CSE 344 Sistem Programlama
Bahar 2018-19, Final Proje Raporu

Bu projede client-server uygulaması kullanarak client'lardaki dosyaları server tarafından kontrol ettik. Server, client'lardaki dosyaları kendi klasörüne alıyor. Burada her client için bir klasör oluşturuyor. Client ların dosyaları karışmaması için. Farklı clientlar aynı klasör ismiyle açılmışlarsa, içerisindeki dosyaları serverda aynı klasörde bulunur. Daha sonra hangi client çıkış yaparsa yani hangi client da ctrl-c gelirse, o client'ın dosyasına diğer clientdaki dosyalar da server tarafından ekleniyor.

Client:

Client dosyasını çalıştırmak için dosyaya argüman olarak bir dosya yolu, soket için ip adresi ve port numarası vermemiz gerekiyor.

Client dosyamıza servera bağlanıyoruz. Server ile iletişim kurmak için iki adet struct bulunuyor. Bunlardan birincisi her file için, içinde filename ve file içindeki bilgileri tutan bir struct. Diğeri ise fileları tutan, silinen dosya bilgilerini tutan ve sinyal yakalandığını iletmek için gerekli olan bir struct bulunuyor. Soket ile haberleşmek için ikinci struct'ı kullanıyoruz.

Client'da ilk olarak server'a ip adresi ve port numarası ile bağlantı (connect) kuruyoruz. Sonrasında gelen dosya yolunun içindeki dosyaları iletişim kuracağımız struct'ımızın içerisine kayıt ediyoruz. Dosyada değişiklik olup olmadığını, ekleme çıkarma olup olmadığını sürekli kontrol ediyoruz. Eğer herhangi bir değişiklik varsa bunu servera bildirmemiz gerekiyor.

Client çalışırken ctrl-c (sigint) gelirse ilk olarak bu sinyali block yapıyoruz. Çünkü o sırada client da herhangi bir işlem yapılıyor olabilir. Bunun bitmesini bekliyoruz. Block kalktıktan sonra clientdan çıkış yapıyoruz. Eğer başka bir client aynı dosya ismi ile ama farklı bir yol ile servera bağlanmış ise serverda bu iki clientın dosyaları beraber tutulduğu için, hangi clientdan çıkış yapılıyorsa o clientın gelen dosya yoluna diğer clientdaki dosyalar da yazılır. Yani çıkış yapılan client da çıktıktan sonra diğer clientın dosyalarıda bulunur.

Server:

Server ın çalışması için yine bir dosya yolu verilir, sonra kaç tane thread oluşrulması isteniyorsa o kadar threadpoolsize ve soketin port numarası argüman olarak girilir.

Serverda ilk olarak soket oluşturulur, binding yapılır ve clientların bağlanması beklenir (listen). Gelen clientlar bağlandıktan sonra clientların soketidlerini queue da saklıyorum. Threadlere teker teker veriyorum. Her bir client bağlandıktan sonra o clienta bir thread verilir. Thread fonksiyonunda file'a yazmada sıkıntı olmaması, senkronizasyon olması için semafor kullandım. Ilk olarak burda soketten okuma yapıyorum. Soketten gelen struct'ı burada okuyup client'a ait olan klasör ismiyle, klasör oluşturup, structdan gelen fileları oluşturulan klasöre yazıyoruz.

Gelen struct'tı okurken, eğer clientda ctrl-c gelmişse ilk olarak bunun kontrolünü yapıyorum. Eğer ctrl-c gelmişse ve diğer clientlar içinde aynı dosya olan client varsa iki clientda aynı dosyalar sahip olacak. Sinyal gelen client'a serverdaki dosyaların hepsi gönderilir.

Clietlar için server dosyasında klasör oluşrutulurken mutex ile lock ve unluck kullandım. Aynı klasörde olan clientler dosyaları yazılırken sıkıntı olmaması için, önce biri o dosyayı oluşurup kendi dosyalarını yazıyor sonra diğeri kendi dosyalarını ekliyor.

Eğer clientda bir dosya silinirse bunuda servera bildiriliyor. Bunu kontrol ediyorum. Eğer silinmiş dosya varsa onun ismini direk alıp siliyoruz.

Running Results: