****

**GÖMÜLÜ SİSTEM ÇÖZÜMLEMELERİ**

**Dr. Öğr. Üyesi Fırat AYDEMİR**

**Mobil Film-Dizi Tanıtma Uygulaması**

**2022-2023 Güz Dönemi**

**16/10/2023**

**İsmail KEYVAN – 201913171030**

**Muhammed Emir GÖZCÜ – 201913171053**

**Giriş**

Teknolojinin ilerlemesi ile insanların günlük alışkanlıkları da değişiyor. Akşamları televizyon başında toplanmalara artık pek rastlanmıyor çünkü aile bireylerinin her birisinin elinde kendi istedikleri içeriği tüketmelerini sağlayan ekranlar var. Aynı şekilde akşamları fazla bir seçim şansı olmadan izlenen dizi veya filmlerin de modası geçti. Artık insanlar kendi izledikleri içerikleri kendileri seçiyor. İnternet, insanların korku, drama, bilim kurgu ve daha birçok kategori üzerinden seçebilecekleri ve Netflix, HBO, Amazon Prime, Disney+ gibi farklı platformlarda izleyebilecekleri film ve dizilerle dolu. Bizim amacımız ise insanlara bu devasa havuzdan kendi istekleri doğrultusunda tüketebilecekleri içerikleri sunmak. Kullanıcılar uygulamaya girince kategorilere ayrılmış olarak içerikleri görebilecek. Kullanıcılar arama çubuğunu kullanarak istedikleri filmleri listeleyebilmesini istiyoruz. Film veya dizilerin başlıklarını, açıklamalarını ve diğer özellikleri arayüzde gösterilecek. Henüz kesin olmamakla beraber seçilen içeriklerin bir “daha sonra izle” listesine eklenmesini planlıyoruz. Filmlerin ve dizlerin kategorilere ayrılarak kullanıcılara ulaştırılmasını planlıyoruz. Akıcı, göze hoş gelen, estetik bir arayüz tasarlamak istiyoruz. İçerikleri güncel olarak bir API’dan almayı ve gerekli düzenlemeler yapıldıktan sonra şık bir arayüz ile kullanıcılarla buluşturmayı planlıyoruz. İstekler ve gereksinimler doğrultusunda uygulamamıza farklı özellikler de kazandırmayı planlıyoruz.

**Yapılan Çalışmalar**

1. **Tubi TV, “Tubi – Movies & Tv Shows”**

**Konu:** TV dizilerini ve hit filmleri tanıtan bir uygulama. Filmleri komedi, dizi, çocuk klasikleri gibi kategorilere ayırarak kullanıcılara sunmuş. Uygulama tüm platformları destekliyor. Web siteleri üzerinden de uygulamaya erişim sağlanıyor.

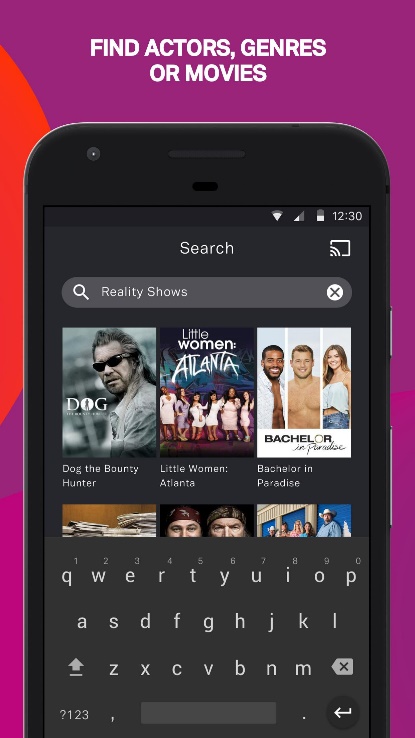
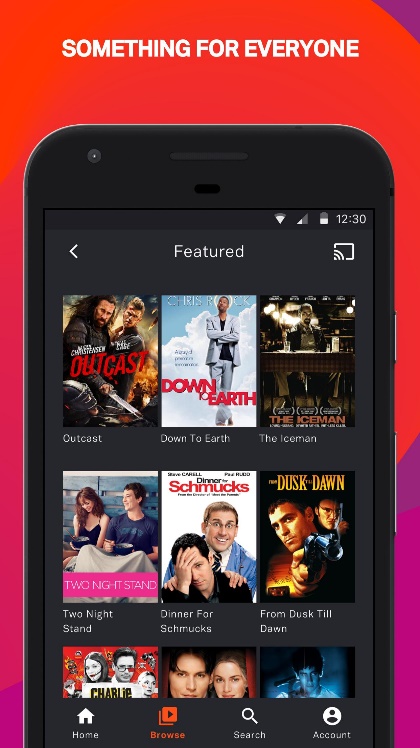
**Kullanılan Teknolojiler:** Swift – Kotlin

