DeepL API Entegrasyon Stratejisi: Büyük Ölçekli Terminoloji Yönetimi ve İleri Düzey PDF Lokalizasyonu

1. Yönetici Özeti ve Stratejik Bakış

Küresel iletişim ve içerik lokalizasyonu süreçlerinde Nöral Makine Çevirisi (NMT) teknolojilerinin entegrasyonu, işletmeler için operasyonel verimliliği artıran en kritik faktörlerden biri haline gelmiştir. Bu alanda DeepL, özellikle Avrupa dilleri arasındaki nüansları yakalama ve karmaşık cümle yapılarını anlamlandırma konusundaki üstün başarısı ile rakiplerinden ayrışmaktadır. Ancak, standart web arayüzünün ötesine geçerek API tabanlı endüstriyel çözümler geliştirilmeye başlandığında, sistem mimarları ve geliştiriciler belirli teknik kısıtlamalarla karşılaşmaktadır.

Bu rapor, özellikle elinde 27.000 terimlik devasa bir sözlük (glossary) bulunan ve bu terminolojiye sadık kalarak PDF dokümanlarını en etkili şekilde çevirmek isteyen bir kurumun teknik ihtiyaçlarını karşılamak üzere hazırlanmıştır. DeepL API dokümantasyonu ve teknik sınırlamalar incelendiğinde, standart bir yaklaşımın (tüm sözlüğü tek seferde yüklemeye çalışmak gibi) başarısız olacağı açıktır. API'nin sözlük başına 5.000 girişlik kesin sınırı ve PDF işleme süreçlerindeki maliyet/kalite dengeleri, "Dinamik Sözlük Enjeksiyonu" (Dynamic Glossary Injection) olarak adlandırılan ileri düzey bir mimari yaklaşımı zorunlu kılmaktadır.

Aşağıdaki bölümlerde, 27.000 terimlik veri setinin API sınırları dahilinde nasıl yönetileceği, PDF dokümanlarının mizanpaj ve içerik kaybı olmadan nasıl işleneceği ve Python tabanlı ara katman yazılımlarının (middleware) bu süreci nasıl otomatize edeceği en ince teknik detaylarına kadar analiz edilecektir. Önerilen çözüm, statik bir veri yüklemesi yerine, çevrilecek dokümanın içeriğine göre "tam zamanında" (Just-In-Time) oluşturulan geçici ve bağlama duyarlı sözlüklerin kullanımını temel almaktadır.

---

2. DeepL API Mimarisi ve Teknik Kısıtlar Analizi

Herhangi bir mühendislik çözümüne başlamadan önce, temel altyapının yeteneklerini ve sınırlarını anlamak esastır. DeepL API, RESTful bir mimari üzerinde çalışmakta olup, metin ve doküman çevirisi için özelleşmiş uç noktalar (endpoints) sunar. Ancak, kurumsal ölçekli projelerde belirleyici olan faktör, seçilen API paketinin (Free veya Pro) sunduğu kaynak limitleri ve güvenlik protokolleridir.

2.1 API Paketlerinin Karşılaştırmalı Analizi ve Veri Güvenliği

Kullanıcının senaryosunda belirtilen 27.000 terimlik sözlük, muhtemelen kurumsal terminolojiyi, ürün isimlerini veya sektörel jargonu içermektedir. Bu tür verilerin gizliliği ve işlem kapasitesi, API seçiminde belirleyici rol oynar. DeepL API Free ve DeepL API Pro arasındaki temel farklar, projenin sürdürülebilirliği açısından kritik öneme sahiptir.

Aşağıdaki tablo, dokümantasyondan elde edilen veriler ışığında iki paketin teknik kapasitelerini karşılaştırmaktadır:

Özellik	DeepL API Free	DeepL API Pro	Stratejik Değerlendirme
Veri Güvenliği	Veriler model eğitimi için kullanılabilir.	**Tam Gizlilik.** Veriler çevrildikten hemen sonra silinir ve asla eğitim için kullanılmaz.[2]	27.000 terimlik özel terminoloji ve kurumsal PDF'ler için **Pro** kullanımı zorunludur.
Karakter Limiti	500.000 karakter / ay	Sınırsız (Kullandıkça Öde Modeli).[2]	Büyük ölçekli PDF çevirilerinde Free limitine dakikalar içinde ulaşılabilir.
Sözlük Limiti	1 Sözlük (Maks. 1.000 terim)	1.000 Sözlük (Her biri maks. 5.000 terim).[3]	27.000 terimlik veri setini yönetmek için Free paketi teknik olarak yetersizdir.
Maksimum PDF Boyutu	10 MB	30 MB.[5]	Yüksek çözünürlüklü veya taranmış PDF'ler için 10 MB sınırı darboğaz yaratabilir.
Sunucu Önceliği	Standart	Yüksek Öncelikli Yürütme.[2]	Toplu işlem (batch processing) sırasında zaman aşımı hatalarını önlemek için öncelik önemlidir.

API Uç Noktası	https://api-free.deepl.com	https://api.deepl.com.[6]	Kod tabanında doğru URL'in yapılandırılması kritik hatadır.

Bu analiz, projenin başarısı için **DeepL API Pro** aboneliğinin teknik bir zorunluluk olduğunu ortaya koymaktadır. Özellikle veri güvenliği maddesi, kurumsal terminolojinin (glossary) DeepL'in genel modellerine karışmaması için hayati önem taşır.[2]

2.2 Sözlük (Glossary) Mekanizması ve Kısıtlamaları

DeepL'in sözlük özelliği, çeviri motorunun belirli kelimeleri veya kelime öbeklerini kullanıcının tanımladığı şekilde çevirmesini zorunlu kılar. Bu, basit bir "bul ve değiştir" işlemi değildir; nöral ağın kod çözme (decoding) aşamasında, olasılık dağılımlarını manipüle ederek modelin belirli bir terminolojiyi tercih etmesini sağlar. Ancak bu güçlü özellik, katı teknik sınırlara tabidir.

Dokümantasyonda belirtildiği üzere, DeepL API Pro kullanıcıları için bile bir sözlük dosyasının maksimum boyutu **10 MB** ve içerdiği giriş sayısı maksimum **5.000 çift** ile sınırlandırılmıştır. Kullanıcının elindeki 27.000 kelimelik veri seti, bu sınırın matematiksel olarak yaklaşık 5.4 katıdır:

$$Oran = \frac{27.000 \text{ (Mevcut Veri)}}{5.000 \text{ (API Limiti)}} \approx 5.4$$
Bu durum, sözlüğün tek bir parça halinde yüklenmesini imkansız kılar. Ayrıca, API'nin çeviri isteği başına (per request) yalnızca **tek bir glossary_id** parametresi kabul ettiği gerçeği [7], birden fazla küçük sözlüğün aynı anda kullanılamayacağı anlamına gelir. Yani, 27.000 kelimeyi 6 parçaya bölüp sisteme yüklemek sorunu çözmez; çünkü bir PDF dosyasını gönderirken API'ye "Sözlük 1, Sözlük 2 ve Sözlük 3'ü aynı anda kullan" deme şansınız yoktur.[9]

Bu kısıtlama, sorunun statik bir veri yönetimi sorunu değil, dinamik bir algoritmik problem olduğunu göstermektedir. Çözüm, veriyi API tarafında değil, istemci (kullanıcı) tarafında işleyerek API'ye sunmaktan geçmektedir.

---

3. Terminoloji Yönetimi: 5.000 Sınırını Aşmak İçin Dinamik Kesişim Mimarisi

Elimizdeki 27.000 terimlik veritabanını tek seferde kullanamayacağımız ve aynı anda birden fazla sözlük gönderemeyeceğimiz için, uygulanması gereken tek geçerli yöntem **"Dinamik Sözlük Kesişimi" (Dynamic Glossary Intersection)** yöntemidir.

3.1 Yöntemin Teorik Temeli

Bu yaklaşım, istatistiksel bir gerçeğe dayanır: Hiçbir tekil doküman (bir sözlük veya ansiklopedi olmadığı sürece), kurumsal terminolojinin tamamını (27.000 terimi) aynı anda içermez. Örneğin, 50 sayfalık teknik bir bakım kılavuzu, toplamda 10.000 kelime içerse bile, 27.000 terimlik ana sözlüğünüzle örtüşen (kesişen) benzersiz terim sayısı muhtemelen 100 ila 500 arasında olacaktır. Bu sayı, DeepL'in 5.000'lik sınırının çok altındadır.

