Firestore  
 firestore = FirebaseFirestore.getInstance()  
   
 recyclerView = view.findViewById(R.id.*recyclerViewPosts*)  
 postsAdapter = PostsAdapter(postsList)  
 recyclerView.*layoutManager* = LinearLayoutManager(requireContext())  
 recyclerView.*adapter* = postsAdapter  
  
 loadPosts()  
  
 return view  
 }  
  
 private fun loadPosts() {  
 firestore.collection("Yazilar")  
 .orderBy("timestamp", Query.Direction.*DESCENDING*)  
 .get()  
 .addOnSuccessListener **{** result **->** postsList.clear()  
 val userIds = *mutableSetOf*<String>()  
  
 for (document in result) {  
 val post = document.toObject(Post::class.*java*)  
 postsList.add(post)  
 userIds.add(post.userId) // Benzersiz kullanıcı kimliklerini topluyoruz  
 }  
  
 Log.d("LoadPosts", "UserIds: $userIds") // userIds'yi kontrol edin  
 fetchUserNames(userIds) // Kullanıcı adlarını almak için çağırıyoruz  
 **}** .addOnFailureListener **{** exception **->** Toast.makeText(*context*, "Yazılar yüklenemedi: ${exception.message}", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
  
  
 private fun fetchUserNames(userIds: Set<String>) {  
 val userNamesMap = *mutableMapOf*<String, String>()  
  
 Log.d("FetchUserNames", "UserIds: $userIds") // userIds kümesinin içeriğini kontrol edin  
  
 firestore.collection("Kullanicilar")  
 .whereIn("uid", userIds.*toList*())  
 .get()  
 .addOnSuccessListener **{** result **->** Log.d("FetchUserNames", "Firestore result size: ${result.size()}")  
  
 for (document in result) {  
 val userId = document.getString("uid") ?: ""  
 val userName = document.getString("kullaniciAdi") ?: "Bilinmeyen"  
  
 Log.d("FetchUserNames", "Found user: $userId -> $userName")  
 userNamesMap[userId] = userName  
 }  
  
 // Kullanıcı adlarını yazılarla eşleştiriyoruz  
 postsList.*forEach* **{** post **->** val userName = userNamesMap[post.userId] ?: "Bilinmeyen"  
 Log.d("FetchUserNames", "Post UserId: ${post.userId}, UserName: $userName")  
 post.kullaniciAdi = userName  
 **}** postsAdapter.notifyDataSetChanged()  
 **}** .addOnFailureListener **{** exception **->** Toast.makeText(*context*, "Kullanıcı bilgileri yüklenemedi: ${exception.message}", Toast.*LENGTH\_SHORT*).show()  
 **}** }  
  
}  
  
data class Post(  
 val title: String = "",  
 val content: String = "",  
 val userId: String = "",  
 var kullaniciAdi: String = "", // Kullanıcı adı veya e-posta  
 val timestamp: Long = 0  
)  
  
class PostsAdapter(private val posts: List<Post>) : RecyclerView.Adapter<PostsAdapter.PostViewHolder>() {  
  
 class PostViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 val titleTextView: TextView = itemView.findViewById(R.id.*textViewTitle*)  
 val contentTextView: TextView = itemView.findViewById(R.id.*textViewContent*)  
 val userNameTextView: TextView = itemView.findViewById(R.id.*textViewUserName*) // Kullanıcı adı  
 val timestampTextView: TextView = itemView.findViewById(R.id.*textViewTimestamp*)  
 }  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): PostViewHolder {  
 val view = LayoutInflater.from(parent.*context*).inflate(R.layout.*item\_post*, parent, false)  
 return PostViewHolder(view)  
 }  
  
 override fun onBindViewHolder(holder: PostViewHolder, position: Int) {  
 val post = posts[position]  
 holder.titleTextView.*text* = post.title  
 holder.contentTextView.*text* = post.content  
 holder.userNameTextView.*text* = "Paylaşan: ${post.kullaniciAdi}"  
  