**Yöntem:** Turuncu ve pembe renkleri ana arayüz renkleri olarak kullanılmış ve yumuşak hatlı bir arayüz hazırlanmış.

**Sonuç:** Google Play üzerinde 100 milyondan fazla indirmeye ulaşmış ve 4.8 yıldız alan başarılı bir uygulama.

**Başlık:** Tubi TV

**Ekran Görüntüleri:**



1. **Popcornflix – Movies & TV – “Crackle Plus”.**

**Konu:** Filmleri ve TV dizilerini içerik olarak gösteren bir uygulama. İçerikler hakkında bilgiler göstermenin yanı sıra bu içeriklere erişim de sağlanmış. Kullanıcının seçtiği filmi uygulama içerisinden izleyebilmesi sağlanmış.

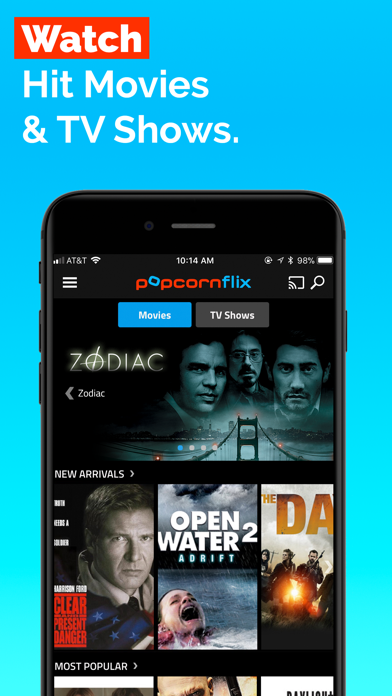
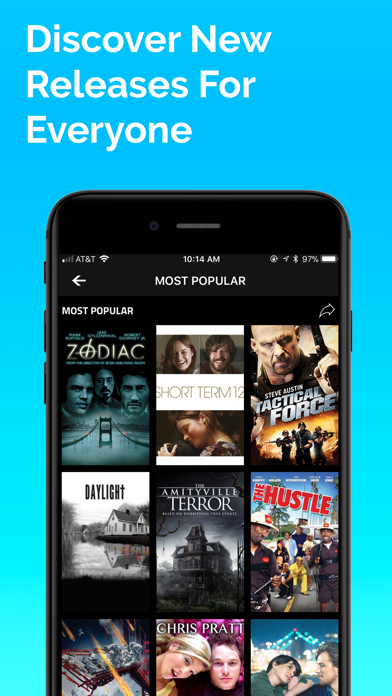
**Kullanılan Teknolojiler:** Swift– Kotlin

**Yöntem:** Koyu renkli bir tema rengi kullanılmış. Home, ekranın büyük bir kısmını önerilen dizlerin olduğu yatay bir liste ve geri kalanını kategoriler ve bu kategorilere ait içeriklerin gösterildiği bir liste kaplamış.

**Sonuç:** Google Play üzerinde 1 milyondan fazla indirilmiş, 2.9 yıldız almış bir uygulama.

**Başlık:** Popcornflix

**Ekran Görüntüleri:**



1. OlssonMatilda, Blekinge Institute of Technology, Faculty of Computing, Department of Software Engineering. When to prefer Flutter over native in mobile application development.

