T.C.
SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜ

Ders: Ağ Programlama

ÖDEV _RAPORU

Ödevin Konusu: 1-Soket programlama (TCP veya UDP temelli) ile kişilerin mesajlaşmasına olanak veren güvenli bir uygulama geliştermesi

Grub Üyeleri :B201210604- Soureya Rozzi Malli,

B201210605-Mahi Abdulhakim Mohammed,

Muhammed Kreek- B211210582

SAKARYA

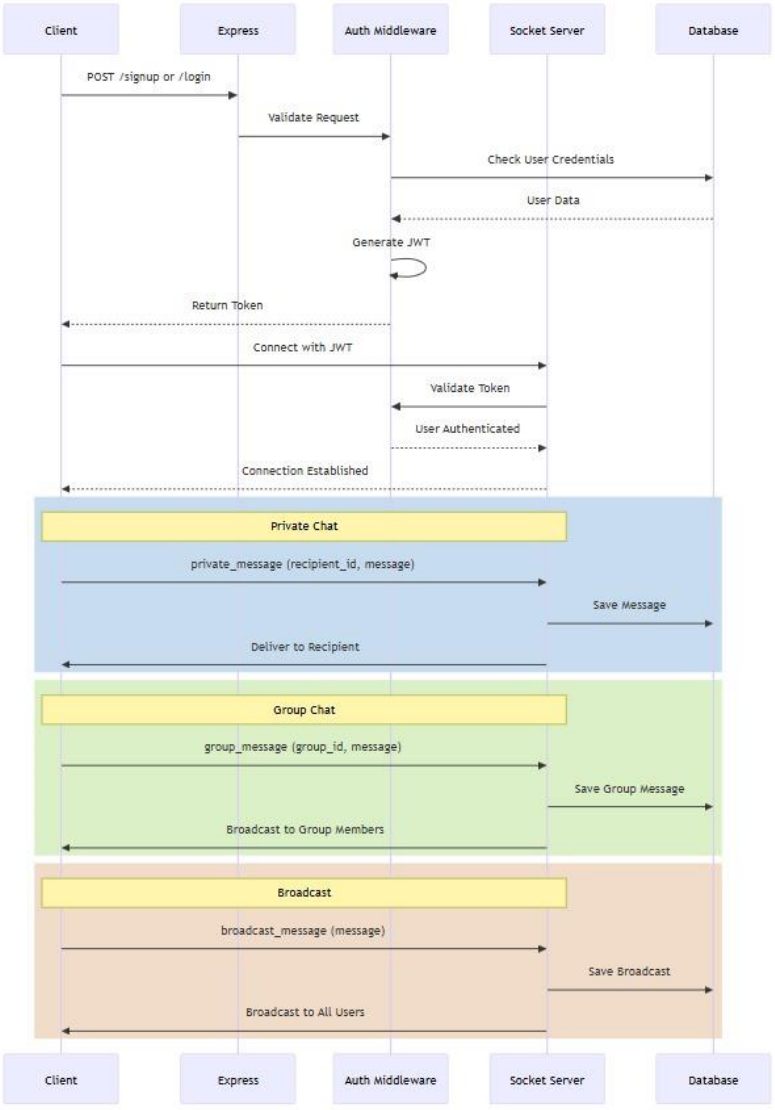
Aralık, 2024

1. Proje Tanımı

Bu proje, TCP protokolü tabanlı bir soket programlama uygulaması olarak tasarlanıp, gerçek zamanlı mesajlaşma sistemi olarak hayata geçirilmiştir. Amaç, kullanıcılar arasında güvenli ve etkili bir iletişim sağlamaktır. Özellikler şu şekildedir:

1. Sunucuya bağlı tüm kullanıcılara mesaj gönderimi.
2. Belirli bir grup kullanıcıya mesaj iletimi.
3. Tek bir kullanıcıya mesaj gönderme olanağı.
4. Mesajların uçtan uca şifrelenmesi ve hashlenerek veritabanında saklanması.
5. Çevrimdışı kullanıcılar için mesaj kaydı ve çevrimiçi olduklarında teslimat.

2.UML Diyagramı



3. Backend Tasarımı ve İncelemesi

a. Veritabanı Tasarımı

Kullanıcı Modeli (User)

- **Alanlar:**
  - email: Kullanıcı e-posta adresi (unique).
  - fullName: Kullanıcı tam adı.
  - password: Hashlenmiş kullanıcı şifresi.
  - profilePic: Profil resmi URL'si.
  - publicKey ve privateKey: RSA şifreleme için anahtarlar.

Mesaj Modeli (Message)

- **Alanlar:**
  - senderId: Mesajı gönderen kullanıcı ID'si.
  - receiverId: Mesajı alan kullanıcı ID'si.
  - encryptedText: Şifrelenmiş mesaj metni.
  - isEncrypted: Mesajın şifreleme durumu (boolean).
  - image: Mesaja eklenmiş görselin URL'si (opsiyonel).

b. Uçtan Uca Şifreleme

Mesajların şifrelenmesi ve çözülmesi için RSA algoritması kullanılmıştır. Bu, mesaj içeriğinin yalnızca hedeflenen alıcı tarafından okunabilmesini sağlar.

