Akıllı Durak

1. Betül Erdik
211307033
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Kocaeli,Türkiye
Betuler5301@gmail.com

2. Seher Köroğlu
211307016
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Kocaeli,Türkiye
211307016@kocaeli.edu.tr

3. Melisa Ceylan
191307022
Bilişim Sistemleri Mühendisliği
Kocaeli Üniversitesi
Kocaeli,Türkiye
melisacy22@outlook.com

***Özet*—Bu çalışmada görme engelliler için yol tarifi uygulaması geliştirilmiştir. Uygulamamız android studio üzerinden java dili kullanılarak geliştirilmiştir. Akıllı Durak uygulaması görme engelli kullanıcıları gitmek istedikleri varış noktalarına ulaşım sağlayabilecek en yakın durağa varabilmeleri için gerekli yönlendirmeleri sağlar.**

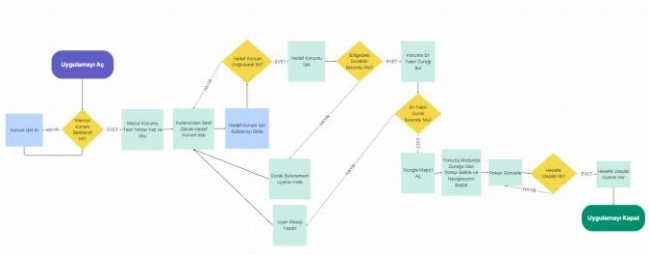
Anahtar Kelimeler —Android Studio, mobil app, akıllı durak, engelsiz app, Java

I. GIRIŞ

Kalabalıklaşan şehir hayatında artan trafik yoğunluğu ve karmaşıklığının en çok etkilediklerinden biri de görme engelliler. Bu yoğunlukta doğru, güvenli ve hızlı bir şekilde istenilen yere varmak için tasarlanmış uygulamalar bulunuyor. Bunlardan birisi de bizim yapmış olduğumuz akıllı durak uygulaması. Bu uygulamayla kullanıcı gitmek istediği yere gidebilmek için hangi durağa yürümesi gerektiği tespit ediliyor. Ardından bulunduğu konumdan varması gereken durağa kadar sesli rota kullanıcıya eşlik ediyor ve onu doğru yönlendirmelere durağa ulaştırıyor. Tasarlanmış olan bu uygulamada android studio üzerinde Java dilinde yazılmakta ve google maps API kullanılmaktadır.

II. ANDROID STUDIO KURULUMU

Uygulamanın gerçekleştirileceği platform olan Android Studio'nun kurulumu yapılır. Kurulum için Android Studio'nun sayfasında bulunan son sürüm(jellyfish) indirme şartları kabul edilerek indirilir. İlgili indirme dosyasına kaynakçalardan da ulaşabilirsiniz [4]. Kurulum indirilen exe dosyası üzerinden tamamlanır. Ve yeni proje başlatılır. Proje empty activity ve API 24 olacak şekilde istenilen dizinde oluşturulur. Ardından önce projenin akış diyagramı oluşturularak projeye başlanır [7] Resim1.



Resim1

III. GOOGLE MAP API

Gerçekleştirilecek proje daha çok harita üzerinden ilerleyeceği için önce projeye entegre edeceğimiz bir API belirlenir. Yapılan araştırmalar sonucunda Google Map API kullanmaya karar verildi [5]. API kullanımı için çeşitli kaynakları takip ederek izlenmesi gereken süreçler sırayla gerçekleştirildi [6]. Elde edilen API'yi projeye entegre etme aşamasında önce build.gradle.kts dosyasına google play servisini eklendi. Manifests/AndroidManifest.xml dosyası içerisine API KEY'le ilgili meta verileri, Main Activity içerisine de harita oluşturma kodları eklenir. Daha sonra values/google_maps_api.xml dosyasına API Key'i ekleyerek API tanımlaması bitirilir.

IV. OVERPASS API

A. *Overpass Response*

Uygulamada olacak duraklar için Overpass API kullanılır. Bunun için java klsörü altında bir OverpassResponse dosyası oluşturulur. Bu dosya üzerinden Overpass sayfasına istek yollanır. Ilgili istek kodu busStop adlı nesne listesinin bulunacağı elements adlı bir JSON alanı ve bu elementlerin get ve set edildiği fonksiyonu içerir.

B. *Bus Stop*

Belirtilen java/com.example.akillidurak/BusStop dosyasıysa get ve set methodlarıyla otobüs duraklarının ilgili verilerini tutumak için kullanılan bir java sınıfıdır.

C. *Overpass API Helper*

Overpass API'nin asıl kullanıldığı yer olan java/com.example.akillidurak/OverpassApiHelper dosyası ise otobüs duraklarının alındığı yerdir. OverpassApiHelper sınıfı Overpass API ile etkileşim kuracak yardımcı methodları içerir. Otobüs duraklarıyla ilgili geri bildirim sağlamak için BusStopListener arayüzü kullanılır ve gerekli doğrulama ve hata fonksiyonları tanımlanır.

