Ses Verisi

Betül Erdik   
betulerdik41@gmail.com  
211301033Bilişim Sistemleri Mühendisliği  
Kocaeli Üniversitesi

Buse Çakal   
221307084@gmail.com  
221307084Bilişim Sistemleri Mühendisliği  
Kocaeli Üniversitesi

*Özet*—Bu proje, belirli bir web sitesindeki ses dosyalarının otomatik olarak taranması, indirilmesi ve Google Drive’a yüklenmesini sağlamayı amaçlamaktadır. Python programlama dili ile geliştirilen projede Selenium, Requests ve Google Drive API gibi araçlar kullanılmıştır. Bu sayede ses dosyaları, güvenli bir şekilde toplanıp depolanabilmektedir.

Anahtar Kelimeler: Python, Selenium, Google Drive API, Requests, Web Scraping.

# giriş

# Günümüzde dijital veri yönetimi, giderek daha önemli bir hale gelmiştir. Özellikle ses dosyaları gibi medya içeriklerinin toplanması, düzenlenmesi ve paylaşılması, hem bireysel kullanıcılar hem de organizasyonlar için büyük bir ihtiyaç oluşturmuştur. Bu bağlamda, web tarama (web scraping) teknikleri ve Google Drive gibi bulut tabanlı depolama çözümleri, verimliliği artırmak için etkili araçlar sunmaktadır.

# Bu projede, belirli bir web sitesinde bulunan ses dosyalarının (örneğin, .wav formatındaki dosyaların) taranarak tespit edilmesi, kullanıcı sistemine indirilmesi ve Google Drive platformuna yüklenmesi amaçlanmıştır. Bu süreç, Python programlama dilinde geliştirilmiş bir otomasyon sistemi ile gerçekleştirilmiştir. Projede, Selenium, Requests ve Google Drive API gibi modern teknolojilerden yararlanılmıştır.

# Proje, yalnızca verilerin toplanmasını değil, aynı zamanda Google Cloud Platform üzerinden sağlanan yetkilendirme mekanizmaları sayesinde güvenli bir şekilde depolanmasını da sağlamaktadır. Bu yaklaşım, ses dosyalarının düzenli bir şekilde saklanmasını ve yönetilmesini mümkün kılarken, aynı zamanda manuel işlem gereksinimini en aza indirmektedir.

# Bu raporda, projenin amacına, kullanılan teknolojilere, yapılandırma detaylarına ve geliştirilen kodun işlevselliğine ilişkin ayrıntılar sunulmaktadır. Projenin geliştirme süreci boyunca karşılaşılan zorluklar ve bu zorluklara yönelik çözümler de rapor kapsamında ele alınmıştır.

# amaç

# Bu proje, belirli bir web sayfasındaki ses dosyalarını otomatik olarak tespit etmeyi, indirmeyi ve daha sonra bu dosyaları bir bulut depolama alanına yüklemeyi hedeflemektedir. Ses dosyaları genellikle internet üzerinden müzik ve ses efekti arayışı içinde olan kullanıcılar tarafından sıklıkla kullanılmaktadır. Proje, web scraping tekniği ile ses dosyalarını toplayarak, bunları Google Drive gibi bir bulut platformuna yüklemeyi amaçlamaktadır.

Bu projede, Selenium ve Google Drive API gibi araçlar kullanılarak, web sayfalarındaki ses dosyaları verimli bir şekilde toplanacak ve kullanıcının sistemi yerine bulut ortamına yedeklenecektir. Bu işlem, özellikle büyük miktarda ses dosyasını manuel olarak indirme zorunluluğunu ortadan kaldırarak, zamandan tasarruf sağlamaktadır.