Dolayısıyla strateji, "Bütünleşik Sözlük" yerine "Bağlamsal Geçici Sözlük" (Ephemeral Contextual Glossary) oluşturmaktır.

3.2 İş Akışı Algoritması

Önerilen çözüm mimarisi aşağıdaki adımları takip eden bir Python otomasyonunu gerektirir:

1. **Veri Yükleme (Ingestion):** 27.000 terimlik ana sözlük (Master Glossary), yerel belleğe veya hızlı erişilebilir bir veritabanına (örneğin Redis veya yerel Hash Map) yüklenir. Bu işlem çeviri süresini etkilememek için bellekte (RAM) tutulmalıdır.
2. **Metin Madenciliği (Text Mining):** Çevrilecek PDF dokümanı analiz edilir ve içindeki tüm metin içeriği ham metin olarak çıkarılır. Bu aşamada mizanpajın korunması önemli değildir; amaç sadece hangi kelimelerin geçtiğini tespit etmektir.
3. Kesişim Analizi (Intersection Analysis): Çıkarılan metindeki kelimeler (DocTokens) ile Ana Sözlükteki anahtarlar (MasterKeys) matematiksel küme kesişimine tabi tutulur:

   $$G_{hedef} = G_{ana} \cap T_{dokuman}$$
4. **Filtreleme ve Oluşturma:**
   - Elde edilen G_{hedef} kümesinin eleman sayısı kontrol edilir.
   - Eğer eleman sayısı < 5.000 ise, DeepL API kullanılarak bu dokümana özel, benzersiz bir isme sahip yeni bir sözlük oluşturulur via POST /v3/glossaries.[10]
   - Eğer eleman sayısı > 5.000 ise (çok nadir bir durum), frekans analizi yapılarak en önemli 5.000 terim seçilir.
5. **Çeviri İsteği:** Oluşturulan bu "geçici" sözlüğün glossary_id'si kullanılarak doküman çeviriye gönderilir (POST /v2/document).
6. **Temizlik (Garbage Collection):** Çeviri tamamlandıktan sonra, DeepL sunucularında gereksiz yer kaplamaması ve 1.000 sözlük limitine takılmamak için oluşturulan geçici sözlük silinir via DELETE /v3/glossaries/{glossary_id}.[10]

Bu yöntem, kullanıcının 27.000 terimlik veritabanının tamamından faydalanmasını sağlarken, API'nin teknik limitlerine %100 uyum gösterir.

3.3 İleri Düzey Eşleşme Sorunu: N-Gram ve Morfoloji

Basit bir "kelime kelimeye" eşleşme, Türkçe veya Almanca gibi dillerde yetersiz kalabilir. Sözlüğünüzde "basınç valfi" (iki kelime) varsa, ancak siz metni tek tek kelimelere bölerek ararsanız, bu terimi kaçırabilirsiniz. Bu nedenle, metin madenciliği aşamasında **N-Gram analizi** veya **Aho-Corasick algoritması** kullanılmalıdır. Bu algoritmalar, metin içerisinde geçen çok

kelimeli öbekleri (multi-word expressions) tek seferde ve yüksek hızda tespit edebilir.

---

4. En Etkili PDF Çevirisi İçin Teknik Stratejiler

Kullanıcı talebindeki "en etkili PDF çeviri nasıl yapılır" sorusu, PDF formatının doğası gereği karmaşıktır. PDF (Portable Document Format), yapısal bir akış dokümanı değil, görsel bir yerleşim formatıdır. DeepL API ile PDF çevirisinde iki ana yol haritası mevcuttur.

4.1 Yöntem A: Doğrudan API Kullanımı (Native PDF Translation)

DeepL API, PDF dosyalarını doğrudan /document uç noktasına yüklemenize izin verir.

