Yazılım Geliştirme Labratuvarı-II

SeeStop

https://github.com/ArtunKARA/SeeStop

Emirhan Kayhan[1]
[1]211307023
[1]211307023@kocaeli.edu.tr

Artun Kara[2]
[2]211307030
[2]211307030@kocaeli.edu.tr

Esmagül Kılınçat[3]
[3]211307025
[3]211307025@kocaeli.edu.tr

I. ÖZET

Yazılım Geliştirme Laboratuvarı-II dersi kapsamında bizden görme bozukluğu veya engeli olan kişiler için sesli şekilde yönlendirme yapacak mobil tabanlı bir uygulama geliştirilmesi istenmektedir. Bu kapsamda geliştirilen projenin amacı, engelli bireylerin istedikleri lokasyonlara ulaşabilmeleri için sesli bir yönlendirme sistemi oluşturmaktır. Kullanıcılar, gitmek istedikleri lokasyonu sesli komutlarla belirleyebilecek ve bu komutlar harita üzerinde konum tespitine dönüştürülerek kullanıcının belirtilen hedefe ulaşması sağlanacaktır. Kullanıcının hareketi sırasında sesli yönlendirmelerle yol tarifi sağlanacaktır.

SeeStop ismini verdiğimiz bu uygulama Android Studio'da Java programlama dili kullanılarak geliştirilmiştir ve geliştirirken Google Map API'lerinden yararlanılmıştır.

Uygulamada süreç şu şekilde işlemektedir;

Kullanıcı gitmek istediği lokasyonu telefona sesli olarak söyler. Bu sesli girdi apiler ile metne çevrilir ve ardından kullanıcının gitmek istediği lokasyon harita üzerinde tespit edilir. Konum tespit edildikten sonra, kullanıcının mevcut konumundan gitmek istediği lokasyona doğru süreç başlar ve kullanıcının konumu anbean takip edilir.. Bu süreçte, "10 adım düz git, 5 adım sonra sola dön" gibi sesli yönlendirmeler ile kullanıcının doğru lokasyona ulaşmı sağlanır. Bu yönlendirmeler, belli şablonlardan oluşan kısıtlı bir setten oluşmuş ve tanımlı olabilir, eğer tanımlı değil ise apiden çekilir.

Lokasyona ulaşıldığında sesli yönlendirme sona erer ve uygulama hedefin sona erdiğini bildirir. Eğer çok gürültülü bir ortamdaysam ve sesli yönlendirmeyi uygulama anlamaz ise girdinin tekrar edilmesi için tekrar sesli komut ile kullanıcı uyarılır. Ayrıca yanlış bir yöne yönelirsem, "Rotadan çıkıldı" gibi uyarılarla doğru yola dönmem sağlanabilir.

Anahtar kelimeler: Android Studio , JAVA, Google Map API

Abstract

Within the scope of the Software Development Laboratory-II course, it is required to develop a mobile-based application that will provide voice guidance for people with visual impairments or disabilities. The aim of the project is to create an audio guidance system for stops that are transportation points for disabled people. Users will be able to determine the location they want to go to with voice commands, and these commands will be converted into location determination on the map, allowing the user to reach the specified destination. Directions will be provided with voice guidance during the user's movement.

This application was developed in Android Studio using the Java programming language and made use of Google Map APIs.

The user speaks the location he wants to go to the phone. This voice input is translated into text and then the location where the user wants to go is determined on the map. Once the location is determined, movement begins from the current user's location to the location they want to go to. In this process, voice guidance such as "Go straight for 10 meters, turn right, turn left after 5 meters" is used to ensure that the user reaches the correct location. These redirects may consist of a limited set of certain templates, if they are not defined, they will be withdrawn from the API.

When the location is reached, the voice guidance ends and the application notifies you that the destination has ended. If I am in a very noisy environment and the application does not understand the voice guidance, the user is warned with a voice command to repeat the input. Additionally, if I head in the wrong direction, warnings such as "Off route" can help me get back on the right path.