**Konu:** Mobil uygulama geliştirmede Flutter’ın ne zaman Kotlin veya Swift gibi Native dillerin yerine tercih edilmesi gerektiği performans ve görünüm açısından araştırmacılar tarafından ele alınmıştır.

**Kullanılan Teknolojiler:** Flutter, Dart, Kotlin, Swift

**Yöntem:** Bir mobil uygulama, her gün geliştirilen tüm yeni uygulamalarla rekabet edebilmek için ağır taleplere ayak uydurabilmelidir. İyi bir performans ve göze hitap eden görseller, mobil uygulamaların geliştirilmesi için temel gereksinimlerdir. Geliştirme sırasında araçlar için birçok seçenek vardır ve bunlardan biri daha iyi performansa sahip olduğu ve mobil ortama daha uygun olduğu söylenen Native uygulamalardır. Başka bir seçenek ise birden çok platform için yalnızca bir kod gerektiren ve bakımı daha kolay olan bir dildir.

Flutter, Google tarafından oluşturulan, açık kaynaklı bir Kullanıcı Arayüzü (UI) olan ve Native uygulama geliştirmenin bakış açılarına sahip, tek bir kod ile Cross-Platform (Çapraz-Platform) uygulamalar geliştirmemize izin veren yeni nesil bir yazılım dilidir.

Bu makale, Flutter'ın, şu anda mobil davranış ve performansta üstün görülen Native uygulamalarla nasıl karşılaştırıldığını araştırıyor. Bir deney

Flutter'ın bir çapraz derleyici Android Studio ile yapılmış bir Kotlin uygulaması ve XCode kullanılarak yapılmış bir Swift uygulaması olarak iki Native uygulamaya kıyasla CPU performansı açısından nasıl olduğunu test etmek için yapıldı. Flutter inanılmaz bir hızla büyümeye devam eden yeni bir araçtır. Flutter, iş CPU performansına gelince Native bir uygulama ile başa baş rekabet edebiliyor. Flutter, basit bir uygulama oluşturmak için karmaşık kod gerektirmez ve Native uygulamalar ile karşılaştırıldığında geliştirme aşamasında önemli ölçüde daha az kod satırı kullanımı olanağı sağlar. Flutter, küçük ila orta ölçekli oluştururken kullanılması en verimli yazılım dillerinden birisidir. Ancak daha büyük çaplı uygulamalar geliştirirken ortaya çıkan mevcut dezavantajlarının üstesinden gelmek için büyüme potansiyeline sahiptir.

**Native Mobil Uygulamalar**: Native’in mobil uygulamalar alanındaki anlamı belirli bir platformda veya işletim sisteminde çalışacak şekilde oluşturulmuş uygulamaları ifade eder. Native mobil uygulamalar oluşturmak için birçok dil var. Bunlara bazı örnekler: Kotlin, Java ve Swift. Esas olarak geliştiricilerin Native uygulama geliştirmeye odaklanmalarının sebebi Native cihazlar için kamera erişimi gibi özelleştirilmelerin etkinleştirilmesidir.

Yerel uygulamalarla ilişkilendirilen başka bir şey, daha akıcı görünen animasyonlar ve mobil teknoloji ile daha kolay entegre olmalıdır. Native uygulamalar hedeflenen platformların görünümünü devralır ve kendi platformlarına uygular. Buna “Native UI” (yerel görünüm) denir. Bu, Native uygulamaların kullanıcıya “Native” sisteme daha uygun davranan, görünen ve mobil platform işletim sisteminin kendisiyle daha ilişkilendirilebilir bir deneyim sağlar.