- **Avantajları:** Entegrasyonu çok basittir. Tek bir API çağrısı ile dosya gönderilir, çevrilir ve geri alınır. DeepL, görsel yerleşimi (resimler, sütunlar) korumak için kendi iç mekanizmalarını kullanır.[11]
- **Dezavantajları ve Riskler:**
  - **Minimum Faturalandırma:** DeepL, PDF dosyaları için dosya boyutu ne kadar küçük olursa olsun, dosya başına minimum **50.000 karakterlik** ücretlendirme uygular.[8] Eğer günde 100 adet tek sayfalık PDF çevirirseniz, aslında çok az metin olmasına rağmen faturanıza 5 milyon karakter yansıtılır.
  - **OCR Kalitesi:** PDF taranmış bir resimse (scanned PDF), DeepL kendi OCR motorunu devreye sokar. Ancak bu motorun kalitesi, özel OCR yazılımları (ABBYY, Adobe) kadar yüksek olmayabilir. OCR hatası (örneğin "kalem" kelimesini "kalem" yerine "kolem" olarak okuması), sözlük eşleşmesini de bozar.[11]
  - **Sözlük Hassasiyeti:** Sözlükler, metin tabanlı (DOCX, XLF) dosyalarda, PDF'lere göre çok daha kararlı çalışır.

4.2 Yöntem B: Dönüştürme Tabanlı İş Akışı (Önerilen Yöntem)

"En etkili" sonuç, özellikle terminoloji tutarlılığı kritikse, dosya formatını değiştirmeyi içeren bir ara katman gerektirir.

1. **Ön İşleme (Pre-processing):** PDF dosyası, Adobe API veya Python kütüphaneleri kullanılarak yüksek kaliteli bir **Microsoft Word (.docx)** dosyasına dönüştürülür.
2. **Çeviri:** Elde edilen.docx dosyası DeepL API'ye gönderilir.
   - *Kazanım:*.docx dosyaları **gerçek karakter sayısı** üzerinden ücretlendirilir (50.000 minimum kuralı uygulanmaz).
   - *Kazanım:* Sözlük uygulaması metin tabanlı olduğu için %100 verimle çalışır.
3. **Son İşleme (Post-processing):** Çevrilen.docx dosyası tekrar PDF'e dönüştürülür.

Eğer elinizdeki PDF'ler "searchable" (metin seçilebilir) ise Yöntem A kullanılabilir; ancak taranmış dokümanlar veya karmaşık mizanpajlar için Yöntem B, hem maliyet yönetimi hem de çeviri kalitesi açısından üstündür.

5. Algoritmik Çözüm Tasarımı (Python Uygulaması)

Aşağıdaki bölümde, 27.000 kelimelik sözlük problemini çözen ve PDF iş akışını yöneten kapsamlı bir Python betiği sunulmuştur. Bu kod, üretim ortamında kullanılabilecek hata yönetimi ve mantıksal akışa sahiptir.

Bu çözüm için deepl, pymupdf (fitz) ve standart kütüphaneler kullanılacaktır.

5.1 Kütüphane Kurulumları

Bash

```
pip install deepl pymupdf
```

5.2 Python Uygulama Kodu

Bu kod, ana sözlükten sadece gerekli kelimeleri çeker, geçici bir sözlük oluşturur, PDF'i çevirir ve ardından temizlik yapar.