# projede kullanılan kütüphaneler

Bu projede aşağıdaki Python kütüphaneleri kullanılmıştır:

* Selenium: Web sayfalarındaki kullanıcı hareketlerini taklit ederek otomatik işlemler yapılmasına olanak tanır. Bu proje kapsamında, web sitesini açmak, gerekli sayfalarda gezinmek ve HTML içeriklerini almak için kullanılmıştır.
* Requests: HTTP istekleri yaparak web sayfalarından veya dosyalardan veri çekilmesini sağlar. Proje boyunca, ses dosyalarını indirmek için HTTP GET istekleri yapılmıştır.
* Google Drive API: Yüklenen dosyaların Google Drive'a aktarılması için kullanılmıştır. API, dosyaları belirli bir klasöre yüklemek için gerekli işlemleri otomatik olarak yerine getirir.
* Webdriver Manager: Selenium’un gereksinim duyduğu tarayıcı sürücülerini (örneğin, ChromeDriver) otomatik olarak indirir ve kurar. Bu kütüphane sayesinde, tarayıcı sürücüsünün manuel olarak güncellenmesi gerekmez.
* os: Python'un işletim sistemi modülü, dosya ve klasör yönetimi için kullanılmıştır. Bu modül, indirilen ses dosyalarının kaydedilmesi ve gerektiğinde düzenlenmesi için kullanılmıştır.
* google.oauth2: Google API'leri ile güvenli bir şekilde iletişim kurabilmek için kullanılan yetkilendirme ve kimlik doğrulama işlemlerini yönetir.
* googleapiclient.discovery: Google API'lerine erişim sağlamak için kullanılır. Bu proje için Google Drive API'ye erişim sağlamak amacıyla kullanılmıştır.

# Google Cloud API Ayarları

Projeye başlamak için ilk olarak Google Cloud Console üzerinde yeni bir proje oluşturulmuştur. Proje oluşturulduktan sonra, "API ve Hizmetler" sekmesinden Google Drive API'si etkinleştirilmiştir. Bu API, Google Drive ile veri alışverişi yapmak için kullanılır.

Daha sonra, "Kimlik Bilgileri" sekmesinde Hizmet Hesabı oluşturulmuştur. Hizmet hesabı, Google Drive üzerindeki dosyalara erişim izni sağlamak için kullanılır. Hizmet hesabı oluşturulurken uygun bir rol (örneğin, Drive File Access) atanmış ve API işlemleri için gerekli izinler verilmiştir.

Hizmet hesabının kimlik bilgileri, bir JSON dosyası şeklinde dışa aktarılmıştır. Bu JSON dosyasındaki bilgiler, service\_account.json olarak adlandırılmış ve proje dizinine kaydedilmiştir.

Son olarak, Google Drive üzerinde bir klasör oluşturulmuş ve bu klasöre yükleme yapmak için oluşturulan hizmet hesabına düzenleme yetkisi verilmiştir. Bu sayede, uygulama çalıştığında yüklenen dosyaların doğru şekilde bu klasöre aktarılması sağlanmıştır.

# Kod Açıklamaları

## Google Drive API Yetkilendirme ve Başlatma

Google Drive API'si ile dosya yükleyebilmek için yetkilendirme yapılır. service\_account.json dosyasındaki hizmet hesabı bilgileri kullanılarak yetkilendirme yapılır: **Kod1**

# Google Drive API için ayarlar

SCOPES = ['https://www.googleapis.com/auth/drive.file']

SERVICE\_ACCOUNT\_FILE = 'service\_account.json' # JSON dosyanızın adı

FOLDER\_ID = '13t9QvPmNiUe-fcdriNDRfSOX2b2gAv-\_' # Yüklemek istediğiniz klasörün ID'si

# Yetkilendirme Bilgilerini Ayarla

credentials = service\_account.Credentials.from\_service\_account\_file(

SERVICE\_ACCOUNT\_FILE, scopes=SCOPES)

# Google Drive Servisini Başlat

service = build('drive', 'v3', credentials=credentials)

**Kod1**

SERVICE\_ACCOUNT\_FILE: Google API için gerekli hizmet hesabı bilgilerini içerir.