**Çapraz Platform Mobil Uygulama Geliştirme:** Çapraz platform, geliştirildiği platformdan başka bir platformda kullanılabilen bir ürün veya yazılımı ifade eder. Uygulama geliştirmede çapraz platform oluşturmanın amacı yalnızca te bir kod tabanı oluşturmak ve sürdürmektir. Tek bir platform için tek bir kod tabanı geliştiren Native ile karşılaştırıldığında geliştirmede zaman kazandırması ve tek bir kod tabanı kullanması, çapraz platform geliştirmenin en çekici parçasıdır. Çapraz platform araçlarına örnekler Flutter, React Native ve Ionic olarak verilebilir.

metin, elektronik eşyalar içeren bir resim

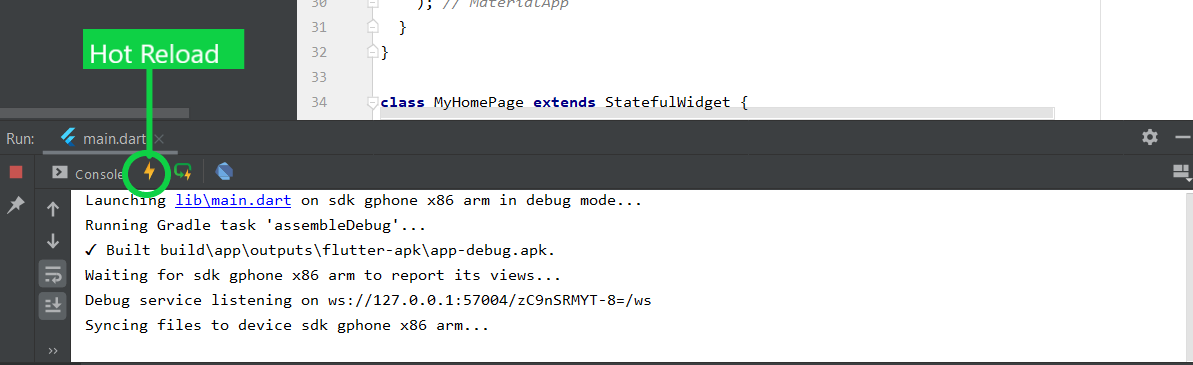
Açıklama otomatik olarak oluşturuldu

Native dillerini kullanarak Android ve IOS platformları için ayrı ayrı uygulama kodlamak gerekir. Swift ile IOS platformuna uygun bir uygulama kodlanırken Kotlin ile Android platformu için bir uygulama kodlanır. Flutter ile Cross-Platform (Çapraz Platform) uygulama kodlamak mümkündür. Bu sayede Android ve IOS platformları için tek bir uygulama oluşturmak yeterlidir.

**Flutter:** Flutter, Google tarafından geliştirilen ve Dart programlama dili tabanlı bir mobil SDK (Software Development Kit – Yazılım Geliştirme Kiti) ve UI (User Interface – Kullanıcı Arayüzü) aracıdır. Geliştirmecilerin amacı Native uygulamaların performansına ve görünümüne en yakın hissi veren uygulamalar sunmaktır.

**Dart:** Dart, Google tarafından oluşturulmuş bir programlama dilidir. Dart client-optimised ve birden fazla platformda kullanılabilen bir dildir. C tarzı bir syntax’a (sözdizimi) sahiptir ve sınıfları kullanan nesneye yönelimli bir dildir. Dart kodu hem JavaScript’e hem de Native tabanlı koda derlenebilir.

**Flutter/Dart UI Yönetimi:** Flutter koddaki ana kavram olarak widget’ları kullanır. Widget, Flutter’da yerleşik olarak bulunan her bir bileşen parçasının takma adıdır. Bu “box” veya “text” olarak adlandırılan bir widget anlamına gelir. Widget’ların göze çarpan bir parçası, Flutter geliştiricileri tarafından Native olarak gözükecek şekilde ve tamamıyla özelleştirilebilecek şekilde tasarlanmış olmalarıdır. Flutter, geliştiricilerin daha en baştan kalabilecekleri Material Library Components (Malzeme Kitaplığı Bileşenleri) kütüphanesini varsayılan olarak kullanır. Flutter “hot reload” denilen bir geliştirme fonksiyonelliğine sahiptir. Hot reload, değişiklikler yapıldığında Dart Sanal Makinesine enjekte edilir ve Flutter yapıyı yeniden oluşturur. Bu da yapılan değişikliklerin kullanıcıya daha hızlı gösterilmesini sağlar.



