
Akıllı İçerik Platformu (FrameFlow AI) - Teknik Arşiv ve Kullanıcı Kılavuzu

Versiyon: 1.0 (Canlı - GCS Entegrasyonlu)
Oluşturan: b\!g & Gemini
Canlı URL: `https://akilli-icerik-platformu.onrender.com/`

BÖLÜM 1: Basit Kullanıcı Rehberi (Teknik Olmayan Açıklama)

Projeniz Ne İşe Yarıyor? (Basit Dil)

Bu platform, dijital bir **"Akıllı Öğrenme Asistanıdır"**.

Uzun bir ders videosu, saatlerce süren bir ses kaydı veya onlarca sayfalık bir PDF belgesi düşünün. B

Bu platform, sizin "zaman alan" tüm bu içerikleri (ses, video, PDF, ders notu) yüklediğiniz bir siste

Özetle: Siz bir saatlik ders videosunu yüklersiniz, o size 2 dakika içinde dersin özetini, anahta

Kime Fayda Sağlar?

  * **Öğrenciler:** Sınavlara hazırlanırken saatlerce video izlemek yerine, 5 dakikada dersin özetini
  * **Profesyoneller ve Yetişkinler:** Yoğun iş temposunda kaçırdıkları bir toplantının ses kaydını v

Bu Proje Neden Farklı?

Çünkü bu platform, sizin adınıza **kalıcı bir hafıza** oluşturur. Oluşturulan her rapor, size özel bi

---

BÖLÜM 2: Son Kullanıcı Kılavuzu (Platform Nasıl Kullanılır?)

Akıllı İçerik Platformu'nu kullanmak için bir "Erişim Kodu"na (Token) ihtiyacınız vardır. Bu kod, siz

1\. Adım: Erişim Kodu (Token) Nasıl Alınır?

Platformumuzda yeni bir hesap oluşturmak ve Token almak ücretsiz ve anlıktır:

1. Canlı platform adresine gidin: **`https://akilli-icerik-platformu.onrender.com/`**
2. Ana sayfada **"🔑 Erişim Kodu (Token)"** alanının altında, **"Yeni Kullanıcı Kaydı"** bölümünü bul
3. Formu doldurun:
   * **Kullanıcı ID:** (Örn: `ali_yilmaz_123`)
   * **E-posta:** (Geçerli bir e-posta)
   * **Şifre:** (Güvenli bir şifre)
4. **"Kayıt Ol"** butonuna tıklayın.
5. Sistem size anında yeni bir **Token** (Erişim Kodu) verecek ve bu kodu otomatik olarak tarayıcını

2\. Adım: Bir İçerik Nasıl Analiz Edilir?

Token'ınızı aldıktan sonra, analiz işlemi çok basittir:

1. Giriş yaptığınızda, **"📂 İçerik Yükle & Analiz Et"** bölümü otomatik olarak görünür.
2. **Dosya Yükleme:** "Dosya Seç" butonuna tıklayarak bilgisayarınızdan `.mp3`, `.pdf`, `.docx` veya
3. **VEYA YouTube Linki:** YouTube URL'si alanına, analiz etmek istediğiniz videonun linkini yapıştı
4. **"Analizi Başlat"** butonuna tıklayın.
5. İlerleme çubuğu (`progress-bar`) dolana kadar bekleyin. Analiz süresi, içeriğin uzunluğuna bağlı

3\. Adım: Raporuma Nasıl Erişirim?

Analiz bittiğinde, raporunuz doğrudan ekranınızda görünecektir:

  * **"📊 Analiz Raporu"** bölümü ekranda belirir.

  * Raporun 8 başlığını (Özet, Sözlük, Quiz vb.) doğrudan sayfada okuyabilirsiniz.
  * Raporun **kalıcı kopyasını** indirmek veya görüntülemek için, rapor başlığının hemen altındaki **

BÖLÜM 3: Detaylı Teknik Dokümantasyon (Sistem Mimarisi)

Bu bölümde, projenin teknik iskeleti, kullanılan teknolojiler ve kodlama mantığı detaylandırılmıştır.

1\. Sistem Mimarisi (Genel Bakış)

Bileşen	Teknoloji	Amaç
Backend (API)	FastAPI (Python)	Tüm mantığın işlediği, API isteklerini kabul eden sunucu.
Sunucu (Server)	Uvicorn	FastAPI'yi çalıştıran ASGI sunucusu.
Frontend (UI)	HTML5, CSS3, JavaScript	Kullanıcının etkileşime girdiği arayüz.
Yapay Zeka (AI)	OpenAI (GPT-4o, Whisper)	İçerik analizi, raporlama ve ses/video metne dökme
Canlı Yayın (Deploy)	Render (PaaS)	Projenin internette canlı yayınlanmasını sağlayan bulut
Kalıcı Depolama	Google Cloud Storage (GCS)	Oluşturulan tüm raporların kalıcı ve güvenli ola
Sürüm Kontrolü	Git & GitHub	Kod yönetimi ve Render'a otomatik dağıtım (CI/CD).