GetBusStopsInKocaeli methoduyla istenilen bölgedeki duraklara ulaşmak için Overpass API çağrılır. Burada istenilen bölgenin koordinatları bulunup eklenir. OverpassUrl ile overpass API sorgusunu içeren URL oluşturulur.

HTTP isteklerini yapmak için okHttpClient nesnesi tanımlanır. Oluşturulan URL request nesnesi oluşturulup kullanılarak HTTP isteği yapılır. İsteği asenkron olarak kuyruğa alıp beklemek içinse enqueue ve callback methodları kullanılır.

Istek başarısız olduğunda onFailure methodu çağırılarak hata mesajı loglanıp kullanıcıya iletilir.

Başarılı olduğunda yani onResponse methodu çağırıldığında yanıt responseBody nesnesine alınır ve build.gradle dosyasına gson kütüphanesi eklendikten [9] sonra gson kullanılarak overpassResponse nesnesine dönüştürülür. Otobüs durakları varsa onBusStopsReceived methodu ile dinleyiciye aktarılır. Yoksa onError methodu ile kullanıcıya iletilir.

V. GÖRÜNÜM OLUŞTURMA

Res/layout/activity_main.xml sayfasında iki adet editText oluşturulur. Bunlardan birinde bulunduğumuz konum

diğerindeyse gitmek istenilen konum belirtilir. Textlerin içleri "bulunduğunuz konum:" ve "hedef konum:" olarak düzenlenip sayfa içerisine sabitlemeleri yapılır. Ardından ekrana oynayan bir animasyon konulur. Bunun içinse build.gradle dosyasına lottie kütüphanesi eklenir [10]. Sonrasında sayfa üzerinde gerekli renklendirme ve düzenlemeler yapılarak önyüzü tamamlanır.

VI. Main Activity

A. Hedef Konum Belirleme

Sabitlemesi yapılan textler için Main Activity üzerinde textin dolu olmasına bağlı olarak fonksiyonlar oluşturulur. Uygulamadaki metni sesli komutlara dönüştürmek için bir textToSpeech nesnesi oluşturulur. Daha sonra konumun kontrolü için activityResultLauncher başlatılır. Burada kullanıcının söylediği sesli konumun içerik kontrolü yapılır. Matches adıyla sesli komutların oluşturulacağı bir liste oluşturulur ve atanır. Gelen sonuçların boş olmadığı ve doğru olduğunun kontrolü yapılır. Değilse gerekli geri bildirimler yollanır. Doğruysa verinin destination ögesine ve xml dosyasındaki destination id'sindeki editText ögesine atanması sağlanır. SetText(destination) ile hedef konum içerisine yazılması sağlanır. Kullanıcıdan girilen hedef konumun işlenmesi için handler nesnesi oluşturulur ve postDelayed ile ve ardından 1 saniye gecikme verildikten sonra findNearestBusStop(destination) fonksiyonu çağırılarak hedef konumun işlenmesi sağlanır.

B. Konum Erişimi, Durak Belirleme ve Navigasyon

1) OnInit Fonksiyonu:

OnInit fonksiyonu textToSpeech fonksiyonu başlatıldığında çağırılır. Bu fonksiyonda LocationManager ile cihazın konum servislerine erişilir. Oluşturulan criteria nesnesiyle en iyi konum bilgisine ulaşılır.

EditText bileşeni xml dosyasındaki id'si source olan bileşenle eşleştirilir.

CheckSelfPermission ile kullanıcının konumuna erişmek için gerekli izinlerin olup olmadığı kontrol edilir. Eğer izin verilmemişse requestPermission methodu çağırılarak izin istenir. Manifests/AndroidManifest.xml dosyasında da gerekli izinlerin tanımlamlamaları yapılır.

GetLastKnownLocation ile kullanıcının bilinen son konumu alınır ve location değişkenine atanır. Eğer location değişkeni boş değilse geocoder nesnesi ile coğrafi kordinatlar adres bilgisine çevrilir. Konum bilgisi mevcutsa enlem ve boylam değerleri alınıp bir adrese dönüştürülür ve currentLocation değişkenine atanır. Ardından bu adres editText bileşenine yazılır.

TextToSpeech değişkeni ile currentLocation değişkeni sesli olarak söyletilir.

Hedef konumu sormak için kullanılacak cümle bir prompt değişkenine atanır ve speak fonksiyonu ile söyletilir. Kullanıcıdan sesli bir komut girişi almak içinse listenForSpeechInput() fonksiyonu başlatılır.

2) FindNearestBusStop Fonksiyonu:

findNearestBusStop fonksiyonunda kullanıcının konumuna erişmek için gerekli izinler verilmemişse önceden oluşturmuş olduğumuz konum izni isteme methodu olan requestPermission methodu çağırılır. Manifests/AndroidManifest.xml dosyasında gerekli izinlerin tanımlamlamaları yapılır. Eğer izin verilmişse LocationManager kullanılarak bir Criteria nesnesi oluşturulup son kullanılan konum bilgisi alınır.