Key words: Android Studio , JAVA, Google Map API

II. GİRİŞ

A. Android Studio

Android uygulama geliştirme için resmi entegre geliştirme ortamı (IDE). Geliştiricilere uygulama yazmak, derlemek, hata ayıklamak ve dağıtmak için bir dizi araç sunar. Android uygulamaları, Java veya Kotlin gibi diller kullanılarak Android Studio'da geliştirilir.

B. Java

Oracle Corporation tarafından geliştirilen ve popüler bir programlama dilidir. Android uygulama geliştirme sürecinde sıklıkla kullanılan dillerden biridir. Android uygulamaları genellikle Java veya Kotlin dilleriyle yazılır.

C. Google Maps API

Google Haritalar platformunun bir parçası olan bir dizi programlama arayüzüdür. Geliştiricilere, haritaları uygulamalarına entegre etmelerine, harita verilerini özelleştirmelerine ve kullanıcı etkileşimleri için harita işlevselliği eklemelerine olanak tanır. Bu API, yer işareti eklemek, konumları işaretlemek, rota hesaplamak, trafik durumunu göstermek gibi birçok harita işlevselliği sağlar. Android uygulamaları genellikle Google Haritalar API'lerini kullanarak harita entegrasyonu yaparlar.

III. UYGULAMA ÖZELLİKLERİ

A. Sesli Komut Alma ve Metne Çevirme

Kullanıcı gitmek istediği lokasyonu telefona sesli olarak söyler. Bu sesli komut, sistem tarafından metne çevrilir.

B. Konum Tespiti

Çevrilen metin, harita üzerinde analiz edilerek hedef konum tespit edilir.

C. Sesli Yönlendirme

Kullanıcıya “10 adım düz git, sağa dön, 5 adım sonra sola dön” gibi talimatlar verilir. Bu yönlendirme komutları, belirli bir şablon setine dayalıdır ve kısıtlı bir komut dizisinden oluşur.

D. Rota Dışı Uyarısı

Kullanıcının yanlış yöne yönelmesi durumunda, “Rotadan çıkıldı” gibi uyarılar verilerek kullanıcının doğru rotaya dönmesi sağlanır.

IV. KULLANILAN API’LER

A. com.google.android.gms:play-services-maps:18.2.0

Google Haritalar Hizmetleri API'si, Android uygulamalarına Google Haritalar'ı entegre etmek için kullanılır. Bu API, harita görüntüleme, konum belirleme, yol tarifi gibi harita işlevselliğini sağlar.

B. com.google.maps.android:android-maps-utils:0.5

Bu kütüphane, Android için Google Haritalar'a yardımcı araçlar sağlar. Örneğin, marker'lar, poligonlar, yol çizgileri gibi harita öğelerini özelleştirmek için kullanışlı işlevler içerir.

C. com.squareup.okhttp3:okhttp:4.9.3

OkHttp, Android uygulamalarının HTTP istekleri yapmasını sağlayan bir kütüphanedir. İstekleri göndermek ve yanıtları almak için kullanılır. JSON, XML gibi veri formatları ile çalışırken kullanılabilir.

D. com.android.volley:volley:1.2.1

Volley, Android uygulamaları için HTTP istekleri yapmak ve yanıtları işlemek için kullanılan bir kütüphanedir. Network işlemlerini yönetmek ve kolayca kullanmak için tasarlanmıştır.

E. com.google.code.gson:gson:2.8.8

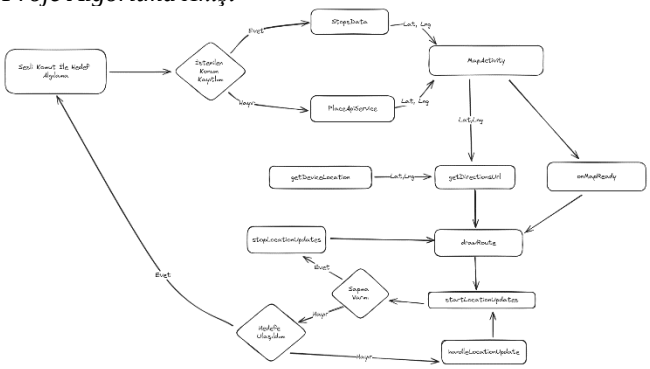
Gson, Java nesnelerini JSON formatına dönüştürmek ve tersine dönüştürmek için kullanılan bir kütüphanedir. Özellikle web servisleriyle iletişim kurarken JSON verilerini kolayca işlemek için yaygın olarak kullanılır.