- **Şifreleme Mekanizması:** encryptMessage fonksiyonu, mesajı kullanıcının public key'iyle şifreler.
- **Çözme Mekanizması:** decryptMessage fonksiyonu, mesajı private key'iyle çözer.

c. API Tasarımı

- **Authentication:**
  - Kullanıcı kimlik doğrulama JWT (Özet Web Token) tabanlı yapılmıştır.
  - JWT, oturum bilgisini saklamak için çerezlerde kullanılmıştır (httpOnly).
- **Mesaj Gönderme:**
  - Mesajın göndericiden alıcıya ulaştırılması.
  - Çevrimdışı mesajların veritabanında saklanması.
- **Hata Yönetimi:**
  - Hatalar hem istemciye hem de sunucu loglarına kaydedilir.

d. Socket.IO Entegrasyonu

- **Amaç:** Gerçek zamanlı iletişim sağlamak.

- **Online Kullanıcıların Takibi:**
  - userSocketMap kullanılarak her kullanıcının socket ID'si saklanır.
  - Bağlantı kesildiğinde haritadan silinir.

e. Güvenlik Uygulamaları

- **RSA Anahtarları:**
  - Her kullanıcı için bir public ve private key çiftinin oluşturulması.
- **JWT:**
  - Token'ların süresi ve kullanım alanları detaylı kontrol edilir.

4. Frontend Tasarımı ve Uyumluluğu

a. React.js ve Routing

- **Yönlendirme:**
  - react-router-dom kullanılarak dinamik sayfa yönlendirmeleri sağlanır.
  - **Route Koruması:** Doğrulanmamış kullanıcılar, login sayfasına yönlendirilir.

b. Kimlik Doğrulama ve State Yönetimi

- Kullanıcı oturum bilgileri, useAuthStore ile merkezi bir state yönetimi sisteminde tutulur.
- **Oturum Kontrolü:**
  - Kullanıcı doğrulanana kadar bir yükleme ekranı ("Loader") görüntülenir.

c. UI Bileşenleri

- **Navbar:** Kullanıcının oturum durumuna bağlı olarak dinamik çalışır.
- **Login ve Signup Formları:**
  - Geçerli formatta olmayan e-posta adresleri reddedilir.
  - Şifre uzunluğu ve karmaşıklığı kontrol edilir.

d. Mesajlaşma Sistemi

- **Mesaj Listesi:**
  - Seçilen kullanıcıyla olan tüm mesajlar tarih sırasıyla listelenir.
- **Mesaj Gönderimi:**
  - Kullanıcının durumuna göre mesaj gönderme alanı aktif veya pasif olur.

e. Profil ve Tema Seçenekleri

- **Profil Sayfası:** Kullanıcı profil resmi, isim ve e-posta bilgileri güncellenebilir.

- **Tema Seçimi:** Çeşitli görsel temalar arasından tercih yapılabilir.

5. Kurulum ve Çalıştırma

a. Backend

- **Bağımlıklar:**
  - bcryptjs, jsonwebtoken, mongoose, socket.io vb. paketler.
  - Geliştirme ortamında nodemon kullanılabilir.
- **Ortam Değişkenleri:**
  - MONGODB_URL: MongoDB bağlantı URL'si.
  - JWT_SECRET: JWT için gizli anahtar.
  - CLIENT_URL: Frontend istemcisi URL’si.

Çalıştırma Komutları:

```
npm install
npm run dev
```

b. Frontend

- **Bağımlıklar:**
  - react-router-dom, react-hot-toast, tailwindcss vb.
- **Kurulum:**

```
npm install
npm start
```

6. Geliştirme Önerileri

a. Backend

- **Performans Optimizasyonu:**
  - Mesaj sorguları için indeksleme kullanılabilir.
  - Sık mesajlaşma yapan kullanıcı grupları için cache sistemi uygulanabilir.
- **Hata Yönetimi:**
  - Merkezi bir hata izleme ve raporlama sistemi entegre edilebilir.

b. Frontend

- **Bildirimler:**

    - Gerçek zamanlı mesaj bildirimleri eklenebilir.
- **Mesaj Arama:**
    - Kullanıcıların eski mesajları kolayca bulması için arama özelliği eklenebilir.

c. Güvenlik

- **HTTPS:** Sunucu ve istemci arasındaki tüm trafiğin şifrelenmesi sağlanabilir.
- **Rate Limiting:**
    - Brute force saldırılarına karşı API endpointlerinde rate limiting uygulanabilir.

7.Özet

Bu proje, gerçek zamanlı mesajlaşma ve uçtan uca şifreleme alanında kapsamı bir çözüm sunmaktadır. Socket.IO entegrasyonu ve RSA algoritmasıyla desteklenen bu sistem, güvenli ve hızlı bir mesajlaşma altyapısı sunar. Geliştirme önerileriyle daha fazla iyileştirme yapılabilir.