 // Timestamp'i okunabilir bir tarihe çevirme  
 val dateFormat = SimpleDateFormat("dd/MM/yyyy HH:mm", Locale.getDefault())  
 val date = java.util.Date(post.timestamp)  
 holder.timestampTextView.*text* = dateFormat.format(date)  
 }  
  
 override fun getItemCount(): Int {  
 return posts.size  
 }  
}  
  
  
**“Message.kt”**

package com.mychat.messageapp  
  
import com.google.firebase.Timestamp  
import java.util.\*  
  
data class Message(  
 val senderId: String = "",  
 var receiverId: String = "",  
 val message: String = "",  
 val timestamp: Long = 0L  
)

**“Kullanici.kt”**

package com.mychat.messageapp.Model  
  
data class Kullanici(  
 val email: String = "",  
 val kadi: String = "",  
 val isOnline: String = "Offline",  
 val uid: String = "",  
 val profilResmiUrl: String? = null  
  
)

**“MainActivity5.kt”**

package com.mychat.messageapp  
  
import android.os.Bundle  
import android.util.Log  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import android.widget.EditText  
import android.widget.ImageView  
import android.widget.RelativeLayout  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
import androidx.recyclerview.widget.LinearLayoutManager  
import androidx.recyclerview.widget.RecyclerView  
import com.google.firebase.auth.FirebaseAuth  
import com.google.firebase.firestore.FirebaseFirestore  
import com.google.firebase.firestore.Query  
import com.mychat.messageapp.databinding.ActivityMain5Binding  
  
class MainActivity5 : AppCompatActivity() {  
 private lateinit var binding: ActivityMain5Binding  
 private lateinit var firestore: FirebaseFirestore  
 private lateinit var currentUserId: String  
 private lateinit var otherUserId: String  
 private lateinit var messageAdapter: MessageAdapter  
 private val messagesList = *mutableListOf*<Message>()  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 binding = ActivityMain5Binding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
  
 binding.geritusu.setOnClickListener()**{** finish()  
 **}** // Firebase ve kullanıcı bilgileri  
 firestore = FirebaseFirestore.getInstance()  
 currentUserId = FirebaseAuth.getInstance().*currentUser*?.*uid* ?: return  
 otherUserId = *intent*.getStringExtra("uid") ?: return  
  
 val otherUserName = *intent*.getStringExtra("kullaniciAdi") ?: "Bilinmeyen Kullanıcı"  
 binding.kadigor.*text* = otherUserName  
  
 // RecyclerView setup  
 messageAdapter = MessageAdapter(messagesList, currentUserId)  
 binding.recmesaj.*apply* **{** *layoutManager* = LinearLayoutManager(this@MainActivity5)  
 *adapter* = messageAdapter  
 **}** // Mesaj gönderme  
 binding.gonder.setOnClickListener **{** val messageText = binding.mesajgirdi.*text*.toString().*trim*()  
 if (messageText.*isNotEmpty*()) {  
 sendMessage(messageText)  
 binding.mesajgirdi.*text*.clear()  
 }  
 **}** listenForMessages()  
 }  
  
 private fun sendMessage(messageText: String) {  
 val chatId = generateChatId(currentUserId, otherUserId)  
 val message = *hashMapOf*(  
 "senderId" *to* currentUserId,  
 "receiverId" *to* otherUserId,  
 "message" *to* messageText,  
 "timestamp" *to* System.currentTimeMillis()  
 )  
  
 firestore.collection("chats").document(chatId).collection("messages")  
 .add(message)  
 .addOnSuccessListener **{** Log.d("MessageSend", "Mesaj başarıyla gönderildi!")  
 **}** .addOnFailureListener **{** e **->** Log.e("MessageSend", "Mesaj gönderilemedi: ", e)  
 **}** }  
  
 private fun listenForMessages() {  
 val chatId = generateChatId(currentUserId, otherUserId)  
  
 firestore.collection("chats").document(chatId).collection("messages")  
 .orderBy("timestamp", Query.Direction.*ASCENDING*)  
 .addSnapshotListener **{** snapshots, e **->** if (e != null) {  
 Log.e("MessageListen", "Mesaj dinlerken hata: ", e)  
 return@addSnapshotListener  
 }  
  
 if (snapshots != null) {  
 messagesList.clear()  
 for (doc in snapshots) {  
 val message = doc.toObject(Message::class.*java*)  
 messagesList.add(message)  
 }  
 messageAdapter.notifyDataSetChanged()  
 binding.recmesaj.scrollToPosition(messagesList.size - 1)  
 }  
 **}** }  
  