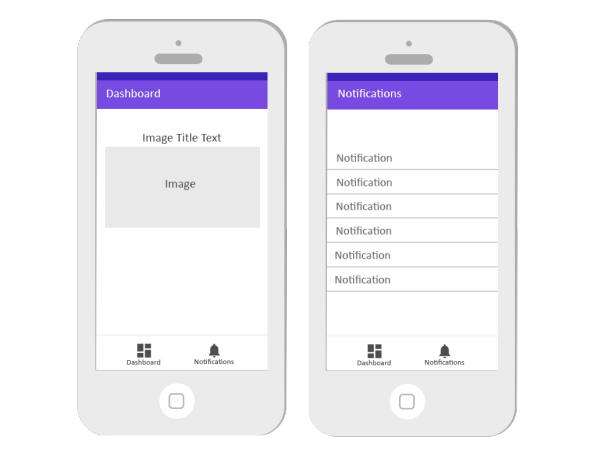
**Kotlin:** Kotlin, JetBrains tarafından geliştirilmiş bir programlama dilidir. Statik olarak yazılmış açık kaynaklı bir yazılım dilidir. Bunu yaparken sonuç bir derleme hatası olabilir. Kotlin, Java ile tamamen uyumludur ve Android uygulamaları oluşturmak için Android platformu tarafından desteklenir

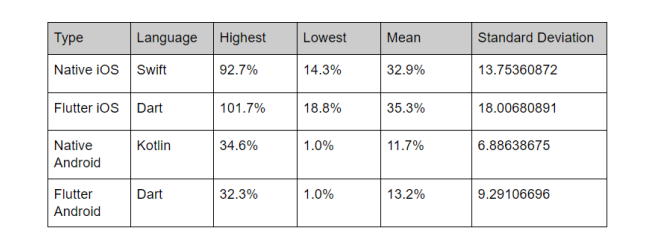
**Kotlin/Android UI Yönetimi:** Android’de Kotlin ile UI (kullanıcı arayüzü) tasarlarken geliştirici Android view (Android görünüm) grubu sistemini kullanmaya eğilimlidir. Düzen ve kullanıcı arayüzü öğeleri, çalışma zamanında (runtime) XML ile veya kodda belirtilmelidir. Sürükle ve bırak, programlanamaz düzen oluşturma için mevcuttur. Bu seçeneklerin her ikisinin de kullanıcı arayüzü parçalarının oluşturulduğunda sınıf düzeni için “onCrate” yöntemi ile koda bağlanması gerekir. Düzendeki (layout) nesneler viewGroups (görünüm grupları) ve views (görünümler) olarak sevk edilir. Görünümler, bir metin, nesne veya görüntü olabilen widget’lar olarak adlandırılır. Bu görünümler, düzen (layout) diyebileceğimiz görünüm grupları tarafından sarılır.

**Swift:** Swift, Apple tarafından geliştirilmiş bir programlama dilidir. Birden fazla platformda çalışabilmesi ve C dillerinin yerini alması için oluşturulmuştur. Swift, parçalara ayrılmış grup projeleri tarafından yönetilir. Bunlar: Swift Compiler (Derleyici), standart kütüphane, çekirdek kütüphaneleri, LLDB kütüphanesi, Swift paket yöneticisi ve XCode olarak yazılabilir. XCode, Swift ile projeler geliştirilebilen bir IDE’ dir (Entegre Geliştirme Ortamı)’.