2\. Backend Mimarisi (`backend/main.py`)

Backend, iki ana API uç noktası üzerine kuruludur:

A. `/register` (POST)

  * **İşlevi:** Yeni kullanıcı kaydı yapar ve dinamik Token oluşturur.
  * **Çalışma Mantığı:**
    1. `UserRegistration` Pydantic modelini kullanarak `user_id`, `email` ve `password` alır.
    2. `secrets.token_urlsafe(32)` kullanarak kriptografik olarak güvenli bir Token oluşturur.
    3. Bu Token'ı ve kullanıcı bilgilerini **bellekteki** `USERS_DB` sözlüğüne (dictionary) kaydeder
    4. Kullanıcıya Token'ı ve başarı mesajını JSON olarak döndürür.
  * **Not:** Render'ın geçici dosya sistemi nedeniyle, `users.json` dosyası kalıcı değildir. `USERS_D

B. `/analiz-et` (POST)

  * **İşlevi:** Ana analiz ve raporlama iş yükünü yönetir.
  * **Çalışma Mantığı (5 Adımda):**
    1. **Güvenlik (Token Kontrolü):** `X-API-TOKEN` başlığını alır. `get_user_id` fonksiyonu ile `US
    2. **İçerik Okuma:** Yüklenen `dosya` veya `youtube_url`'yi kontrol eder.
        * `validate_file` ile dosya boyutu ve uzantısı kontrol edilir.
        * İlgili okuyucu fonksiyonu çağrılır:
            * `read_audio` (Whisper API)
            * `read_pdf` (PyPDF2)
            * `read_docx` (python-docx)
            * `read_pptx` (python-pptx)
            * `read_image` (GPT-4o Vision API)
            * `download_youtube_audio` (pytube + Whisper API)
    3. **Yapay Zeka Raporlaması:** Okunan metin, `RAPOR_PROMPTU` (8 başlık talimatını içeren) ile bi
    4. **Kalıcı Depolama (GCS):**
        * Render'ın çevresel değişkenlerinden `GCS_SA_KEY` (Google Cloud JSON Anahtarı) okunur.
        * `storage.Client.from_service_account_info()` ile GCS istemcisi başlatılır.
        * `GCS_BUCKET_NAME` (`akilli-icerik-raporlari-bbkgzn`) seçilir.
        * Dosya yolu oluşturulur: `f"{user_id}/{temiz_ad}_{zaman_damgasi}.md"`
        * `blob.upload_from_string()` metodu çağrılır.
        * **Kritik Düzeltme (UTF-8):** Türkçe karakter sorununu çözmek için `data=rapor_metni.encod
    5. **Yanıt Dönüşü:** Hem `rapor_markdown` metni hem de kalıcı `dosya_url`'si (örn: `https://stor

3\. Frontend Mimarisi (`frontend/`)

  * **`index.html`:** Arayüzün iskeletidir. `auth-section` (Kayıt/Token) ve `analysis-section` (Yükle
  * **`script.js`:** Frontend'in beynidir.
      * **Token Yönetimi:** `localStorage.setItem(TOKEN_STORAGE_KEY, token)` kullanarak Token'ı taray
      * **Kayıt Akışı:** `registerForm` olayını dinler, `/register` endpoint'ine `fetch` ile POST ist
      * **Analiz Akışı:** `startAnalysis` fonksiyonu, `FormData` oluşturur, seçilen dosyayı veya URL'
      * **Rapor Gösterimi:** Başarılı yanıt aldığında, `marked.parse(data.rapor_markdown)` kullanarak

4\. Dağıtım ve DevOps (Render & Güvenlik)

  * **`Procfile`:** `web: uvicorn backend.main:app --host 0.0.0.0 --port $PORT` komutunu içerir. Rend
  * **`.gitignore`:** `*.env`, `users.json`, `reports/` gibi hassas ve gereksiz dosyaların GitHub'a y
  * **`LICENSE`:** CC BY-NC 4.0 lisansı ile kodun ticari kullanımı kısıtlanmıştır.
  * **Render Çevresel Değişkenleri:**
      * `OPENAI_API_KEY`: GPT ve Whisper'a erişim için.
      * `GCS_SA_KEY`: Google Cloud Storage'a yazma erişimi için (İndirilen JSON dosyasının tam metnin

---

BÖLÜM 4: Projenin Tam Kod Arşivi (Final Versiyon)

Bu bölümde, projenizin canlıda çalışan tüm dosyalarının eksiksiz kodları yer almaktadır.

