Dağıtık Sistemler Proje Raporu

Alperen Şahin
15401696

Protokol

Komut	Parametre	İleti Komutu	Parametre	Açıklama
NIC	rumuz:pin	WEL	rumuz	Kullanıcı kaydı/giriş
PCH	oldPin:newPin	OKP		Pin değişikliği
NRM	oda adı	OKR	oda adı	Yeni oda yaratma
RLS	-	RLS	oda adı:oda adı:oda adı...	Odaları listeleme
RIN	oda adı	OKI	oda adı	Odaya girme
GNL	oda adı : mesaj	OGM	oda adı:rumuz:message	Odaya mesaj atma
PRV	rumuz : mesaj	OPM	rumuz : message	Özel mesaj atma
BAN	oda adı : rumuz	OBN	oda adı : rumuz	Odadan Kullanıcıyı Banlama
RUT	oda adı	BYR	oda adı	Odadan çıkma
RMV	oda adı	BYR	oda adı	Odayı Silme
KCK	oda adı : rumuz	OCK	oda adı : rumuz	Kullanıcıyı odadan atma
ULS	oda adı	ULS	rumuz:rumuz:rumuz	Odadaki kullanıcıları listeleme
MLS	-	MLS	oda adı:oda adı:...	İçinde bulunan odaları listeleme
MAD	rumuz : oda adı	MAD	rumuz:oda adı	Kullanıcıyı admin yapma
QUI		BYE		Sunucudan çıkış

Hata Mesajları

Komut	Parametre	Açıklama
ERR	eolError	Komut sonu işareti gelmedi hatası
ERR	invalidInstruction	Tanınmayan Komut Hatası
ERR	unauthenticatedUserError	Kullanıcı girişi yapılmadan işlem yapıldı hatası
ERR	authenticationDeniedUserHasAlreadyLogIn	Sunucuda giriş yapmış kullanıcının bir daha giriş isteği atması hatası
ERR	invalidPIN	Kullanıcı girişinde hatalı pin girişi
ERR	pinChangeRequestDeniedOldPinHasNotMatched	Pin değişikliğinde eski pinin yanlış girilmesi hatası
ERR	roomNameIsAlreadyExist	Açılacak oda adının hali hazırda kullanımda olma
ERR	permissionDeniedBannedFromRoom	Banlanmadan dolayı yetki dışı kalma hatası
ERR	userNotFound	Kullanıcı Bulunamadı hatası
ERR	roomNotFound	Oda bulunamadı hatası
ERR	permissionDeniedUserIsACreator	Oda yaratıcısı olmamaktan dolayı alınan yetki hatası

Eksikler

- isterlerin hepsi gerçekleştirilmiştir.

Bonuslar

- **Sunucu verilerinin database üzerinde tutulması :** sunucu database olarak Google Firebase Realtime database kullanmaktadır. Realtime database obje temelli bir veri saklaması yapar bu sayede uygulama üzerinde yaratılan nesneler adaptörler yardımı ile birebir database'e yazılır ya da okunur.
- **Kullanıcı banlama :** Konuşma odası objesinin içerisinde banlanmış kullanıcıların listeleri tutulur işlemler bu listelerle denetlenir.

- **Kullanıcı odaları verilerinin databasede tutulması :** Yukarıda bahsedilen nesne tabanlı database altyapı sayesinde, konuşma odalarındaki değişiklik de anlık olarak firebase üzerine aktarılabilmekte.
- **Yöneticiler diğer kullanıcıları yönetici yapabilsin :** Oda nesnesi içerisinde admin listesi tutularak yapılan işlemlerde yetki takibi yapılabiliyor.

Eklemeler

- **Database Thread :** Database işlemlerini gerçekleştirmek için ayrı bir thread açıp, odada yapılan değişiklikleri anlık olarak database'e aktarılabiliyor.
- **Veri güncelleme:** Sunucu nesnelerinin üzerindeki her işlem sonrası, okuma dahil, sunucu nesnesi değiştiğimi sorunu bu nesnelerin içeriğinin sha-256 ile doğrulanması ile gerçekleşiyor. Sha sonuçları farklı ise değişiklikler database'e eşleniyor.
- **Creator :** odayı kura kişi yaratıcı olarak seçiliyor, bu kullanıcı diğer adminler tarafından odadan atılamıyor, yaratıcı dışında kullanıcılar odaları silemiyor.
- **Online/Offline :** Kullanıcının aktif olup olma durumu kullanıcı nesnesi üzerindeki state değişkeninde tutuluyor, bu sayede tek bir liste ile işlemler gerçekleştirilebiliyor.
- **End Of Line Flag :** Client kodumu yanlış yazdığım için eklemek zorunda kaldığım bir özellik, tek mesaj da iki tane komut geliyordu. Bunu engellemek için gönderilen her satırın sonunda endOdLine bayrağını bekliyorum.
- **Connection Checker :** Ayrı bir thread olup her 1 dakikada kullanıcıların bağlantı durumunu kontrol ediyor. Kopanları OFFLINE olarak işaretliyor.
- **Adaptor Methods / To Object Methods :** Nesnelerin database üzerine direkt aktarılması için dict/json yapısına dönüştürülmesi gerekiyordu. python nesnelerini dict formatına çeviren ve database'den okunan dict formatını nesnelere çeviren Python obje tasarımını ekledim.

Yeniden Tasarım

Projeyi gerçekleştirirken tasarımdan kaynaklı bir problemle karşılaşmadım. Bu nedenle aynı tasarımı kullanırdım.

Yerli Mesajlaşma Uygulaması

Güvenlik perspektifiyle bir mesajlaşma uygulaması/sunucusu geliştirmemiz gerekseydi. Öncelikli yapacağımız işlerden birisi özel ve genel mesajların uçtan uca şifrelenmesini sağlamak olurdu. Bunun için de public key olarak telefon numarası kullanabilirdik. Bu sayede mesajlaşan taraflar bir bilgi aktarımına gerek kalmadan mesajlarını şifreleyebilirlerdi.

Kullanıcı girişi için bizim geliştirdiğimiz sunucuda pin verisini hash algoritmasına sokmadan direkt sunucu ve database üzerinde tutuyoruz. Bu durumda hem geliştiriciler kullanıcılarının şifresini öğrenebilir, hemde araya sızan 3. bir kişi bu bilgiyi kolaylıkla elde edilebilir. Bu durumu önlemek adına kullanıcı şifreleri client kodu üzerinden hashlenerek, sunucu ve database üzerine aktarılmalı. Bu hash kodu 2. bir hash algoritması ile işlenip database üzerine kayıt olmalıdır. Bu sayede 3. taraf herhangi bir kişi kullanıcının gerçekte yazdığı şifreyi okuyamaz.

Sunucuya aktarılacak girdilerin türleri ve değerleri, sunucu içerisinde özel olarak kontrol edilmeli ve injection saldırılarına karşı önlem alınmalı.

Kullanıcı ve oda bilgileri sunucu üzerinde değil database üzerinde tutulmalı, bu sayede server miktarı çoğaltılabilir olası ddos atakları ve aşırı yüklenmelerde sistem birden fazla düğüm üzerine dağıtılarak performans düşüşünün önüne geçilir.