FOLDER\_ID: Dosyanın yükleneceği Google Drive klasörünün ID'si.

credentials: Google API'ye bağlanmak için gerekli kimlik doğrulama bilgileri.

service: Google Drive API ile etkileşim kurmak için kullanılır.

## Google Drive'a Dosya Yükleme Fonksiyonu

Dosyayı belleğe indirildikten sonra, bu fonksiyon dosyayı Google Drive'a yükler:**Kod2**

# Google Drive'a dosya yükleme fonksiyonu

def upload\_file(file\_stream, file\_name):

file\_metadata = {

'name': file\_name,

'parents': [FOLDER\_ID] # Yüklemek istediğiniz klasörün ID'si

}

media = MediaIoBaseUpload(file\_stream, mimetype='audio/wav')

# Dosyayı yükle

file = service.files().create(body=file\_metadata, media\_body=media, fields='id').execute()

print(f"Dosya yüklendi: {file\_name} (ID: {file['id']})")

**Kod2**

file\_metadata: Yüklenecek dosyanın adı ve hedef klasör bilgisi.

MediaIoBaseUpload: Dosyanın yükleme işlemi için gerekli medya sınıfı.

## Web Scraping İçin Çerezler ve Başlıkların Ayarlanması

Web scraping yaparken, HTTP isteği gönderirken kullanılan başlıklar ve çerezler ayarlanır: **Kod3**

# Çerezler ve başlıklar

cookies = {

"preferSpectrogram": "no",

"disallowSimultaneousAudioPlayback": "no",

"systemPrefersDarkTheme": "no",

"cookieConsent": "yes",

"csrftoken": "eumpT1uCjeiPan2UMLZyjb5BfQoXxZ5fckz0P9ZGx4bwQG7aHoJlSBRfKbHgZ0ui",

"sessionid": "kt8jkxrrbyd8ywgfbwkjvspipo8vjc00"

}

headers = {

'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.0.0 Safari/537.36',

'Accept': 'application/octet-stream'

}

**Kod3**

Bu başlıklar, web sayfasıyla doğru şekilde etkileşim kurabilmek için gereklidir.

**(Headers) Bulma:**

* Network sekmesinde sayfayı yenileyin (F5).
* Sayfa yenilendikten sonra, burada birçok HTTP isteği (request) görünecektir. Genellikle GET veya POST gibi yöntemleri kullanarak yapılan istekler listelenir.
* İsteklerin üzerine tıkladığınızda sağ panelde Request Headers kısmını göreceksiniz. Burada tarayıcınızın yaptığı HTTP isteklerinde kullanılan başlıkları görebilirsiniz.
* **User-Agent:** Tarayıcınızın türünü belirten başlıktır (örneğin, Chrome, Firefox).
* **Accept:** Tarayıcınızın kabul ettiği içerik türlerini belirtir.
* Authorization veya cookie gibi başlıklar da olabilir, bu da oturum bilgisi sağlar.

**Çerezleri (Cookies) Bulma:**

Cookies bilgilerini görmek için:

* Application sekmesine gidin (Chrome'da).
* Sol tarafta Cookies sekmesini tıklayın. Burada, sayfanın yüklediği çerezlerin listesine ulaşabilirsiniz.
* Bu çerezler, oturum açmak için kullanılan ve web sitesiyle olan bağlantınızı sürdüren bilgileri içerir (örneğin, sessionid, csrftoken, vb.).
* Çerezlerin değerlerini kopyalayarak requests veya selenium'da kullanabilirsiniz.

## Dosya İndirme Fonksiyonu

Dosya indirme fonksiyonu, belirli bir URL'den ses dosyasını indirip belleğe yükler ve ardından bu dosyayı Google Drive'a yükler. Bu süreç, dosyanın HTTP isteği ile indirilmesini, bellekte ikili bir akış (binary stream) olarak saklanmasını ve ardından Google Drive API kullanılarak bulut depolama alanına yüklenmesini içerir. Fonksiyon, dosyanın indirilmesinden bulut yüklemesine kadar olan işlemi sorunsuz bir şekilde yönetir.