F. com.mapbox.maps:android:11.4.0

Mapbox Maps SDK for Android, Android uygulamaları için harita işlevselliği sağlayan bir kütüphanedir. Mapbox platformunun sunduğu harita hizmetlerine erişim sağlar ve özelleştirilebilir harita görüntüleme özellikleri sunar.

V. GELİŞTİRME SÜRECİ

A. Proje Algoritma Akışı



Şekil 1

1) Sesli Komut İle Hedef Algılama

Bu kod, bir Android uygulamasında kullanıcının konuşmasını tanıyarak bilgi girişi almayı amaçlar. askspeechinput metodu, konuşma tanıma işlemini başlatır. Bir Intent nesnesi oluşturulur ve RecognizerIntent.ACTION_RECOGNIZE_SPEECH eylemi belirlenir. Dil modeli serbest form olarak ayarlanır (RecognizerIntent.LANGUAGE_MODEL_FREE_FORM) ve konuşma tanıma dili Türkçe ("tr-TR") olarak belirtilir. Kullanıcıya "Durak Konumu İçin Konuşunuz" mesajı gösterilir ve aynı mesaj sesli olarak da iletilir. startActivityForResult(intent, RQ_SPEECH_REC) metodu ile konuşma tanıma aktivitesi başlatılır. Eğer cihazda konuşma tanıma özelliği yoksa, ActivityNotFoundException yakalanır ve kullanıcıya bir Toast mesajı ve sesli uyarı gösterilir.

2) StopsData

Bu Java sınıfı, belirli şehir adlarına göre enlem ve boylam değerlerini sağlayan bir yapı içerir. StopsData sınıfı, şehirlerin enlem ve boylamlarını saklamak için iki ayrı HashMap (cityLatitudes ve cityLongitudes) kullanır. Statik başlatıcı blokta "b kapısı", "a kapısı" ve "hop stop" gibi şehir adlarına karşılık gelen enlem ve boylam değerleri bu haritalara eklenir. getLatitude ve getLongitude metodları, verilen şehir adını küçük harflerle alır ve ilgili haritalardan enlem veya boylam değerini döner; eğer şehir adı

bulunamazsa 0.0 döner. Bu sınıf, durağın adıyla enlem ve boylam koordinatlarına kolay erişim sağlar.

3) PlaceApiService

Bu PlaceApiService sınıfı, Google Places API'yi kullanarak yer arama ve yer detaylarını alma işlevselliği sağlar. İki sabit URL (PLACE_SEARCH_URL ve PLACE_DETAILS_URL) ve bir API anahtarı (API_KEY) tanımlanmıştır. Sınıf, bir RequestQueue nesnesi kullanarak HTTP isteklerini yönetir. Yapıcı metod, Volley kütüphanesini kullanarak bu kuyruk nesnesini başlatır. getPlaceId metodu, verilen sorguya dayalı olarak bir yerin kimliğini almak için bir URL oluşturur ve bir JsonObjectRequest ile isteği yapar; başarılı olursa, PlaceIdCallback aracılığıyla yer kimliğini döner, aksi takdirde hata mesajı döner. getPlaceDetails metodu, belirli bir yer kimliği için yer detaylarını almak üzere benzer şekilde çalışır ve yerin enlem ve boylamını PlaceDetailsCallback ile döner; bir hata durumunda hata mesajı döner. Sınıfın içinde PlaceIdCallback ve PlaceDetailsCallback arayüzleri tanımlanmış olup, bu arayüzler başarı ve hata durumlarını yönetir.