 private fun generateChatId(user1: String, user2: String): String {  
 return if (user1 < user2) "$user1$user2" else "$user2$user1"  
 }  
}  
class MessageAdapter(  
 private val messages: List<Message>,  
 private val currentUserUid: String  
) : RecyclerView.Adapter<MessageAdapter.MessageViewHolder>() {  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): MessageViewHolder {  
 val view = LayoutInflater.from(parent.*context*).inflate(R.layout.*item\_message*, parent, false)  
 return MessageViewHolder(view)  
 }  
  
 override fun onBindViewHolder(holder: MessageViewHolder, position: Int) {  
 val message = messages[position]  
 if (message.senderId == currentUserUid) {  
 holder.bindSender(message) // Gönderenin mesajlarını bağla  
 } else {  
 holder.bindReceiver(message) // Alıcının mesajlarını bağla  
 }  
 }  
  
 override fun getItemCount(): Int = messages.size  
  
 // ViewHolder Sınıfı  
 inner class MessageViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {  
 private val senderTextView: TextView = itemView.findViewById(R.id.*senderTextView*)  
 private val messageTextView: TextView = itemView.findViewById(R.id.*messageTextView*)  
  
 fun bindSender(message: Message) {  
 // Gönderen için görünüm ayarları  
 senderTextView.*visibility* = View.*GONE* // Gönderenin adı gösterilmeyecek  
 messageTextView.*text* = message.message // Mesaj içeriğini göster  
 messageTextView.setBackgroundResource(R.drawable.*message\_background\_right*) // Sağda gösterim için arka plan  
 alignMessageToEnd() // Mesajı sağa hizala  
 }  
  
 fun bindReceiver(message: Message) {  
 // Alıcı için görünüm ayarları  
 senderTextView.*visibility* = View.*GONE* // Gönderenin adını göster  
 senderTextView.*text* = message.senderId // Gönderenin ID'sini göster  
 messageTextView.*text* = message.message // Mesaj içeriğini göster  
 messageTextView.setBackgroundResource(R.drawable.*message\_background\_left*) // Solda gösterim için arka plan  
 alignMessageToStart() // Mesajı sola hizala  
 }  
  
 private fun alignMessageToEnd() {  
 val layoutParams = messageTextView.*layoutParams* as RelativeLayout.LayoutParams  
 layoutParams.addRule(RelativeLayout.*ALIGN\_PARENT\_END*, RelativeLayout.*TRUE*)  
 layoutParams.removeRule(RelativeLayout.*ALIGN\_PARENT\_START*)  
 messageTextView.*layoutParams* = layoutParams  
 }  
  
 private fun alignMessageToStart() {  
 val layoutParams = messageTextView.*layoutParams* as RelativeLayout.LayoutParams  
 layoutParams.addRule(RelativeLayout.*ALIGN\_PARENT\_START*, RelativeLayout.*TRUE*)  
 layoutParams.removeRule(RelativeLayout.*ALIGN\_PARENT\_END*)  
 messageTextView.*layoutParams* = layoutParams  
 }  
 }  
}

**Proje teslim sürecinde yapılması gerekenler**

* Proje dosyaları, “öğrenci no\_ögrenci ad\_Mobil uygulamalar projesi” adında klasör oluşturulmalıdır.
* Projenin tamamı video kaydı ile çekilmeli, drive veya youtube linki Word dosyasına eklenmelidir. Video linki bulunmayan projeler değerlendirilmeyecektir.
* Proje dosyasına, uygulama geliştirilirken kullanılan tüm kodlamanın hepsi eklenmelidir.
* Proje teslimi, sınav tarihte toplanacaktır. Sınava girmeyen öğrencilerden final projesi teslim alınmayacaktır. Mail ile gönderilen ödevler değerlendirme kapsamına alınmayacaktır.
* Final projesi, oluşturulan Word dosyası ve kaydedilen video dikkate alınarak değerlendirilecektir.

**Not: Final notuna ödüllü puanların dahil edilmesi için Final sınavının gününde imza atılmalıdır.**