Fonksiyonun temel amacı, verilen bir URL'den ses dosyasını indirip bellekte geçici olarak saklamak ve ardından bu dosyayı Google Drive'a yüklemektir. Bu yöntem, dosyanın yerel diske kaydedilmeden önce doğrudan belleğe indirilmesini sağlayarak I/O işlemlerini ve yerel depolama kullanımını minimize eder.

Fonksiyon, indirme ve yükleme işlemini tamamlamak için aşağıdaki adımları takip eder:

Dosya Adının Çekilmesi: Fonksiyon, sağlanan URL'den dosya adını çıkararak işlemeye başlar. URL, '/' karakteri ile bölünür ve son segment (dosya adı) seçilir. Bu işlem, Python'un split metodu kullanılarak yapılır. **Kod4**

file\_name = url.split("/")[-1]

**Kod4**

HTTP İsteği Gönderme: Fonksiyon, ardından sağlanan URL'ye HTTP GET isteği gönderir ve ses dosyasını alır. requests.get() fonksiyonu kullanılır, bu fonksiyon URL, çerezler (cookies) ve başlıklar (headers) gibi parametreleri alarak web sunucusu ile geçerli bir oturum simüle eder. stream=True parametresi, dosyanın içeriğini doğrudan belleğe akış olarak almamızı sağlar: **Kod5**

response = requests.get(url, cookies=cookies, headers=headers, stream=True)

**Kod5**

Dosyanın Belleğe Yüklenmesi: İstek başarılı olduğunda (status code 200), yanıtın içeriği Python'un BytesIO sınıfı kullanılarak belleğe aktarılır. Bu, dosyanın ikili veri akışı (binary stream) olarak bellekte tutulmasını sağlar: **Kod6**

file\_stream = io.BytesIO(response.content)

**Kod6**

Dosyanın Google Drive'a Yüklenmesi: Dosya belleğe yüklendikten sonra, fonksiyon Google Drive'a yükleme işlemini başlatır. Dosyanın ikili akışı ve dosya adı, upload\_file() fonksiyonuna iletilir ve bu fonksiyon Google Drive API kullanarak dosyayı belirli bir klasöre yükler: **Kod7**

upload\_file(file\_stream, file\_name)

**Kod7**

Hata Yönetimi: Fonksiyon, indirme sırasında karşılaşılabilecek olası hataları yönetmek için hata kontrolü içerir. Örneğin, ağ hataları veya geçersiz yanıtlar durumunda fonksiyon bir hata mesajı basar. Diğer istisnalar da yakalanarak kaydedilir: **Kod8**

except Exception as e:

print(f"İndirme hatası: {e}")

**Kod8**

## Web Tarayıcısını Başlatma ve Giriş Yapma

Selenium kullanılarak tarayıcı başlatılır ve Freesound giriş sayfasına gidilir: **Kod9**

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))

driver.get('https://freesound.org/')

**Kod9**

Selenium'un sağladığı WebDriver ile Chrome tarayıcısı başlatılır.

Login işlemi yapılır ve kullanıcı adı, şifre girilir:**Kod10**

login\_button = WebDriverWait(driver, 10).until(

EC.element\_to\_be\_clickable((By.XPATH, '//button[@data-toggle="login-modal"]'))

)

login\_button.click()

username\_field = driver.find\_element(By.ID, 'id\_username')

password\_field = driver.find\_element(By.ID, 'id\_password')

username\_field.send\_keys('kullanıcı adın')

password\_field.send\_keys('şifren')

password\_field.send\_keys(Keys.RETURN)