4) getDirectionsUrl

Bu getDirectionsUrl metodu, iki LatLng nesnesi (başlangıç ve varış noktaları) alarak, Google Directions API'yi kullanarak yürüme yönlerini almak için gerekli URL'yi oluşturur. İlk olarak, başlangıç ve varış noktalarının enlem ve boylam değerleri str_origin ve str_dest değişkenlerine atanır. Ardından, ulaşım modu "yürüyüş" olarak belirlenir (mode=walking) ve API anahtarı (key) eklenir. Bu parametreler parameters değişkeninde birleştirilir ve tüm parametreler bir araya getirilerek Google Directions API'nin URL'si oluşturulup geri döndürülür.

5) getDeviceLocation

Bu getDeviceLocation metodu, cihazın mevcut konumunu almayı ve işlemi gerçekleştirmeyi amaçlar. İlk olarak, gerekli konum izni (Manifest.permission.ACCESS_FINE_LOCATION) kontrol edilir. Eğer izin verilmişse, bir LocationRequest nesnesi oluşturulur ve konum güncelleme parametreleri ayarlanır: konum güncelleme aralığı 5000 milisaniye, en hızlı aralık 2000 milisaniye ve yüksek doğruluk önceliği (PRIORITY_HIGH_ACCURACY).

Bir LocationCallback nesnesi tanımlanır ve onLocationResult metodu ile konum sonuçları işlenir. locationResult boş değilse, içindeki her Location nesnesi için mevcut konum LatLng nesnesine dönüştürülür. Eğer origin (başlangıç noktası) henüz ayarlanmamışsa, mevcut konum origin olarak belirlenir, bu konuma bir işaretçi eklenir, hedef konuma bir işaretçi eklenir ve kamera origin konumuna yakınlaştırılır. Ardından drawRoute metodu ile başlangıç ve hedef konum arasında bir rota çizilir.

Son olarak, fusedLocationClient ile konum güncellemeleri başlatılır ve bu güncellemeler ana iş parçacığı döngüsünde (Looper.getMainLooper()) işlenir. Eğer bir SecurityException yakalanırsa, bu hata yazdırılır.

6) onMapReady

Bu onMapReady metodu, harita kullanıma hazır olduğunda çalışır ve kullanıcının konum izni olup olmadığını kontrol eder. Harita kullanıma hazır olduğunda, GoogleMap nesnesi mMap değişkenine atanır. İlk olarak, konum izninin verilip verilmediği ContextCompat.checkSelfPermission ile kontrol edilir. Eğer izin verilmişse, getDeviceLocation metodu çağrılarak cihazın mevcut konumu alınır. Eğer izin verilmemişse, ActivityCompat.requestPermissions metodu ile kullanıcıdan konum izni istenir ve LOCATION_PERMISSION_REQUEST_CODE değeri ile izin isteği başlatılır. Bu izin kontrolü, uygulamanın kullanıcı konumuna erişim izni almasını ve konum bilgilerini kullanabilmesini sağlar.

7) drawRoute

Bu drawRoute metodunun amacı, verilen iki konum arasında bir rota çizmek için Google Directions API'yi kullanmaktır. İlk olarak, getDirectionsUrl metoduyla API isteği için gerekli URL oluşturulur. Daha sonra, yeni bir iş parçacığı başlatılır ve bu iş parçacığında URL'ye HTTP bağlantısı yapılır, yanıt okunur ve işlenir. Son olarak, UI iş parçacığında rota yanıtı işlenir ve haritada çizilir. Ayrıca, rota çizildikten sonra konum güncellemeleri yeniden başlatılır. Bu metodun kullanımı, uygulamanın mevcut konum ile hedef arasında bir rota çizmesini sağlar ve bu işlemi arka planda gerçekleştirerek uygulamanın kullanıcı arayüzüne etkileşimli bir şekilde yanıt vermesini sağlar.