Bu konum bilgisine bağlı olarak overpass API ile seçilmiş olunan Kocaeli bölgesindeki durakların listesi alınır. Alınan durakların sonucunda bir otobüs durağı bulunamazsa uyarı mesajı gönderilir.

Gelen duraklar arasından en yakın otobüs durağını saklaması için nearestBusStop değişkeni, en kısa mesafeyi saklaması için minDistance değişkeni tanımlanır. For döngüsüyle tüm otobüs durakları dolaşılarak kullanıcıya en yakın otobüs durağı tespit edilir.

En yakın otobüs durağı bulunursa bir URI oluşturulur ve Google Maps "&mode=w" kullanılarak yaya modunda navigasyon başlatılır. Google Maps tanımlanmış bir intent nesnesi kullanılarak startActivity(intent) fonksiyonuyla haritalar başlatılır.

3) ListenForSpeech Fonksiyonu:

ListenForSpeech fonksiyonunda belirli bir süre sonra işlem başlatmak için handler nesnesi oluşturulur. Ve postDelayed ile belirli bir süre sonra çalışması için runnable tanımlanır.

Intent nesnesi tanımlandıktan sonra bu nesneye bağlı olarak konuşma tanıma özelliği için ACTION_RECOGNIZE_SPEECH, konuşma tanıma dili için EXTRA_LANGUAGE özelikleri ayarlanır.

Try-catch ile daha önce tanımlanmış olan ActivityResultLauncher kullanılarak intent başlatılır.

4) OnDestroy Fonksiyonu:

OnDestroy fonksiyonu kullanıcı uygulamadan çıktığında çağırılır. Super.onDestroy methoduyla üst sınıfın onDestroy methodu çağırılarak temel işlevsellik korunur.

Eğer textToSpeech nesnesi doluysa yani konuşma devam ediyorsa stop() methoduyla konuşma durdurulur ve shutdown() methoduyla textToSpeech methodu kapatılır.

SpeechTimeoutHandler ile handler nesnesinin dolu olup olmadığı kontrol edilir ve doluysa beklemekte olan tüm runnable ve mesajlar iptal edilip kuyruktan çıkarılır.

VII. Sonuç

Böylece java diliyle yazılmış olan akılllı durak mobil uygulaması tamamlanmış olur ve gerekli koşulları yerine getiren çalışan bir mobil uygulama elde edilmiş olur.

VIII. Zorluklar

Uygulamada kullanılacak harita Apisi belirlenirken başta Google Map API'si seçilse de sonradan Mapbox API farkedildi. Mapbox'ın kişiselleştirmeye açık olması, gerek görünüş gerekse üzerinde daha fazla geliştirme yapılabilmesi cezbediciydi. Bunla beraber Mapbox API'ye geçilme kararı alındı. Mapbox API için çok fazla entegrasyon gerekliydi. Gerekli olan entegrasyon yapılmasına rağmen istenilen oranda kalite elde edilemedi. Mapbox'ın daha çok yazılımda ileri seviyeye gelmiş kişiler için olduğunu anlaşıldı. Günlerin belki ayların bu yazılımın öğrenilip uygulanması için harcanması gerekiyordu ama kısıtlı vakit içerisinde buna ayrılacak zaman yoktu. Bu sebepten dolayı tekrardan Google Map API'ye geçilerek çalışmalar orada devam ettirildi.

Kaynaklar

[1] https://www.boniglobal.com/en/products/loud-steps-assistive-technology/

[2] https://hq-project.github.io/CS491-Website-Project/#reports

[3] https://github.com/BlindMapsOrg/BlindMaps

[4] Android Studio İndirme

[5] https://mapsplatform.google.com/

[6] https://kodnova.com/google-maps-api-key-nasil-alinir

[7] https://www.canva.com/design/DAGF-LE00v8/iejRHj3JmT5pHOkZG5fN0A/view?utm_content=DAGF-LE00v8&utm_campaign=designshare&utm_medium=link&utm_source=editor

[8] https://github.com/seherkoroglu/akilli_durak_mobil/tree/main

[9] https://semihcelikol.com/android-studio-gson-ile-json-parse-etme-islemleri/

[10] https://lottiefiles.com/free-animations/library

[11] https://wiki.openstreetmap.org/wiki/Overpass_API

[12] https://overpass-turbo.eu/

[13] https://gislayer.com/tr/docs/mobile/26-overpass-api-ile-filtreleme/

[14] https://stackoverflow.com/questions/19846541/what-is-windowmanager-in-android

[15] https://stackoverflow.com/questions/15516595/directions-api-on-android

[16] https://www.youtube.com/watch?v=ewfnxnGCK7w&list=PL20Zn-5nPIPHvLPq5xJTTImOd0qeNd9rW&index=4

[17] https://www.youtube.com/results?search_query=+speech+to+text+on+android+java+like+siri

[18] https://docs.mapbox.com/android/navigation/v2/api/2.17.13/libnavigation-core/com.mapbox.navigation.core.trip.session/-location-observer/

[19] https://developer.android.com/studio/releases?hl=tr

[20] https://www.youtube.com/watch?v=usJ9ro4XSPA&t=68s