**Kod10**

## Sayfa Gezinme ve Ses Linklerini Çekme

Freesound platformunda yer alan ses dosyalarına ait bağlantıları toplamak için kullanılan fonksiyon, sayfa gezintisini otomatikleştirir ve her sayfada yer alan ses dosyalarının URL'lerini çeker. Bu işlem, web scraping (web kazıma) teknikleri kullanarak gerçekleştirilir ve ses dosyalarının hızlı bir şekilde toplanmasını sağlar. Bu fonksiyon, Selenium WebDriver kullanarak dinamik içerikli sayfalarda gezinir ve ses dosyalarına ait bağlantıları çıkarır.

Fonksiyonun temel amacı, Freesound platformundaki sayfalarda gezilerek, belirli kriterlere (örneğin, dosya türü wav olan sesler) uygun ses dosyalarına ait bağlantıları toplamaktır. Bu işlem, belirli sayfa aralıklarıyla yapılır ve her sayfa için ses dosyalarının bağlantıları toplanır.

Fonksiyon, sayfalarda gezinme ve ses dosyalarına ait bağlantıları toplama işlemini şu adımlarla gerçekleştirir:

Sayfa Gezinme: Fonksiyon, for döngüsü içerisinde, Freesound'daki belirli bir sorgu ve filtreyle (örneğin, .wav formatındaki müzik dosyaları) ilgili 350 sayfa boyunca gezinir. Her bir sayfa URL’si, Selenium WebDriver kullanılarak yüklenir: **Kod11**

driver.get(f'https://freesound.org/search/?q=music&f=type%3A%22wav%22&page={page}#sound')

**Kod11**

Bekleme Süresi: Sayfa yüklendikten sonra, sayfanın tamamen yüklenmesi için kısa bir süre beklenir. Bu işlem, sayfanın tüm dinamik öğelerinin yüklenmesini ve sayfa içeriğinin tamamlanmasını sağlar: **Kod12**

time.sleep(2)

**Kod12**

Ses Linklerinin Bulunması: Sayfa yüklendikten sonra, find\_elements() fonksiyonu kullanılarak sayfadaki ses dosyalarına ait bağlantılar (<a> etiketleri) bulunur. Bu işlem, XPATH kullanarak ses bağlantılarını filtreler ve sadece ses dosyasına ait bağlantıları çeker. Ses dosyalarının URL’lerini içeren bağlantılar, belirli sınıflara ve URL desenlerine sahip olmalıdır: **Kod13**

sound\_links = driver.find\_elements(By.XPATH, '//a[contains(@class, "bw-link--black") and starts-with(@href, "/people/")]')

**Kod13**

URL'lerin Çekilmesi: Bulunan bağlantılardan sadece geçerli URL'ler alınır. Bunun için her bir bağlantının href özelliği çekilir ve geçerli (boş olmayan) URL'ler bir listeye eklenir:**Kod14**

sound\_urls = [link.get\_attribute("href") for link in sound\_links if link.get\_attribute("href")]

**Kod14**

Linklerin Kaydedilmesi: Her sayfa için bulunan ses bağlantıları, fonksiyon tarafından toplanarak bir listeye eklenir ve işlem tamamlanır. Böylece, her bir sayfadan toplanan ses linkleri bir araya getirilmiş olur.

## Ses Dosyalarını İndirme ve Yükleme

Freesound platformundaki ses dosyalarına ait URL'ler toplandıktan sonra, her bir ses dosyasının indirilmesi ve bu dosyanın Google Drive'a yüklenmesi işlemi gerçekleştirilir. Bu süreç, Selenium WebDriver kullanılarak her bir ses dosyasının sayfasına gidilmesi, indirme butonunun tıklanması ve dosyanın belirtilen hedefe yüklenmesiyle tamamlanır.