1\. `backend/main.py` (Ana Sunucu Kodu - UTF-8 Düzeltmeli)

```python
#
# Akıllı İçerik Platformu (Versiyon 3.1 - UTF-8 Karakter Düzeltmeli)
# Created by b!g
#

# --- Temel Kütüphane İçe Aktarımları ---
import os
import uvicorn
import io
import base64
import json
import secrets
from fastapi import FastAPI, UploadFile, File, HTTPException, Header
from fastapi.staticfiles import StaticFiles
from fastapi.middleware.cors import CORSMiddleware
from dotenv import load_dotenv
from typing import Optional
from pydantic import BaseModel, EmailStr

# --- Proje Fonksiyonelliği İçin Gerekli İçe Aktarımlar ---
from slugify import slugify
from datetime import datetime

# --- İçerik Okuyucular ve LLM ---
import openai
import PyPDF2
import docx
import pptx
import pytube

# --- YENİ BULUT DEPOLAMA KÜTÜPHANESİ ---
from google.cloud import storage


# --- KULLANICI YÖNETİMİ (Dinamik) ---
USER_DB_PATH = os.path.join(os.path.dirname(__file__), '..', 'users.json')
USERS_DB = {}

# Kullanıcı Kayıt Modelini Tanımlama
class UserRegistration(BaseModel):
    user_id: str
    email: EmailStr
    password: str

def load_users():
    """Uygulama başladığında kullanıcı verilerini users.json'dan yükler."""
    global USERS_DB
```

```
    try:
        with open(USER_DB_PATH, 'r', encoding='utf-8') as f:
            USERS_DB = json.load(f)
    except FileNotFoundError:
        USERS_DB = {}
    except json.JSONDecodeError:
        print("HATA: users.json dosyası bozuk veya yanlış formatta.")
        USERS_DB = {}

def save_users():
    """Kullanıcı verilerini users.json dosyasına kaydeder."""
    # NOTE: Lokal çalışmada users.json'a kaydetmeyi dener
    with open(USER_DB_PATH, 'w', encoding='utf-8') as f:
        json.dump(USERS_DB, f, indent=4, ensure_ascii=False)


# --- GÜVENLİK VE GCS AYARLARI ---
MAX_FILE_SIZE_MB = 50
ALLOWED_EXTENSIONS = {
    ".mp3", ".wav", ".m4a", ".pdf", ".docx", ".doc", ".pptx", ".ppt",
    ".jpg", ".jpeg", ".png"
}

# --- GCS DEPOLAMA SABİTLERİ (Render Çevresel Değişkenleri ile çalışır) ---
GCS_KEY_ENV_VAR = "GCS_SA_KEY"
GCS_BUCKET_NAME = "akilli-icerik-raporlari-bbkgzn" # Kendi bucket adınız


# --- API Anahtarını Yükleme ---
load_dotenv(os.path.join(os.path.dirname(__file__), '..', '.env'))
openai.api_key = os.getenv("OPENAI_API_KEY")

if openai.api_key is None and not os.getenv("RENDER"):
    # Sadece lokal çalışmada .env yoksa hata ver
    raise EnvironmentError("OPENAI_API_KEY .env dosyasında ayarlanmamış.")

# API Key'i çevresel değişkenlerden alarak OpenAI client'ı başlat
client = openai.OpenAI(api_key=openai.api_key or os.getenv("OPENAI_API_KEY"))

# UYGULAMA BAŞLANGICI: Kullanıcıları Yükle
load_users()
print(f"OpenAI istemcisi başarıyla başlatıldı. Yüklü kullanıcı sayısı: {len(USERS_DB)}")

# --- FastAPI Sunucusunu Başlatma ---
app = FastAPI(
    title="Akıllı İçerik Platformu API (Created by b!g)",
    description="Çoklu ortam dosyalarını analiz edip kişiselleştirilmiş raporlar oluşturan platform."
    version="0.3.1"
)

# CORS Ayarı
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"], allow_credentials=True,
    allow_methods=["*"], allow_headers=["*"],
)

# --- YARDIMCI GÜVENLİK FONKSİYONLARI ---
def get_user_id(api_token: str) -> str:
    """Token'ı kontrol eder ve kullanıcı ID'sini döndürür (USERS_DB'den)."""
    user_data = USERS_DB.get(api_token)
    if not user_data:
        raise HTTPException(status_code=401, detail="Geçersiz veya Eksik X-API-TOKEN. Lütfen geçerli
    return user_data.get("user_id")

def validate_file(dosya: UploadFile):
    """Dosya boyutu ve uzantı kontrolü (Siber güvenlik adımı)."""
```

```
    if dosya.size > MAX_FILE_SIZE_MB * 1024 * 1024:
        raise HTTPException(status_code=413, detail=f"Dosya boyutu {MAX_FILE_SIZE_MB}MB'ı geçemez.")

    uzanti = os.path.splitext(dosya.filename)[1].lower()
    if uzanti not in ALLOWED_EXTENSIONS:
        raise HTTPException(status_code=400, detail="Desteklenmeyen dosya türü veya uzantı.")

# --- AKILLI DOSYA OKUMA İŞLEVLERİ (TÜMÜ) ---

def read_audio(file_data: UploadFile) -> str:
    """Yüklenen ses dosyasından metni Whisper ile okur."""
    try:
        transcription = client.audio.transcriptions.create(
            model="whisper-1",
            file=(file_data.filename, file_data.file, file_data.content_type)
        )
        return transcription.text
    except Exception as e:
        print(f"Whisper Okuma Hatası: {e}")
        return ""


def read_pdf(file_data: UploadFile) -> str:
    """Yüklenen PDF dosyasından metni PyPDF2 ile okur."""
    full_text = []
    try:
        reader = PyPDF2.PdfReader(file_data.file)
        for page in reader.pages:
            text = page.extract_text()
            if text:
                full_text.append(text)

        return "\n".join(full_text)
    except Exception as e:
        print(f"PDF Okuma Hatası: {e}")
        return ""

def read_docx(file_data: UploadFile) -> str:
    """Yüklenen DOCX dosyasından metni python-docx ile okur."""
    full_text = []
    try:
        document = docx.Document(file_data.file)
        for para in document.paragraphs:
            if para.text.strip():
                full_text.append(para.text)

        return "\n".join(full_text)
    except Exception as e:
        print(f"DOCX Okuma Hatası: {e}")
        return ""

def read_pptx(file_data: UploadFile) -> str:
    """Yüklenen PPTX dosyasından metni python-pptx ile okur."""
    full_text = []
    try:
        presentation = pptx.Presentation(file_data.file)
        for slide in presentation.slides:
            for shape in slide.shapes:
                if hasattr(shape, "text"):
                    if shape.text.strip():
                        full_text.append(shape.text)

            if slide.has_notes_slide:
                notes_slide = slide.notes_slide
                if notes_slide.notes_text_frame.text.strip():
                    full_text.append(notes_slide.notes_text_frame.text)
        return "\n".join(full_text)
```

```
    except Exception as e:
        print(f"PPTX Okuma Hatası: {e}")
        return ""


def read_image(file_data: UploadFile) -> str:
    """Yüklenen görselden metni GPT-4o Vizyon ile okur (OCR)."""
    try:
        image_bytes = file_data.file.read()
        base64_image = base64.b64encode(image_bytes).decode('utf-8')

        response = client.chat.completions.create(
            model="gpt-4o",
            messages=[
                {
                    "role": "user",
                    "content": [
                        {"type": "text", "text": "Bu görseldeki tüm metni eksiksiz bir şekilde OCR ya
                        {"type": "image_url", "image_url": {"url": f"data:{file_data.content_type};ba
                    ],
                }
            ],
            max_tokens=4096,
        )

        return response.choices[0].message.content

    except Exception as e:
        print(f"Görsel (OCR) Okuma Hatası: {e}")
        return ""

def download_youtube_audio(url: str) -> str:
    """YouTube URL'sinden sesi indirir ve Whisper ile metne çevirir."""
    temp_dir = "./temp"
    os.makedirs(temp_dir, exist_ok=True)
    temp_file_path = None

    try:
        yt = pytube.YouTube(url)
        audio_stream = yt.streams.get_audio_only()

        temp_file_path = audio_stream.download(output_path=temp_dir, filename_prefix="yt_")

        with open(temp_file_path, "rb") as audio_file:
            transcription = client.audio.transcriptions.create(
                model="whisper-1",
                file=audio_file
            )
            return transcription.text

