Valorant Ajanlar Uygulaması

1. Giriş

Valorant API kullanarak ajanları ve yeteneklerini gösteren, favorilere ekleme ve not alma özelliklerine sahip bir Flutter uygulamasıdır.

2. Özellikler

- Tüm Valorant ajanlarını görüntüleme
- Detaylı ajan bilgilerini inceleme
- Rollere göre filtreleme
- Arama fonksiyonu
- Ajanlara özel notlarla favorilere ekleme
- Karanlık/Aydınlık tema desteği
- Çoklu dil desteği (TR/EN)
- Çevrimdışı kullanım için önbellekleme
- Responsive tasarım

3. Teknoloji Altyapısı

3.1 Ana Paketler

- **Durum Yönetimi**: Stacked
- **Ağ İşlemleri**: Dio, Dio Nexus
- **Yerel Depolama**: Hive
- **Lokalizasyon**: Easy Localization
- **Bağımlılık Enjeksiyonu**: Get It
- **Animasyonlar**: Lottie
- **Kod Üretimi**: Build Runner

3.2 Modül Yapısı ve Bağımlılıklar

```
module/
├── core/                  # Temel altyapı modülü
│   ├── lib/               # Core fonksiyonlar ve utilities
│   └── dependencies:
│       └── very_good_analysis
│
├── common/                # Ortak bileşenler modülü
│   ├── lib/               # Shared widgets ve utilities
│   └── dependencies:
│       ├── cached_network_image
│       └── shimmer
│
├── widgets/               # UI bileşenleri modülü
│   ├── lib/               # Reusable widgets
│   └── dependencies:
│       ├── responsive_framework
│       └── shimmer
│
```

```
└── gen/                    # Kod üretimi modülü
    ├── lib/                # Generated code ve assets
    └── dependencies:
        ├── envied
        ├── dio_nexus
        ├── equatable
        ├── json_annotation
        ├── flutter_svg
        └── lottie
```

4. Dio Nexus Entegrasyonu

Dio Nexus, Dio için geliştirilmiş olduğum bir ağ yönetimi sağlayan bir pakettir. Bu uygulamada API isteklerini yönetmek ve hata yakalamalarını iyileştirmek için kullanılmıştır.

4.1 Kullanım Örneği

```
Future<IResponseModel<User?>?> getUsers() async {
    IResponseModel<Users?>? response =  await
nexusManager.sendRequest<Users, Users>(
        "api/users",
        requestType:  RequestType.GET,
        responseModel:  Users());
    return response;
}
```

sendRequest ile generic olarak model bazlı istek atabilir ve json verisini veya list json verisini parse ederek kullanıcıya direk model bazlı dönüşüm sağlar. Ayrıca primitive verileri için(bool, String, double veya int gibi.) sendPrimitiveRequest olarak istek atabilir ve dönüşlerini alabiliriz.
Dio Nexus sayesinde ağ istekleri merkezi olarak yönetilir ve gelişmiş hata işleme desteği sunar.

Dio Nexus sayesinde ağ istekleri merkezi olarak yönetilir ve gelişmiş hata işleme desteği sunar.

5. Proje Yapısı

```
lib/
├── app/                    # Uygulama konfigürasyonu
├── product/                # Ürüne özel implementasyonlar
│   ├── cache/             # Önbellekleme modelleri
│   ├── config/            # Uygulama ayarları
│   ├── init/              # Başlangıç mantığı
│   ├── manager/           # Yöneticiler (Network, Error, vb.)
│   └── widget/            # Yeniden kullanılabilir widgetlar
├── repository/            # Repository pattern implementasyonu
│   ├── agent/             # Ajan repository
│   └── favorite_agent/    # Favori ajan repository
├── services/              # Servis katmanı
├── ui/                    # UI katmanı
│   ├── themes/           # Tema konfigürasyonu
│   └── views/            # Uygulama ekranları
└── main.dart             # Giriş noktası
```

6. Mimari

Uygulama MVVM pattern ile geliştirilmiş olup, temiz mimari prensipleri uygulanmıştır.

7. UI Bileşenleri

7.1 Özel Widget'lar

- **CustomNetworkImage**: Resim yükleme ve önbellekleme
- **LottieError**: Hata durumu animasyonu
- **LottieNotFound**: Bulunamadı durumu animasyonu
- **LottieNoInternetConnection**: Ağ hatası animasyonu

7.2 Tema Desteği

- **Kalıcı tema seçimi**
- **Responsive boyutlandırma**

8. Güvenlik

- **Ortam değişkenleri Envied ile şifrelenir**
- **API yanıtları Hive ile güvenli şekilde önbelleklenir**
- **Hassas veriler düz metin olarak saklanmaz**

9. Kullanım Örnekleri

9.1 API Kullanımı

```
final agents = await agentRepository.getAgentsFromCache();
final agent = await agentRepository.getAgentByIdFromCache(id: 'agent-
uuid');
```