Bu fonksiyon, Freesound platformundaki ses dosyalarının her birini indirip, önceden tanımlanmış bir fonksiyon aracılığıyla bu dosyaları Google Drive'a yüklemeyi amaçlar. Bu işlem, her bir ses dosyasının URL'sinin alınmasını ve bu URL'yi kullanarak dosyanın otomatik olarak indirilmesini içerir.

Fonksiyon, ses dosyalarını indirme ve yükleme işlemini şu adımlarla gerçekleştirir:

Ses Sayfasına Gidilmesi: Fonksiyon, daha önce toplanan ses dosyası URL'lerini içeren sound\_urls listesindeki her bir URL'ye giderek, her ses dosyasının sayfasını yükler. Bu işlem, enumerate() fonksiyonu kullanılarak yapılır. İlk URL atlanır ve kalan URL'ler sırasıyla ziyaret edilir:**Kod15**

for index, sound\_page\_url in enumerate(sound\_urls[1:], start=2):

driver.get(sound\_page\_url)

time.sleep(2)

**Kod15**

İndirme Butonunun Tıklanması: Sayfa yüklendikten sonra, Selenium WebDriver kullanılarak indirme butonunun bulunması sağlanır. Bu buton, sayfada bulunan <a> etiketlerinden, "sound-download-button" sınıfına ve geçerli bir href özelliğine sahip olanlar ile filtrelenir. İndirme butonunun görünüp görünmediği, WebDriverWait fonksiyonu ile kontrol edilir:**Kod16**

download\_button = WebDriverWait(driver, 30).until(

EC.presence\_of\_element\_located(

(By.XPATH, '//a[contains(@class, "sound-download-button") and @href]')

)

**)**

**Kod16**

İndirme URL’sinin Alınması: İndirme butonunun tıklanabilir olduğu tespit edildikten sonra, butonun href özelliği (indirme bağlantısı) alınır. Bu URL, ses dosyasının indirileceği bağlantıyı içerir. Eğer URL geçerli ise, bu URL kullanılarak ses dosyası indirilir: **Kod17**

download\_url = download\_button.get\_attribute("href")

**Kod17**

Dosyanın İndirilmesi ve Yüklenmesi: Alınan indirme URL'si, daha önce tanımlanan download\_file() fonksiyonuna gönderilir. Bu fonksiyon, belirtilen URL'den ses dosyasını indirir ve ardından dosyayı Google Drive'a yükler. İndirme işlemi, HTTP istekleri kullanılarak yapılır ve başarılı bir şekilde indirilen dosya, Google Drive API aracılığıyla belirtilen hedef klasöre yüklenir:**Kod18**

download\_file(download\_url, cookies, headers)