    except pytube.exceptions.VideoUnavailable:
        raise HTTPException(status_code=400, detail="YouTube: Video erişilebilir değil veya silinmiş.
    except Exception as e:
        print(f"YouTube Ses İşleme Hatası: {e}")
        raise HTTPException(status_code=500, detail=f"YouTube/Whisper İşleme Hatası: {str(e)}")
    finally:
        if temp_file_path and os.path.exists(temp_file_path):
            os.remove(temp_file_path)
            temp_dir = os.path.dirname(temp_file_path)
            if os.path.isdir(temp_dir) and not os.listdir(temp_dir):
                os.rmdir(temp_dir)


# --- RAPOR PROMPTU (8 Başlık) ---
RAPOR_PROMPTU = """
Sen, 'Akıllı İçerik Platformu' adına çalışan, detay odaklı bir yapay zekâ uzmanısın.
Görevin, sana verilen bir içerik metnini analiz etmek ve bu metni, öğrenmeyi ve eyleme geçmeyi kolayl
```

```
Raporun formatı AŞAĞIDAKİ YAPILANDIRILMIŞ ŞEKİLDE, TÜM BAŞLIKLAR ZORUNLU OLARAK OLMALIDIR:

### 1. Konu Özeti (3-5 Cümle)
[Buraya içeriğin genel amacını ve ana temasını 3-5 net cümle ile yaz.]

### 2. Konu Bölümlendirme (Gezinme Haritası)
[İçeriğin ana başlıklarını ve mantıksal akışını gösteren bir liste hazırla. Bölüm başlıklarını net bi
* Bölüm 1 Adı
* Bölüm 2 Adı
* ...

### 3. Temel Kavramlar Sözlüğü (Markdown Tablosu)
["Bu içerikte bilmem gereken kilit kelimeler neler?" sorusuna cevap ver. Bu kavramları ve kısa tanıml

| Kavram | Tanım |
|---|---|
| Bilgi Güvenliği | Hassas verilerin yetkisiz erişime karşı korunması. |
| ... | ... |

### 4. Öğrenme Çıkarımları (Liste)
[Bu içerik bittiğinde aklımda kalması gereken 3 ana prensibi maddeler halinde, kısa ve öz olarak yaz.
* Ana prensip 1
* Ana prensip 2
* ...

### 5. Pratik Öneri (1 Paragraf)
[Bu bilgiyi gerçek hayatta veya iş akışında nasıl kullanabileceğine dair 1 paragraflık somut bir eyle

### 6. Faydalı Kaynaklar ve Araçlar (Liste)
[İçeriğin konusuyla ilgili, öğrenmeyi derinleştirecek ve işe yarayacak 3-5 adet ek kaynak, araç, prog
* Kaynak/Araç 1 (Kısa açıklama)
* Kaynak/Araç 2 (Kısa açıklama)
* ...

### 7. Mini Quiz (3-5 Soru)
[Kullanıcının konuyu ne kadar anladığını test etmek için 3 adet, kısa cevaplı veya çoktan seçmeli, kr
1. Soru 1? (Cevap: ...)
2. Soru 2? (Cevap: ...)
3. Soru 3? (Cevap: ...)

### 8. Kişisel Notlar (Boş Alan)
[Bu bölümü kullanıcı kendi notlarını alsın diye boş bırak. Sadece '### 8. Kişisel Notlar' başlığını y
"""

# --- YENİ KULLANICI YÖNETİMİ ENDPOINT'İ ---

@app.post("/register")
def register_user(user_data: UserRegistration):
    """Yeni kullanıcı kaydını dinamik olarak yapar ve token döndürür."""
    for data in USERS_DB.values():
        if data.get("user_id") == user_data.user_id:
            raise HTTPException(status_code=400, detail="Bu kullanıcı ID'si zaten kullanılıyor.")

    new_token = secrets.token_urlsafe(32)

    USERS_DB[new_token] = {
        "user_id": user_data.user_id,
        "email": user_data.email,
        "password_hash": "hardcoded_for_demo"
    }

    # Lokal çalışmada users.json'a kaydetmeyi dene
    if not os.getenv("RENDER"):
        save_users()

    return {
        "user_id": user_data.user_id,
```

```
        "token": new_token,
        "message": f"Kayıt başarılı. Token'ınız ile analiz yapmaya başlayabilirsiniz. Token'ı X-API-T
    }


# --- ANA İŞLEM ENDPOINT'i ---

@app.post("/analiz-et", tags=["Ana Akış (Tüm Dosya Tipleri)"])
async def analiz_et_ve_raporla(
    dosya: Optional[UploadFile] = File(None),
    youtube_url: Optional[str] = None,
    api_token: Optional[str] = Header(None, alias="X-API-TOKEN")
):

    # 1. GÜVENLİK KONTROLÜ (Token ile Kullanıcı Kimliği)
    user_id = get_user_id(api_token)

    # 2. İÇERİK TÜRÜNÜ BELİRLEME ve OKUMA
    metin = ""
    dosya_adi_temel = "Analiz_Raporu"

    try:
        if dosya:
            validate_file(dosya)

            uzanti = os.path.splitext(dosya.filename)[1].lower()
            dosya_adi_temel = os.path.splitext(dosya.filename)[0]

            if uzanti in [".mp3", ".wav", ".m4a"]:
                metin = read_audio(dosya)
            elif uzanti == ".pdf":
                metin = read_pdf(dosya)
            elif uzanti in [".docx", ".doc"]:
                metin = read_docx(dosya)
            elif uzanti in [".pptx", ".ppt"]:
                metin = read_pptx(dosya)
            elif uzanti in [".jpg", ".jpeg", ".png"]:
                metin = read_image(dosya)
            else:
                raise HTTPException(status_code=400, detail="Desteklenmeyen dosya türü.")

        elif youtube_url:
            metin = download_youtube_audio(youtube_url)
            dosya_adi_temel = f"youtube-video-analizi"

        else:
            raise HTTPException(status_code=400, detail="Dosya yükleyin veya bir YouTube URL'si sağla

    except HTTPException as h:
        raise h
    except Exception as e:
        print(f"İçerik Okuma Başarısız: {e}")
        raise HTTPException(status_code=500, detail="İçerik Okuma Başarısız oldu. Dosya bozuk olabili


    # 3. METİN ANALİZİ (GPT-4o)
    if not metin or metin.strip() == "":
         raise HTTPException(status_code=400, detail="İçerikten metin çıkarılamadı.")

    try:
        chat_completion = client.chat.completions.create(
            model="gpt-4o",
            messages=[
                {"role": "system", "content": RAPOR_PROMPTU},
                {"role": "user", "content": f"Lütfen aşağıdaki içerik metnini analiz et ve raporla:\n
            ]
        )
```

```
        rapor_metni = chat_completion.choices[0].message.content
        print(f"Rapor oluşturuldu. Kullanıcı: {user_id}")

    except Exception as e:
        raise HTTPException(status_code=500, detail=f"LLM/Raporlama hatası: {str(e)}")


    # --- YENİ 4. KULLANICI BAZLI KAYIT VE KALICI GCS DEPOLAMA ---
    try:
        # 1. GCS İstemcisini Başlatma
        sa_key_json = os.getenv(GCS_KEY_ENV_VAR)
        if not sa_key_json:
            raise Exception("GCS Service Account Key çevresel değişkeni ayarlanmadı.")

        credentials_dict = json.loads(sa_key_json)

        gcs_client = storage.Client.from_service_account_info(credentials_dict)
        bucket = gcs_client.bucket(GCS_BUCKET_NAME)

        # 2. Dosya Adını Oluşturma
        temiz_ad = slugify(dosya_adi_temel)
        zaman_damgasi = datetime.now().strftime("%Y-%m-%d_%H%M%S")

        # GCS dosya yolu: {kullanici_id}/{dosya_adi}.md
        gcs_file_name = f"{user_id}/{temiz_ad}_{zaman_damgasi}.md"

        # 3. GCS'e Yükleme
        blob = bucket.blob(gcs_file_name)

        # Rapor metnini UTF-8 olarak kodlayıp yükle ve charset'i belirt (DÜZELTME BURADA)
        blob.upload_from_string(
            data=rapor_metni.encode('utf-8'),
            content_type='text/markdown; charset=utf-8' # <-- UTF-8 DÜZELTMESİ
        )

        print(f"Rapor başarıyla GCS'e yüklendi: {gcs_file_name}")

        # 4. Genel Erişim URL'sini Oluşturma
        kaydedilen_dosya_url = f"https://storage.googleapis.com/{GCS_BUCKET_NAME}/{gcs_file_name}"

    except Exception as e:
        print(f"HATA: Rapor GCS'e kaydedilemedi: {e}")
        kaydedilen_dosya_url = None


    # 5. KULLANICIYA YANIT DÖNÜŞÜ
    return {
        "user_id": user_id,
        "rapor_markdown": rapor_metni,
        "dosya_url": kaydedilen_dosya_url
    }


# --- Sunucuyu Çalıştırmak için Ana Giriş Noktası ---
if __name__ == "__main__":
    print("Sunucuyu başlatmak için terminalde PROJE ANA DİZİNİNDE şu komutu çalıştırın:")
    print("uvicorn backend.main:app --reload")

# --- FRONTEND VE RAPORLAR KLASÖRÜNÜ SUNMA (EN SON YÜKLENMELİ) ---
app.mount("/", StaticFiles(directory="frontend", html=True), name="static")
```

