**Proje Modüler Yapı Şeması (Metin Tabanlı)**

Bu şematik, dp.py modülünüzün iç yapısını, dış bağımlılıklarını ve kapsamlı test paketinin (tests/ dizini) bu modülle nasıl etkileşim kurduğunu metin tabanlı bir formatta sunmaktadır.

**1. Harici Ortam & Servisler (Girdiler)**

* **Ortam Değişkenleri:** Sistem seviyesinde tanımlanmış değişkenler (örn. DB\_FILE\_PATH).
* **.env Dosyaları:** .env, .env.local, .env.dev gibi uygulama konfigürasyon dosyaları (şu an için doğrudan kullanılmasa da, gelecekteki genişletmeler için bir not).
* **Harici Servisler:**
  + Redis (Mocked)
  + FAISS (Vektör Arama)

**2. dp.py (Veritabanı Yönetim Modülü)** Bu, uygulamanın tüm veritabanı işlemlerini yöneten ana Python dosyasıdır.

* **AsyncDatabaseManager Sınıfı:**
  + **Konfigürasyon Alanları:**
    - db\_file (Veritabanı dosya yolu)
    - pool\_size (Bağlantı havuzu boyutu)
    - timeout (İşlem zaman aşımı)
    - schema\_version (Veritabanı şema versiyonu)
  + **Metotlar:**
    - \_\_init\_\_(): Sınıfın başlatılması.
    - \_create\_connection(): Yeni bir SQLite bağlantısı oluşturur (WAL modu aktif).
    - \_init\_pool(): Bağlantı havuzunu başlatır ve migration'ları çalıştırır.
    - \_get\_connection(): Havuzdan asenkron olarak bir bağlantı alır (round-robin).
    - \_run\_migrations(): Veritabanı şemasını günceller.
    - \_migrate(): Versiyon bazlı şema geçişlerini yönetir.
    - transaction(): Asenkron işlem (transaction) yöneticisi (@contextlib.asynccontextmanager).
    - close(): Bağlantı havuzundaki tüm bağlantıları kapatır.
    - save\_profile(): Kullanıcı profili kaydeder/günceller.
    - load\_profile(): Kullanıcı profili yükler.
    - update\_profile\_preferences(): Kullanıcı tercihlerini günceller.
    - add\_to\_history(): Kullanıcı geçmişine etkileşim ekler.
    - get\_history(): Kullanıcı geçmişini getirir (limit ve zaman filtreli).
    - save\_memory(): Vektör gömme (embedding) ile hafıza kaydı ekler/günceller.
    - load\_memories(): Hafıza kayıtlarını getirir.
    - delete\_profile(): Kullanıcı profilini ve ilişkili tüm verileri siler (atomik işlem).
    - delete\_memory(): Belirli bir hafıza kaydını siler.
  + **Yardımcı Fonksiyonlar ve Entegrasyonlar:**
    - async\_retry(): Üstel geri çekilme (exponential backoff) ile asenkron retry dekoratörü.
    - serialize\_embedding(): Embedding listesini byte'a dönüştürür.
    - deserialize\_embedding(): Byte'tan embedding listesine dönüştürür.
    - logger: Loglama için standart Python logging modülü.

**3. tests/ (Test Paketi)** Bu dizin, dp.py modülünün farklı yönlerini test eden Python dosyalarını içerir.

* **conftest.py:**
  + Ortak pytest fixture'ları:
    - db: Her test için geçici bir SQLite veritabanı.
    - index: FAISS vektör dizini.
    - redis\_client: Temiz bir Redis bağlantısı.
    - mock\_redis: Redis'i mock'lamak için.
* **test\_profile.py:** Kullanıcı profili kaydetme, yükleme, güncelleme ve eşzamanlı güncellemelerle ilgili testler.
* **test\_history.py:** Kullanıcı geçmişi ekleme ve yükleme testleri.
* **test\_embedding.py:** Vektör gömme verisi kaydetme, yükleme ve hatalı veri işleme testleri.
* **test\_delete.py:** Kullanıcı profili ve hafıza kayıtlarının silinmesi (özellikle atomik cascade silme) testleri.
* **test\_utility.py:** async\_retry dekoratörü, zaman aşımı yönetimi ve mock Redis entegrasyonu testleri.
* **test\_faiss\_and\_hypothesis.py:** FAISS entegrasyonu ve hypothesis ile rastgele veri üretimi kullanarak profil kaydetme/yükleme testleri.

**4. pyproject.toml (Proje Yapılandırması - Önerilen)**

* Bağımlılık yönetimi ve proje meta verileri için modern standart.

**5. README.md**

* Proje açıklaması, kurulum, kullanım, test çalıştırma talimatları ve katkıda bulunma yönergeleri.