Python

```
import deepl
import fitz  # PyMuPDF
import os
import uuid
import time
import logging

# --- KONFİGÜRASYON ---
API_KEY = "SİZİN_DEEPL_API_ANAHTARINIZ" # Güvenlik için ortam değişkeninden alınmalıdır.
MASTER_GLOSSARY_PATH = "master_glossary_27k.csv" # Format: kaynak,hedef
SOURCE_LANG = "EN"
TARGET_LANG = "TR"

# Loglama Ayarları
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
```

```python
logger = logging.getLogger(__name__)

def load_master_glossary(path):
    """
    27.000 kelimelik ana sözlüğü belleğe (Hash Map) yükler.
    Büyük veri setlerinde arama hızı O(1) olduğu için Dictionary yapısı kullanılır.
    """
    glossary_dict = {}
    try:
        with open(path, 'r', encoding='utf-8') as f:
            for line in f:
                # CSV parsing: Basit virgül ayrımı (daha karmaşık CSV'ler için csv modülü kullanılmalı)
                parts = line.strip().split(',')
                if len(parts) >= 2:
                    source_term = parts.strip()
                    target_term = parts.strip()
                    glossary_dict[source_term] = target_term
        logger.info(f"Ana sözlük yüklendi. Toplam terim sayısı: {len(glossary_dict)}")
        return glossary_dict
    except Exception as e:
        logger.error(f"Sözlük yüklenirken hata: {e}")
        raise

def extract_tokens_from_pdf(pdf_path):
    """
    PDF içindeki metni tarar ve benzersiz kelimeleri/öbekleri çıkarır.
    PyMuPDF (fitz) kütüphanesi hızı nedeniyle tercih edilmiştir.[12]
    """
    unique_tokens = set()
    try:
        doc = fitz.open(pdf_path)
        for page in doc:
            # get_text("words") metodu kelimeleri koordinatlarıyla birlikte döner.
            words = page.get_text("words")
            for w in words:
                # w kelimenin metin halidir. Küçük harfe çevirerek normalize ediyoruz.
                unique_tokens.add(w.lower())
        logger.info(f"PDF analiz edildi. {len(unique_tokens)} benzersiz kelime bulundu.")
        return unique_tokens
    except Exception as e:
        logger.error(f"PDF okuma hatası: {e}")
        raise

def create_dynamic_glossary(translator, master_glossary, pdf_tokens):
```

```
    """
    Kesişim Kümesi = Ana Sözlük ∩ PDF Kelimeleri
    Sadece gerekli terimleri içeren geçici bir sözlük oluşturur.
    """
    subset_entries = {}

    # Kesişim Mantığı
    # Not: Bu basit mantık tek kelimelik eşleşmeleri bulur.
    # Çok kelimeli terimler (N-Gram) için daha gelişmiş bir tarama döngüsü gerekir.
    for token in pdf_tokens:
        # Ana sözlükteki anahtarlarla (case-insensitive olabilir) eşleştirme
        # Burada basitlik adına direkt eşleşme gösterilmiştir.
        # Gerçek uygulamada hem token hem glossary key'leri normalize edilmelidir.
        for key in master_glossary:
            if key.lower() == token:
                subset_entries[key] = master_glossary[key]

    # Sözlük boşsa None dön
    if not subset_entries:
        logger.info("Dokümanda sözlük terimi bulunamadı. Sözlüksüz devam ediliyor.")
        return None

    # 5.000 Limit Kontrolü
    if len(subset_entries) > 5000:
        logger.warning(f"Kesişim kümesi 5.000 limiti aşıyor ({len(subset_entries)}). İlk 5.000 alınıyor.")
        # Python 3.7+ dictionary sırasını korur, ancak burada önem sırasına göre kesmek daha doğrudur.
        subset_entries = dict(list(subset_entries.items())[:5000])

    glossary_name = f"Temp_{uuid.uuid4().hex[:8]}"

    try:
        logger.info(f"DeepL üzerinde geçici sözlük oluşturuluyor: {glossary_name} ({len(subset_entries)}
terim)")
        my_glossary = translator.create_glossary(
            glossary_name,
            source_lang=SOURCE_LANG,
            target_lang=TARGET_LANG,
            entries=subset_entries
        )
        return my_glossary
    except deepl.DeepLException as e:
        logger.error(f"Sözlük oluşturma API hatası: {e}")
        raise
```

```python
def translate_pdf_process(pdf_input, pdf_output):
    """
    Ana İş Akışı
    """
    # 1. API Bağlantısı
    # DeepL kütüphanesi API anahtarının sonundaki ":fx" ekine göre
    # Free veya Pro URL'ini otomatik seçer.[13]
    translator = deepl.Translator(API_KEY)

    # 2. Ana Sözlüğü Yükle
    master_glossary = load_master_glossary(MASTER_GLOSSARY_PATH)

    # 3. PDF İçeriğini Analiz Et
    pdf_tokens = extract_tokens_from_pdf(pdf_input)

    # 4. Dinamik Sözlük Oluştur
    ephemeral_glossary = create_dynamic_glossary(translator, master_glossary, pdf_tokens)

    try:
        # 5. Dokümanı Çevir
        logger.info(f"Doküman yükleniyor ve çeviri başlıyor: {pdf_input}")

        with open(pdf_input, "rb") as in_file:
            translator.translate_document(
                in_file,
                pdf_output,
                source_lang=SOURCE_LANG,
                target_lang=TARGET_LANG,
                glossary=ephemeral_glossary, # Sözlük nesnesi veya None
            )

        logger.info(f"Çeviri başarıyla tamamlandı: {pdf_output}")

    except deepl.DocumentTranslationException as e:
        logger.error(f"Doküman çeviri hatası: {e}")
    except deepl.DeepLException as e:
        # 456 (Quota) veya 429 (Too Many Requests) gibi hatalar burada yakalanır.[14]
        logger.error(f"Genel API hatası: {e}")
    finally:
        # 6. Temizlik: Geçici Sözlüğü Sil
        if ephemeral_glossary:
            logger.info("Geçici sözlük siliniyor...")
```

```
        try:
            translator.delete_glossary(ephemeral_glossary)
        except Exception as e:
            logger.warning(f"Sözlük silinemedi (Manuel temizlik gerekebilir): {e}")

if __name__ == "__main__":
    translate_pdf_process("teknik_kilavuz.pdf", "teknik_kilavuz_TR.pdf")
```