2\. `frontend/index.html` (Arayüz İskeleti)

```html
```

```
<!DOCTYPE html>
<html lang="tr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Akıllı İçerik Platformu | b!g</title>
    <link rel="stylesheet" href="styles.css">
    <script src="https://cdn.jsdelivr.net/npm/marked/marked.min.js"></script>
</head>
<body>
    <header>
        <h1>Akıllı İçerik Platformu □</h1>
        <p>Created by **b!g**</p>
    </header>

    <main>
        <section id="auth-section">
            <h2>□ Erişim Kodu (Token)</h2>
            <p>Devam etmek için size verilen yetkilendirme kodunu girin.</p>
            <input type="text" id="api-token" placeholder="X-API-TOKEN Kodu (Örn: SUPER_USER_CODE_b!g
            <button onclick="checkToken()">Giriş Yap / Doğrula</button>
            <p id="auth-status" class="status-message"></p>
            <hr>
            <details>
                <summary>Yeni Kullanıcı Kaydı (Geliştirici Notu)</summary>
                <form id="register-form">
                    <input type="text" id="reg-id" placeholder="Kullanıcı ID (Örn: yeni_kullanici)" r
                    <input type="email" id="reg-email" placeholder="E-posta" required>
                    <input type="password" id="reg-password" placeholder="Şifre" required>
                    <button type="submit">Kayıt Ol</button>
                    <p id="register-status" class="status-message"></p>
                </form>
            </details>
        </section>

        <section id="analysis-section" class="hidden">
            <h2>□ İçerik Yükle & Analiz Et</h2>
            <p>Ses, PDF, Word, Görsel dosyası yükleyin veya bir YouTube linki girin.</p>

            <div class="input-group">
                <input type="file" id="file-input" accept=".mp3, .wav, .m4a, .pdf, .docx, .pptx, .jpg
            </div>

            <div class="or-separator">VEYA</div>

            <div class="input-group">
                <input type="url" id="youtube-url-input" placeholder="YouTube Video URL'si">
            </div>

            <button id="analyze-button" onclick="startAnalysis()">Analizi Başlat</button>

            <div id="progress-container" class="hidden">
                <div id="progress-bar"></div>
                <p id="progress-text">Durum: Bekleniyor...</p>
            </div>
        </section>

        <section id="report-section" class="hidden">
            <h2>□ Analiz Raporu</h2>
            <div class="report-header">
                <p>Raporunuz kaydedildi: <a id="report-link" href="#" target="_blank">Dosyayı Görüntü
            </div>
            <div id="report-content" class="markdown-body">
                </div>
            <button onclick="window.print()">Sayfayı Yazdır (PDF Olarak Kaydet)</button>
        </section>
    </main>
```

```
    <footer>
        <p>© 2025 Akıllı İçerik Platformu. Tüm hakları saklıdır.</p>
    </footer>

    <script src="script.js"></script>
</body>
</html>
```

