**PROJE AÇIKLAMASI**

---

**PROJE AMAÇLARI**

Bu proje 4 ana amaca hizmet eder:

1. **EEG verilerini simüle etmek** (çünkü elimizde fiziksel cihaz yok)
2. **Bu verilerden zihin durumunu anlamak** (stres/odak/uykusuzluk)
3. **Kullanıcıya önerilerde bulunmak** (frekans bazlı müzik, egzersizler)
4. **Her şeyi web üzerinden görselleştirmek** (grafikler, gerçek zamanlı akış)

---

**DOSYALARIN AMAÇLARI**

**modules/eeg_simulator.py**

Gerçekçi EEG verisi üretir. 4 farklı zihin durumu modunu (rahat/odaklı/stresli/uykulu) simüle edebilir.

**modules/signal_processor.py**

Ham EEG örneklerini alır, 2 saniyelik pencerelere böler ve her dalga bandının güç değerini hesaplar.

**modules/mental_state_classifier.py**

Dalga güçlerine bakarak zihin durumunu hesaplar. Kural tabanlı mantık kullanır (ML yerine matematik).

**modules/recommendation_engine.py**

Zihin durumuna göre kişiselleştirilmiş öneriler üretir. Frekans bazlı müzik ve egzersiz önerileri verir.

**app.py**

Flask web sunucusu. Arka planda EEG verisi üretir, WebSocket ile frontend'e gönderir, JSON kayıt yapar.

**templates/index.html**

Kullanıcı arayüzü. Grafikleri gösterir, butonlarla kontrol sağlar, WebSocket ile backend'den veri alır.

---

**SİSTEM AKIŞI**

**1. app.py başlar**
**└> Arka plan thread'i EEG verisi üretmeye başlar**

**2. Kullanıcı tarayıcıda açar (localhost:5000)**
**└> WebSocket bağlantısı kurulur**

**3. "Başlat" butonuna tıklar**
**└> is_streaming = True olur**

**4. Her 0.25 saniyede:**
**└> Simulator 64 örnek üretir**
**└> Buffer'a eklenir**
**└> 512 örnek dolunca (2 saniye):**
**├─> Processor analiz yapar**
**├─> Classifier zihin durumunu hesaplar**
**├─> Engine öneri üretir**
**└> WebSocket ile frontend'e gönderilir**

**5. Frontend veriyi alır:**
**└> Metrikleri günceller**
**└> Grafikleri çizer**
**└> Önerileri gösterir**

**6. "Kaydet" butonuna tıklanınca:**
**└> Tüm session verisi JSON'a yazılır**

**MODÜL TESTLERI**

Her modül bağımsız çalışır ve test edilebilir:

```
# Simulator test
python modules/eeg_simulator.py
# Çıktı: Her mod için örnek EEG değerleri

# Processor test
python modules/signal_processor.py
# Çıktı: Band güçleri ve oranlar

# Classifier test
python modules/mental_state_classifier.py
# Çıktı: Her mod için stres/odak/uykusuzluk değerleri

# Engine test
python modules/recommendation_engine.py
# Çıktı: Her senaryo için öneriler
```

**FRONTEND YAPISI**

HTML 3 ana bölümden oluşur:

1. **Kontrol Paneli**: Başlat/durdur, mod değiştir, kaydet butonları
2. **Sol Kolon**: Stres/odak/uykusuzluk metrikleri + Öneriler
3. **Sağ Kolon**: 2 grafik (dalga güçleri + zihin durumu geçmişi)

JavaScript kısmı:

- **Socket.IO**: Backend ile WebSocket bağlantısı

- **Chart.js**: Grafikler için
- **Update fonksiyonları**: Gelen veriyi ekrana yansıtır

---

**ÖNEMLİ NOKTALAR**

**Threading**: app.py'de arka plan thread'i sürekli veri üretir, ana thread Flask'ı çalıştırır.

**Buffer Yönetimi**: Son 256 örnek tutulur (overlap için), bu sayede pencereler kesintisiz devam eder.

**WebSocket vs HTTP**: Veri akışı için WebSocket, kaydetme gibi tek seferlik işler için HTTP kullanılır.

**Kural Tabanlı Sistem**: ML yerine matematiksel kurallar var (ileride ML eklenebilir).

---

**BAŞLATMA SIRASI**

```
1. python app.py
2. Tarayıcıda http://localhost:5000
3. "Başlat" butonuna bas
4. Mod seç (😌 😰 🎯 😴)
5. Grafikleri izle
6. "Kaydet" ile JSON'a yaz
```

---

### ** 📊 VERİ AKIŞI**

```
EEGSimulator → SignalProcessor → MentalStateClassifier → RecommendationEngine
   |          |              |                |
 (Ham veri)  (Band güçleri)   (Stres/Odak/Uyku)      (Öneriler)
   |          |              |                |
   └──────────┴──────────────┴────────────────┘
                      |
                  WebSocket
                      |
                  Frontend
```

**ÖĞRENECEKLERİN**

1. **Python OOP**: Dataclass, Enum, modüler yapı
2. **Numpy/Scipy**: Sinyal işleme, matematiksel hesaplamalar
3. **Flask**: Web framework, route'lar, API
4. **WebSocket**: Gerçek zamanlı iletişim, Socket.IO
5. **Threading**: Çok threaded Python uygulaması
6. **Frontend**: HTML/CSS/JavaScript, Chart.js

7. **Domain Modeling**: İş mantığını koddan ayırma
8. **JSON**: Veri kayıt ve okuma

---

**NOTLAR**

- **Simülatör modu değişikliği**: Frontend'den mod seçilince current_mode değişir, yeni üretilen veriler o moda göre olur.
- **2 saniyelik pencere**: Çünkü EEG analizinde en az 1-2 saniye veri gerekir.
- **256 Hz örnekleme**: Gerçek EEG cihazlarında standart değer.
- **JSON kayıt**: Her veri noktası 2 saniye, 100 veri noktası = 200 saniye = 3.3 dakika.