9.2 Widget Kullanımı

```
CustomNetworkImage(
  imageUrl: agent.displayIcon,
  height: 200,
  width: 200,
)
```

9.3 Tema Kullanımı

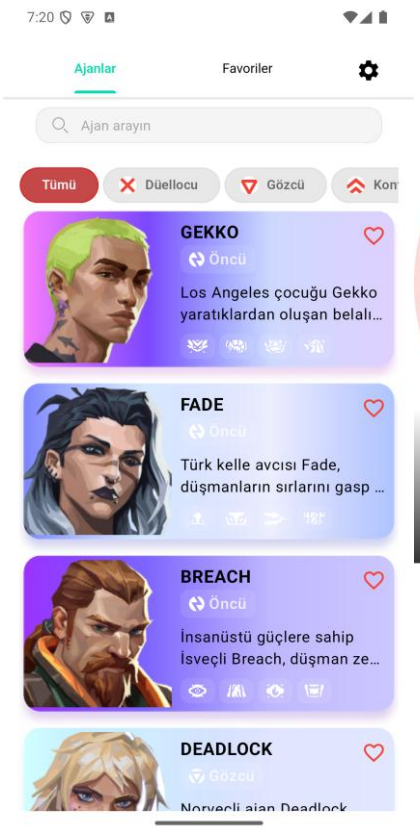
```
Locator<ThemeService>().updateThemeMode(AppThemeMode.dark);
Locator<LocalizationService>().updateLanguage(Locales.tr);
```

9.4 Önbellekleme Kullanımı

```
await productCache.agentCacheOperation.add(AgentCacheModel(agent: agent));
final cachedAgent = await productCache.agentCacheOperation.get(agentId);
```

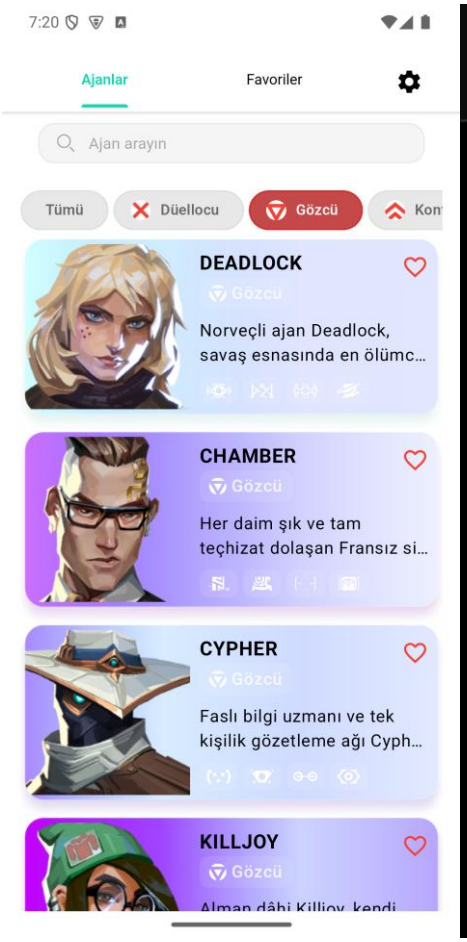
10. Sonuç

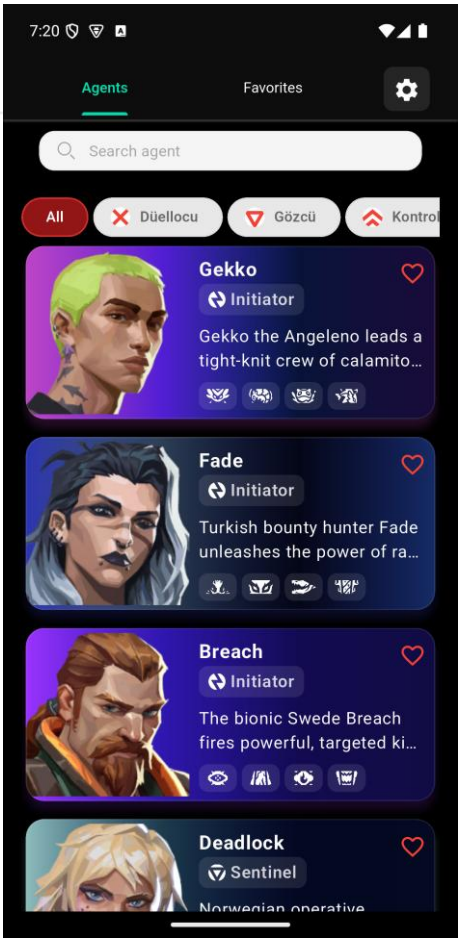
Bu doküman, Valorant Ajanlar uygulamasının teknik altyapısını, proje yapısını ve temel kullanım örneklerini kapsamaktadır. Geliştiricilerin projeyi genişletmesi ve katkı sağlaması için kapsamlı bir rehber niteliğindedir.

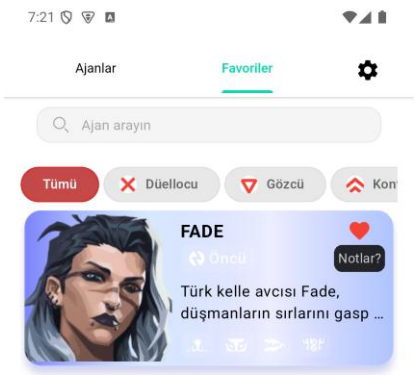


Los Angeles çocuğu Gekko yaratıklardan oluşan belalı tayfasına liderlik etmekte. Gekko'nun tayfası ileri atılarak düşmanların başına bela olur. Sonrasında Gekko onları toplar ve birlikte tekrar bela olacak düşman aramaya koyulurlar.

Yetenekler






7:21
Ajanlar
Favoriler
Ajan arayın
Tümü
Düellocu
Gözcü
Kon
FADE
Öncü
Notlar?
Türk kelle avcısı Fade, düşmanların sırlarını gasp ...