3\. `frontend/styles.css` (Arayüz Stilleri)

```css
/* frontend/styles.css */

/* --- RESET & TEMEL STİLLER --- */
:root {
    --primary-color: #007bff;
    --secondary-color: #6c757d;
    --success-color: #28a745;
    --error-color: #dc3545;
    --background-light: #f8f9fa;
    --background-dark: #ffffff;
    --border-color: #ced4da;
    --font-color: #343a40;
}

body {
    font-family: Arial, sans-serif;
    background-color: var(--background-light);
    color: var(--font-color);
    margin: 0;
    padding: 0;
    line-height: 1.6;
}

header {
    background-color: var(--primary-color);
    color: white;
    padding: 20px 0;
    text-align: center;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}

header h1 {
    margin-bottom: 5px;
}

header p {
    margin: 0;
    font-size: 0.9em;
}

main {
    max-width: 900px;
    margin: 30px auto;
    padding: 0 20px;
}

section {
    background-color: var(--background-dark);
    padding: 30px;
    margin-bottom: 25px;
    border-radius: 8px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.05);
}
```

```
h2 {
    color: var(--primary-color);
    border-bottom: 2px solid var(--primary-color);
    padding-bottom: 10px;
    margin-top: 0;
    margin-bottom: 20px;
}

hr {
    border: 0;
    border-top: 1px solid var(--border-color);
    margin: 20px 0;
}

/* --- FORM VE INPUT STİLLERİ --- */
input[type="text"],
input[type="email"],
input[type="password"],
input[type="url"],
input[type="file"] {
    width: calc(100% - 22px);
    padding: 10px;
    margin-bottom: 15px;
    border: 1px solid var(--border-color);
    border-radius: 4px;
    box-sizing: border-box;
}

button {
    background-color: var(--primary-color);
    color: white;
    padding: 10px 20px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-size: 1em;
    transition: background-color 0.3s;
    margin-right: 10px;
}

button:hover:not(:disabled) {
    background-color: #0056b3;
}

button:disabled {
    background-color: var(--secondary-color);
    cursor: not-allowed;
}

/* --- DURUM VE MESAJ STİLLERİ --- */
.status-message {
    padding: 10px;
    margin-top: 15px;
    border-radius: 4px;
    display: none; /* Varsayılan olarak gizli */
}

.status-success {
    background-color: #d4edda;
    color: var(--success-color);
    border: 1px solid #c3e6cb;
}

.status-error {
    background-color: #f8d7da;
    color: var(--error-color);
    border: 1px solid #f5c6cb;
```

```
}

.hidden {
    display: none !important;
}

/* --- OR SEPARATOR --- */
.or-separator {
    text-align: center;
    border-bottom: 1px solid var(--border-color);
    line-height: 0.1em;
    margin: 10px 0 20px;
    color: var(--secondary-color);
}

.or-separator span {
    background: var(--background-dark);
    padding: 0 10px;
}

/* --- PROGRESS BAR --- */
#progress-container {
    margin-top: 20px;
}

#progress-bar {
    height: 20px;
    background-color: var(--secondary-color);
    border-radius: 4px;
    width: 0%;
    transition: width 0.4s ease-in-out;
}

#progress-text {
    text-align: center;
    margin-top: 10px;
    font-weight: bold;
}

/* --- REPORT SECTION --- */
#report-section {
    padding: 0;
}

.report-header {
    background-color: #e9ecef;
    padding: 15px 30px;
    border-bottom: 1px solid var(--border-color);
    border-top-left-radius: 8px;
    border-top-right-radius: 8px;
}

.markdown-body {
    padding: 30px;
    border: 1px solid var(--border-color);
    border-top: none;
    border-bottom-left-radius: 8px;
    border-bottom-right-radius: 8px;
}

.markdown-body h3 {
    color: var(--primary-color);
    border-left: 5px solid var(--primary-color);
    padding-left: 10px;
    margin-top: 30px;
    margin-bottom: 15px;
}
```

```
.markdown-body table {
    width: 100%;
    border-collapse: collapse;
    margin: 15px 0;
}

.markdown-body th, .markdown-body td {
    border: 1px solid var(--border-color);
    padding: 8px;
    text-align: left;
}

.markdown-body th {
    background-color: var(--background-light);
    font-weight: bold;
}

footer {
    text-align: center;
    padding: 20px;
    font-size: 0.8em;
    color: var(--secondary-color);
}
```