**Kod18**

## Tarayıcıyı Kapatma

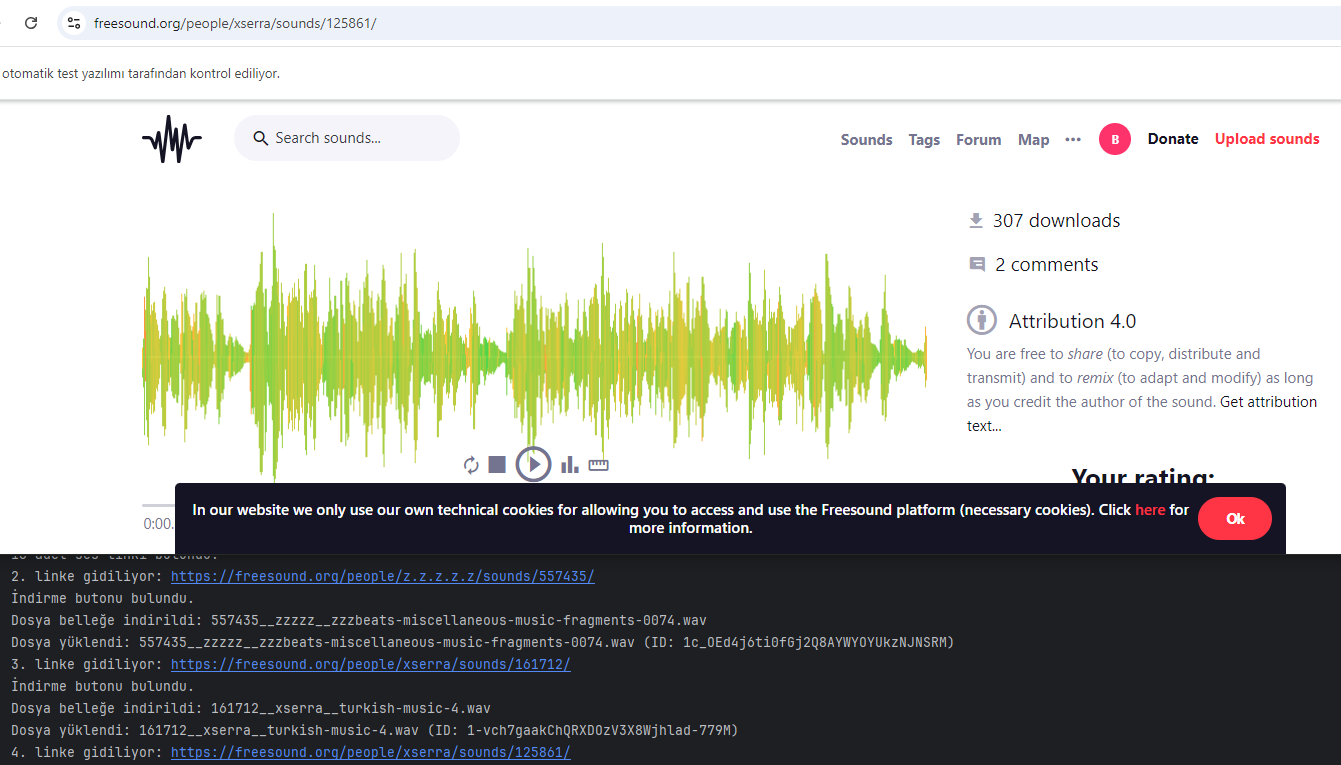
Son olarak, işlem tamamlandığında tarayıcı kapatılır:**Kod19**

driver.quit()

**Kod19**

# Sonuçlar

Proje, ses dosyalarını belirli bir web sayfasından başarılı bir şekilde indirmeyi ve bunları otomatik olarak Google Drive’a yüklemeyi başarmıştır. Ses dosyalarının indirilmesi ve buluta yüklenmesi işlemi, kullanıcının müdahalesine gerek kalmadan otomatik bir şekilde gerçekleşmektedir. Bu süreç, kullanıcıların ses dosyalarını manuel olarak indirmesi ve buluta yüklemesi gereksinimini ortadan kaldırmaktadır.



**Resim1**

# Gelecek Çalışmalar

Projede yapılan geliştirmeler ile birlikte, gelecekte farklı formatlardaki dosyaların da eklenmesi ve diğer bulut depolama servislerine (örneğin, Dropbox, OneDrive) entegrasyon yapılması planlanmaktadır. Ayrıca, ses dosyalarının etiketlenmesi ve daha gelişmiş bir arama fonksiyonu eklenerek proje daha işlevsel hale getirilebilir.

##### kaynaklar

1. https://www.selenium.dev/documentation/en/
2. https://developers.google.com/drive?hl=tr
3. https://docs.python-requests.org/en/latest/
4. https://realpython.com/python-requests/
5. https://realpython.com/beautiful-soup-web-scraper-python/
6. https://pypi.org/project/webdriver-manager/
7. https://docs.python.org/3/
8. https://www.w3schools.com/python/
9. https://stackoverflow.com
10. https://github.com
11. https://github.com/betlet1/sesverisi
12. https://drive.google.com/drive/folders/13t9QvPmNiUe-fcdriNDRfSOX2b2gAv-\_