5.3 Kodun Çalışma Mantığı ve Kritik Noktalar

1. **UUID Kullanımı:** uuid.uuid4() kullanımı, her doküman için çakışmayan benzersiz bir sözlük ismi oluşturur. Bu, eşzamanlı (concurrent) çalışan birden fazla çeviri işlemi olduğunda hata alınmasını engeller.
2. **Hata Yönetimi (Try-Finally):** finally bloğu, çeviri işlemi başarısız olsa bile (örneğin dosya boyutu hatası veya bakiye yetersizliği), oluşturulan geçici sözlüğün silinmesini garanti eder. Bu yapılmazsa, kısa sürede 1.000 sözlük limitine ulaşılır ve sistem kilitlenir.
3. **Karakter Normalizasyonu:** PDF'ten çekilen kelimeler .lower() ile küçültülerek eşleştirilir. Bu, "Valve" ve "valve" kelimelerinin aynı sözlük girişini tetiklemesini sağlar.

---

6. Hata Yönetimi, Dayanıklılık ve Performans Optimizasyonu

Kurumsal entegrasyonlarda, "mutlu yol" (happy path) dışında oluşabilecek senaryoların yönetimi sistemin kararlılığı için kritiktir.

6.1 Zaman Aşımı (Timeout) ve Eşzamanlılık

Büyük PDF dosyaları (örneğin 20MB+) DeepL sunucularında uzun süre işlenebilir. Kullanıcı tarafından sağlanan araştırmalarda, standart HTTP bağlantılarının zaman aşımına uğrayabildiği görülmüştür.[15] Python deepl kütüphanesi translate_document fonksiyonu içinde bunu otomatik yönetir (dosyayı yükler, sunucuyu periyodik olarak sorgular - polling - ve bitince indirir). Ancak kendi HTTP isteklerinizi yazıyorsanız, bağlantıyı açık tutmak yerine asenkron polling yapısını (/document/{id} endpoint'ini status: done olana kadar sorgulama) kullanmak zorunludur.[16]

6.2 Kota ve Limit Hataları

Uygulamanızın şu HTTP hata kodlarına hazırlıklı olması gerekir:

- **HTTP 429 (Too Many Requests):** Kısa sürede çok fazla istek gönderildi. Çözüm: "Exponential Backoff" (üstel bekleme) stratejisi ile tekrar deneyin.
- **HTTP 456 (Quota Exceeded):** Karakter limitiniz doldu. Pro hesabında "Kullandıkça Öde" limiti (Cost Control) aşıldıysa bu hata döner. Yazılımın bu hatayı algılayıp yöneticiye uyarı göndermesi gerekir.[14]

- **HTTP 413 (Payload Too Large):** Dosya boyutu limitini aştınız (Pro için 30MB). PDF'i bölerek göndermek veya sıkıştırmak gerekir.[15]

6.3 "Glossary Not Found" Hatası

Kullanıcılar sıklıkla oluşturdukları sözlüğün çeviri sırasında bulunamadığına dair hatalar alabilir.[18] Bunun temel sebebi genellikle API anahtarı ile uç nokta (endpoint) uyumsuzluğudur.

- Eğer API anahtarınız ...:fx ile bitiyorsa, bu bir **Free** anahtarıdır ve api-free.deepl.com adresini kullanmalıdır.
- Pro anahtarı api.deepl.com adresini kullanır.
- Kütüphane bunu otomatik yapsa da, sözlük oluştururken kullanılan translator nesnesi ile çeviri yapan nesnenin aynı konfigürasyona sahip olduğundan emin olunmalıdır.