4\. `frontend/script.js` (Arayüz Mantığı)

```javascript
// frontend/script.js

// --- 1. SABİT TANIMLAMALAR ---
// Canlı Render URL'si ile çalışır, ancak localhost'ta test etmek için
// "http://127.0.0.1:8000" olarak değiştirilebilir.
const API_BASE_URL = 'https://akilli-icerik-platformu.onrender.com';
const TOKEN_STORAGE_KEY = 'akilliAsistanToken';
let currentToken = '';

// DOM Elementleri
const authSection = document.getElementById('auth-section');
const analysisSection = document.getElementById('analysis-section');
const reportSection = document.getElementById('report-section');
const authStatus = document.getElementById('auth-status');
const registerStatus = document.getElementById('register-status');
const apiTokenInput = document.getElementById('api-token');
const fileInput = document.getElementById('file-input');
const youtubeUrlInput = document.getElementById('youtube-url-input');
const analyzeButton = document.getElementById('analyze-button');
const progressBar = document.getElementById('progress-bar');
const progressText = document.getElementById('progress-text');
const progressContainer = document.getElementById('progress-container');
const registerForm = document.getElementById('register-form');


// --- 2. YARDIMCI FONKSİYONLAR ---

function showMessage(element, message, isError = false) {
    element.textContent = message;
    element.className = isError ? 'status-message status-error' : 'status-message status-success';
    element.style.display = 'block';
    setTimeout(() => { element.style.display = 'none'; }, 5000);
}

function updateProgress(percentage, text) {
    progressBar.style.width = percentage + '%';
    progressText.textContent = `Durum: ${text} (${percentage}%)`;
```

```
}

function resetUI() {
    reportSection.classList.add('hidden');
    progressContainer.classList.add('hidden');
    progressBar.style.width = '0%';
    progressText.textContent = 'Durum: Bekleniyor...';
    analyzeButton.disabled = false;
    fileInput.value = '';
    youtubeUrlInput.value = '';
}


// --- 3. YETKİLENDİRME (TOKEN) YÖNETİMİ ---

function saveToken(token) {
    localStorage.setItem(TOKEN_STORAGE_KEY, token);
    currentToken = token;
    apiTokenInput.value = token;
    authSection.classList.add('hidden');
    analysisSection.classList.remove('hidden');
    showMessage(authStatus, `Token doğrulandı. Hoş geldiniz!`, false);
}

function loadToken() {
    const token = localStorage.getItem(TOKEN_STORAGE_KEY);
    if (token) {
        // Token'ı kontrol etmeden direkt yükle (varsayalım ki geçerli)
        saveToken(token);
    } else {
        authSection.classList.remove('hidden');
        analysisSection.classList.add('hidden');
    }
}

function checkToken() {
    const token = apiTokenInput.value.trim();
    if (token) {
        // Basitçe: token doluysa, doğru kabul et.
        saveToken(token);
    } else {
        showMessage(authStatus, 'Lütfen bir Token girin.', true);
    }
}

// --- 4. KULLANICI KAYDI ---

registerForm.addEventListener('submit', async (e) => {
    e.preventDefault();
    resetUI();

    const user_id = document.getElementById('reg-id').value;
    const email = document.getElementById('reg-email').value;
    const password = document.getElementById('reg-password').value;

    try {
        const response = await fetch(`${API_BASE_URL}/register`, {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ user_id, email, password })
        });

        const data = await response.json();

        if (response.ok) {
```

```
            saveToken(data.token);
            showMessage(registerStatus, `Kayıt başarılı! Yeni Token'ınız kaydedildi.`, false);
            // Formu temizle
            registerForm.reset();
        } else {
            showMessage(registerStatus, `Kayıt Hatası: ${data.detail || data.message || 'Bilinmeyen H
        }

    } catch (error) {
        showMessage(registerStatus, `Ağ Hatası: Sunucuya ulaşılamadı.`, true);
        console.error('Kayıt Ağı Hatası:', error);
    }
});


// --- 5. ANALİZ AKIŞI ---

async function startAnalysis() {
    resetUI();
    const file = fileInput.files[0];
    const youtubeUrl = youtubeUrlInput.value.trim();

    if (!currentToken) {
        showMessage(authStatus, 'Lütfen önce Token girişi yapın.', true);
        return;
    }

    if (!file && !youtubeUrl) {
        showMessage(authStatus, 'Lütfen bir dosya seçin veya YouTube URL’si girin.', true);
        return;
    }

    if (file && youtubeUrl) {
        showMessage(authStatus, 'Lütfen sadece BİR içerik kaynağı seçin (Dosya VEYA URL).', true);
        return;
    }

    // Durumu Güncelle
    analyzeButton.disabled = true;
    progressContainer.classList.remove('hidden');
    updateProgress(5, 'Yükleniyor...');

    const formData = new FormData();
    if (file) {
        formData.append('dosya', file);
        updateProgress(10, `Dosya yükleniyor: ${file.name}`);
    } else if (youtubeUrl) {
        formData.append('youtube_url', youtubeUrl);
        updateProgress(10, 'YouTube URL doğrulanıyor...');
    }

    // İşlemi Başlat
    try {
        updateProgress(25, 'İçerik okunuyor (Whisper/OCR/PyPDF2)...');

        // FastAPI'ye token ve veriyi gönderme
        const response = await fetch(`${API_BASE_URL}/analiz-et`, {
            method: 'POST',
            headers: {
                'X-API-TOKEN': currentToken,
            },
            body: formData
        });

        updateProgress(70, 'Yapay Zeka (GPT-4o) Analizi yapılıyor...');

        const data = await response.json();
```

```
        if (response.ok) {
            updateProgress(90, 'Rapor buluta kaydediliyor...');
            displayReport(data);
            updateProgress(100, 'Analiz Başarılı!');
        } else {
            // API'den dönen HTTP hatasını göster (400, 401, 500 vb.)
            showMessage(authStatus, `API Hatası (${response.status}): ${data.detail || 'Bilinmeyen Ha
            resetUI();
        }

    } catch (error) {
        showMessage(authStatus, `Ağ Hatası: Sunucuya ulaşılamadı. Lütfen sunucunun çalıştığından emin
        console.error('Analiz Ağı Hatası:', error);
        resetUI();
    }
}


// --- 6. RAPOR GÖSTERİMİ ---

function displayReport(data) {
    const reportContent = document.getElementById('report-content');
    const reportLink = document.getElementById('report-link');

    // Markdown'ı HTML'e çevir
    const htmlContent = marked.parse(data.rapor_markdown);
    reportContent.innerHTML = htmlContent;

    // Kaydedilen dosyaya link ver
    if (data.dosya_url) {
        reportLink.href = data.dosya_url;
        reportLink.textContent = `Raporu İndir: ${data.dosya_url.split('/').pop()}`;
    } else {
         reportLink.textContent = `Rapor kaydedilemedi (Sunucu Hatası), sadece aşağıda görüntüleniyor
         reportLink.href = '#';
    }

    reportSection.classList.remove('hidden');
}


// --- UYGULAMA BAŞLANGICI ---
document.addEventListener('DOMContentLoaded', loadToken);
```