**XCode UI Yönetimi:** XCode’da bir kullanıcı arayüzü geliştirirken, storyborad’lar (hikâye tahtası) oluşturmak için Interface Builder (Arayüz Oluşturucu) kullanılır. Sahneler, cihaz ekranında olan olayları temsil etmek için storyboard’da kullanılır. Segues, sahneleri birbirine bağlar ve ilişkilerini destekler. Görüntülenecek yeni bir öğe oluşturmak için sahne nesneleri sürüklenip bırakılabilir. Bu öğeler ve kontroller manuel olarak veya otomatik olarak oluşturulabilen XCode asistanı yardımıyla koda bağlanabilir.

**Flutter, Çalışma Zamanı CPU’sunda Android ve IOS’daki Native Uygulamalara Kıyasla Ne Kadar İyi Performans Gösteriyor?**

Diller ve platformlar arasındaki farkı ölçmek için her bir dil ile küçük bir uygulama yapalım. Flutter (iki uygulama), Kotlin ve Swift ile toplam dört uygulama yapılmıştır. Uygulamalar birbirine benzeyecek şekilde kendi dokümantasyonlarını takip ederek kodlanmış ve çalıştırılırmışlardır. 



**Resim:** Her geliştirme platformu için CPU ölçümünün bireysel sonuçlarını gösteren tablo.

CPU kullanım testlerini çalıştırmak için Native ve Flutter yapılarını kullanarak CPU kullanım testlerinden farklı sonuçlar alınmıştır. Her iki IOS uygulaması tablodan anlaşılabildiği üzere başlangıçta yüksek CPU kullanımı göstermiştir. Android Flutter uygulaması daha düşük maksimum CPU kullanımına sahip oldu ama Native Android daha da düşük bir ortalamaya sahip olmuştur. Genel olarak Android uygulamaları başlangıçta IOS uygulamalar kadar olmasa da yüksek bir CPU kullanımı gösterdi.

**Başlık**: A Comparison of Performance and Looks Between Flutter and Native Applications

**Sonuç:** Flutter ile çalışmayı daha hızlı yapan ve daha az kod kullanılmasını sağlayan şey Material Components (Materyal Bileşenler)’dir. Material Components, kullanıma hazır halde gelen ve Native görünüme sahip olması amaçlanan bir widget kütüphanesidir. Ancak Flutter’ın öğelerini monte etmek için kullandığı yuvalama, geliştirici için kolayca bir çorbaya dönebilir. Bunu tespit etmeyi kolaylaştırmak için olayı işaretleyen otomatik oluşturulmuş etiketler olsa dahi iş içinden çıkılamaz bir duruma girebilir. Flutter Android ve IOS için kod tabanı içeren bir çapraz platform geliştirme aracıdır ve bu dikkate alınması gereken bir gerçektir. Flutter uygulaması oluşturmak için Native ile karşılaştırıldığında daha karmaşık bir kod ihtiyacı yoktur.

Android Native koduyla çalışmayı zorlaştıran şey, Android ortamı ve Kotlin kodunun nasıl ayrılmış kullanıcı arayüz düzenlerine bağlantıyı ele aldıklarıdır. Kotlinde yeni başlayan bir geliştirici için ele alması zor olacak Android’e özel birçok kod parçası vardır.

IOS/Swift, Android’deki gibi kullanıcı arayüzü düzenini bağlayarak hem Flutter hem de Android/Kotlin’e benzer bir yaklaşıma sahiptir. Ayrıca Flutter’daki gibi daha kolay bir şekillendirme sistemine sahiptir.

**Yapılacak Olan Çalışmala**r**:** Bir sonraki rapor döneminde literatür taramalarına devam etmeyi planlıyoruz. Hangi programlama dili ile hangi editörü kullanmayı kararlaştırıp kullanacağımız dil hakkında daha fazla bilgi toplamayı planlıyoruz.

**Kaynakça**

[1] Tubi TV: <https://gdpr.tubi.tv/>

[2] Popcornflix: <https://popcornflix.com/>

[3] <https://www.diva-portal.org/smash/get/diva2:1442804/FULLTEXT01.pdf>

[4] OlssonMatilda, Blekinge Institute of Technology, Faculty of Computing, Department of Software Engineering. When to prefer Flutter over native in mobile application development.