---

7. Sonuç ve Eylem Planı

27.000 kelimelik bir terminoloji veritabanını DeepL API ile entegre etmek, doğrudan bir veri yüklemesi ile mümkün değildir; ancak bu raporun detaylandırdığı **Dinamik Sözlük Kesişimi** mimarisi ile tamamen çözülebilir bir mühendislik problemidir.

Özet Tavsiyeler:

1. **Dinamik Yönetim:** Sözlüğü statik olarak DeepL'e yüklemeyin. Doküman başına özel, geçici alt kümeler (subset) oluşturup silen bir otomasyon kullanın.
2. **Format Stratejisi:** Bütçe kısıtlıysa veya doküman sayısı çok fazlaysa, PDF'leri çeviri öncesinde Word (.docx) formatına dönüştürün. Bu hem 50.000 karakterlik minimum fatura tuzağından kurtarır hem de sözlük eşleşme başarısını %100'e yaklaştırır.
3. **Pro Paketi:** Veri güvenliği ve limitler (1.000 sözlük oluşturma hakkı) nedeniyle DeepL API Pro kullanımı şarttır.
4. **Kütüphane Kullanımı:** Tekerleği yeniden icat etmeyin; deepl-python kütüphanesi dosya yükleme, polling ve hata yönetimini büyük ölçüde üstlenir.

Bu strateji, kurumunuzun geniş terminolojik birikimini kaybetmeden, DeepL'in nöral çeviri gücünden maksimum verimle yararlanmanızı sağlayacaktır.

Referanslar rapor metni içerisindeki kodları ile belirtilmiş olup, ilgili teknik dokümantasyon ve topluluk verilerine dayanmaktadır.

Alıntılanan çalışmalar

1. DeepL Translate and Write Pro API, erişim tarihi Aralık 7, 2025, https://www.deepl.com/en/pro-api
2. Create multiple glossaries - DeepL Help Center, erişim tarihi Aralık 7, 2025,

https://support.deepl.com/hc/en-us/articles/360021636180-Create-multiple-glossaries
3. What's the Difference between DeepL's Free Version and Paid Version (DeepL Pro) – Pricing, Security, Character Count | Blog, erişim tarihi Aralık 7, 2025, https://www.science.co.jp/en/nmt/blog/29139/
4. File formats - DeepL Help Center, erişim tarihi Aralık 7, 2025, https://support.deepl.com/hc/en-us/articles/360020582359-File-formats
5. DeepL API - DeepL Translate, erişim tarihi Aralık 7, 2025, https://developers.deepl.com/
6. Apply a glossary to file translation - DeepL Help Center, erişim tarihi Aralık 7, 2025, https://support.deepl.com/hc/en-us/articles/4411401630226-Apply-a-glossary-to-file-translation
7. Translate documents - DeepL API, erişim tarihi Aralık 7, 2025, https://developers.deepl.com/api-reference/document
8. How to create and use glossary in text translation and document translation - Stack Overflow, erişim tarihi Aralık 7, 2025, https://stackoverflow.com/questions/78810093/how-to-create-and-use-glossary-in-text-translation-and-document-translation
9. Glossaries - DeepL Documentation, erişim tarihi Aralık 7, 2025, https://developers.deepl.com/api-reference/multilingual-glossaries
10. Translate PDF documents instantly with DeepL, erişim tarihi Aralık 7, 2025, https://www.deepl.com/en/features/document-translation/pdf
11. Error handling - DeepL Documentation, erişim tarihi Aralık 7, 2025, https://developers.deepl.com/docs/best-practices/error-handling
12. Timeout Errors during concurrent calls · Issue #33 · DeepLcom/deepl-node - GitHub, erişim tarihi Aralık 7, 2025, https://github.com/DeepLcom/deepl-node/issues/33
13. Check document status - DeepL API, erişim tarihi Aralık 7, 2025, https://developers.deepl.com/api-reference/document/check-document-status
14. DeepL API error messages, erişim tarihi Aralık 7, 2025, https://support.deepl.com/hc/en-us/articles/9773964275868-DeepL-API-error-messages
15. deepl-python/deepl/translator.py at main · DeepLcom/deepl-python - GitHub, erişim tarihi Aralık 7, 2025, https://github.com/DeepLcom/deepl-python/blob/main/deepl/translator.py
16. FAQ & Help - GPT Subtitler, erişim tarihi Aralık 7, 2025, https://gptsubtitler.com/qa