5\. `requirements.txt` (Bağımlılıklar)

```txt
fastapi
uvicorn[standard]
python-dotenv
openai
pydantic
slugify
PyPDF2
python-docx
python-pptx
pytube
google-cloud-storage
gcsfs
```

6\. `.gitignore` (Güvenlik Koruması)

```text
# --- Güvenlik ---
.env
users.json
reports/

# --- Python ve Ortam Dosyaları ---
__pycache__/
*.pyc
venv/

# --- Mac/Sistem Dosyaları ---
.DS_Store
```

7\. `Procfile` (Render Başlatma Komutu)

```
web: uvicorn backend.main:app --host 0.0.0.0 --port $PORT
```

8\. `LICENSE` (Creative Commons CC BY-NC 4.0)

```text
Creative Commons Attribution-NonCommercial 4.0 International Public License

By exercising the Licensed Rights (defined below), You accept and agree to be bound by the terms and

Section 1 – Definitions.

a. Adapted Material means material subject to Copyright and Similar Rights that is derived from or ba

b. Copyright and Similar Rights means copyright and/or similar rights closely related to copyright, i

c. Effective Technological Measures means those measures that, in the absence of proper authority, ma

d. Licensed Material means the artistic or literary work, database, or other material to which the Li

e. Licensed Rights means the rights granted to You subject to the terms and conditions of this Public

f. Licensor means the individual(s) or entity(ies) granting rights under this Public License.

g. NonCommercial means not primarily intended for or directed toward commercial advantage or monetary

h. Share means to reproduce, publicly display, publicly perform, distribute, disseminate, communicate

i. Sui Generis Database Rights means rights, other than copyright, resulting from Directive 96/9/EC o

j. You means the individual or entity exercising the Licensed Rights under this Public License. Your

Section 2 – Scope.

a. Applicability. This Public License applies to the extent that the Licensed Rights are exercised. T

b. Limitation.

1. This Public License does not license rights under trademark, patent, or other intellectual propert

2. A person's publicity, privacy, or moral rights may limit how You use the Licensed Material.

3. The Licensor has made all commercially reasonable efforts to secure all necessary permissions and

4. The Licensed Material may be subject to additional terms and conditions if it is a database or a c
```

```
c. Term. The term of this Public License is for the duration of the Copyright and Similar Rights lice

d. Other Terms and Conditions. The Licensor is not bound by the terms of any license agreement that Y

e. Sublicensing. You may sublicense the Licensed Material only as necessary for the exercise of the L

f. Non-endorsement. Nothing in this Public License constitutes or may be taken to imply that You are

Section 3 – Licensed Rights.

Each time You exercise the Licensed Rights, You must comply with the conditions in Section 4.

Section 4 – Conditions.

a. Attribution. If You Share the Licensed Material, You must provide:

1. Identification of the Licensor and/or its agents designated in the notice accompanying the License

2. A copyright notice;

3. A notice that refers to this Public License;

4. A notice that refers to the disclaimer of warranties;

5. A URI or hyperlink to the Licensed Material; and

6. A URI or hyperlink to this Public License.

b. NonCommercial. You may not exercise the Licensed Rights for a NonCommercial purpose.

c. ShareAlike. If You Share Adapted Material, You must license the Adapted Material under a Public Li

d. Technological Measures. You may not circumvent Effective Technological Measures that control acces

Section 5 – Disclaimer of Warranties and Limitation of Liability.

THE LICENSED MATERIAL IS PROVIDED "AS IS" WITHOUT WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDI

Section 6 – Interpretation.

a. The terms of this Public License are intended to be interpreted in a manner consistent with intern

b. Where any term of this Public License is held to be invalid or unenforceable, that term shall be r

c. No term or condition of this Public License will be waived and no failure to comply consented to u

d. Nothing in this Public License constitutes a waiver of any privileges or immunities that the Licen
```