8) stopLocationUpdates

stopLocationUpdates metodu, konum güncellemelerini durdurmayı amaçlar. Bu metod, fusedLocationClient ve locationCallback değişkenlerinin null olup olmadığını kontrol eder. Eğer her ikisi de null değilse, fusedLocationClient.removeLocationUpdates(locationCallback) çağrısı ile konum güncellemeleri durdurulur. Bu yöntem, konum güncellemelerinin gereksiz yere devam etmemesi için kullanılır ve genellikle uygulama durduğunda veya belirli bir olay gerçekleştiğinde konum izleme işlemlerini sonlandırmak için çağrılır.

9) startLocationUpdates

startLocationUpdates metodunun amacı, konum güncellemelerini başlatmaktır. İlk olarak, ContextCompat.checkSelfPermission ile gerekli konum izni (Manifest.permission.ACCESS_FINE_LOCATION) kontrol edilir. Eğer izin verilmişse, bir LocationRequest nesnesi oluşturulur ve konum güncelleme parametreleri ayarlanır: konum güncelleme aralığı 5000 milisaniye, en hızlı aralık 2000 milisaniye ve yüksek doğruluk önceliği (PRIORITY_HIGH_ACCURACY).

Daha sonra, bir LocationCallback nesnesi tanımlanır ve onLocationResult metodu ile konum sonuçları işlenir. locationResult boş değilse, içindeki her Location nesnesi için mevcut konum LatLng nesnesine dönüştürülür ve handleLocationUpdate metodu ile işlenir.

Son olarak, fusedLocationClient nesnesi ile konum güncellemeleri başlatılır ve bu güncellemeler ana iş parçacığı döngüsünde (Looper.getMainLooper()) işlenir. Bu metod, uygulamanın kullanıcının mevcut konumunu izlemesini

sağlar ve konum güncellemelerini işleyerek uygun şekilde kullanır.

10) handleLocationUpdate

handleLocationUpdate metodunun işlevselliği, kullanıcının mevcut konumunu takip ederek rotayı yönetmeyi içerir. İlk olarak, mevcut konum marker'ı oluşturulur veya güncellenir ve kamera, mevcut konumu odak noktası yapacak şekilde ayarlanır. Ardından, hedefe olan mesafe hesaplanır ve bir eşik değer olan PROXIMITY_THRESHOLD ile karşılaştırılır. Eğer kullanıcı hedefe yeterince yaklaşmışsa, bir sonraki adımın yönergesi hesaplanır ve kullanıcıya gösterilir. Bu adımda, rotadaki her bir nokta arasındaki mesafe ölçülerek kullanıcının bir sonraki adıma ne kadar yakın olduğu belirlenir ve gerekirse kullanıcıya ne kadar adım sonra hangi yöne gitmesi gerektiği söylenir.

Eğer kullanıcı hedefe ulaşmışsa, navigasyon tamamlandı mesajı verilir ve konum güncellemeleri durdurulur. Ancak, eğer kullanıcı rotadan sapmışsa (belirlenen eşik değerden daha fazla uzaklıkta hareket ederse), mevcut konum güncellemeleri durdurulur ve kullanıcıya rotadan sapıldığına dair bir mesaj verilir. Bu durumda, kullanıcıya yeni bir rota hesaplanır ve yönlendirme yeniden başlatılır. Bu yöntem, kullanıcının rotayı izlemesini ve gerektiğinde yeniden yönlendirilmesini sağlar, böylece kullanıcı rotadan saparsa uygun şekilde reaksiyon verilir ve kullanıcı hedefe güvenli bir şekilde ulaşır.

REFERENCES

Android studio. [1]

Kullanılan API'ler[2][3][4]

[1] https://developer.android.com/studio
[15.05.2024]

[2] https://developers.google.com/android/guides/setup
[15.05.2024]

[3] https://maven.apache.org/repository/
[17.05.2024]

[4] https://jfrog.com/blog/secure-jcenter-with-https/
[18.05.2024]