**SUNUCU İSTEK YOĞUNLUĞUNUN MULTITHREAD İLE KONTROLÜ**

|  |
| --- |
| Abdulkadir GÜNDÜZ / PROJE 2 |

***Özet--Projede bir sunucuya gelen isteklerdeki aşırı yoğunluğu, multithread kullanarak alt sunucularla birlikte azaltmaktır.***

***Anahtar Kelimeler – Ana Sunucu, Alt Sunucu, Kabul, Cevap , İstek Sayısı.***

I. GİRİŞ

Proje gelişiminde programlama dili olarak “Java”, geliştirme ortamı olarak “ Eclipse” kullanılmıştır.

Gelen isteklerin yoğunluğuna göre alt sunucu oluşturulur. Ana sunucu kendi içerisinde rastgele sayıda istek, ana sunucu kapasitesinden 2 alt sunucu istek alarak cevaplamaya çalışır. Alt sunuculardan herhangi biri kapasitesinin %70 ve üzeri kadar istek aldıysa yeni bir alt sunucu oluşturarak isteklerin yarısını o alt sunucuya aktarır.

II. YönteM

Uygulamayı çalıştırdıktan sonra istek sayısını girip üret butonuna basarak getText metoduyla textfield alanına yazılan istek sayısı alıp action evet yöntemiyle buton aktif edildi.

Butona tıkladıktan sonra ana sunucu için istek kabul etmek için ve isteklere cevap vermek için 2 thread oluşturuldu. Alt sunucular için de aynı şekilde ikişer thread oluşturuldu. Yani topalmda 6 thread oluşturulmuş oldu.

Her thread içerisinde run metodu override edildi. Ana sunucunun istekleri kabul etmesi için oluşturulan thread içerisindeki run metodunda mainkabul metodu, ana sunucunun istekleri cevaplaması için oluşturulan thread içerisindeki run metodunda maincevap metodu, alt sunucunun istekleri kabul etmesi için oluşturulan thread içerisindeki run metodunda subkabul metodu, alt sunucunun istekleri cevaplaması için oluşturulan thread içerisindeki run metodunda subcevap metodu çağrıldı. Threadler start metoduyla başlatıldı.

Ana sunucu için istek kabul threadi ile istek cevap threadi arasında senkronizasyon sağlanması için synchronized(lock)[5] metodu kullanıldı. Ana sunucu istek kabul ile alt sunucu istek kabul arasındaki senkronizasyon sağlanması için de ReentrantLock[7] metodu kullanıldı. Eğer kapasite dolduysa istek alımını durdurmak için ya da kapasite sıfır ise istek cevaplamayı durdurmak için wait metodunu [6] kullanıldı.

III. deneysel sonuçlar

Proje için örnek olarak resimlerde kullanıcının seçimine göre açılan pencereler gösterilmiştir.

Resim 1:. Istek sayısı girilen ekran görüntüsü



Resim 2: Ana sunucu istek Kabul ve istek cevaplama

main kapasite % 74

main kapasite % 75

main kapasite % 75

main cevaplanan = 5779

main kapasite % 75

main kapasite % 76

main kapasite % 75

main cevaplanan = 5822

Resim 3: Alt sunucular istek Kabul ve istek cevaplama

sub 1 kapasite % 15

SUB 1 cevaplanan istek 2691

main kapasite % 76

main kapasite % 76

main cevaplanan = 5859

main kapasite % 76

sub 2 kapasite % 18

SUB 2 cevaplanan istek 2326

Resim 4: Ana sunucu kapasitesi dolduğu zaman thread.wait() ile beklemesi sonucu konsoldaki görünüm.

main kapasite % 99

main cevaplanan = 7011

main kapasite % 100

main kapasite % 99

main cevaplanan = 7040

main kapasite % 100

sub 2 kapasite % 20

main kapasite % 99

sub 1 kapasite % 29

main kapasite % 99

main kapasite % 100

sub 1 kapasite % 28

SUB 1 cevaplanan istek 3316

sub 2 kapasite % 19

SUB 2 cevaplanan istek 3493

main kapasite % 99

main cevaplanan = 7058

main kapasite % 100

Resim 5: Alt sunuculardan herhangi birinin kapasitesi %70 olduğunda yeni alt sunucu oluşturma durumunun konsoldaki görünümü.

sub 2 kapasite % 70

SUB 2 cevaplanan istek 8471

main kapasite % 99

main kapasite % 99

main cevaplanan = 17191

main kapasite % 99

main kapasite % 99

sub 2 kapasite % 35

main kapasite % 99

sub 1 kapasite % 62

main kapasite % 99

main kapasite % 98

main cevaplanan = 17225

main kapasite % 99

sub 1 kapasite % 61

SUB 1 cevaplanan istek 8854

sub 2 kapasite % 35

SUB 2 cevaplanan istek 8503

main kapasite % 100

main kapasite % 99

main cevaplanan = 17245

main kapasite % 100

sub 1 kapasite % 62

main kapasite % 99

sub 3 kapasite % 35

main kapasite % 99

sub 2 kapasite % 35

main kapasite % 99

main kapasite % 99

sub 3 kapasite % 35

SUB 3 cevaplanan istek 1

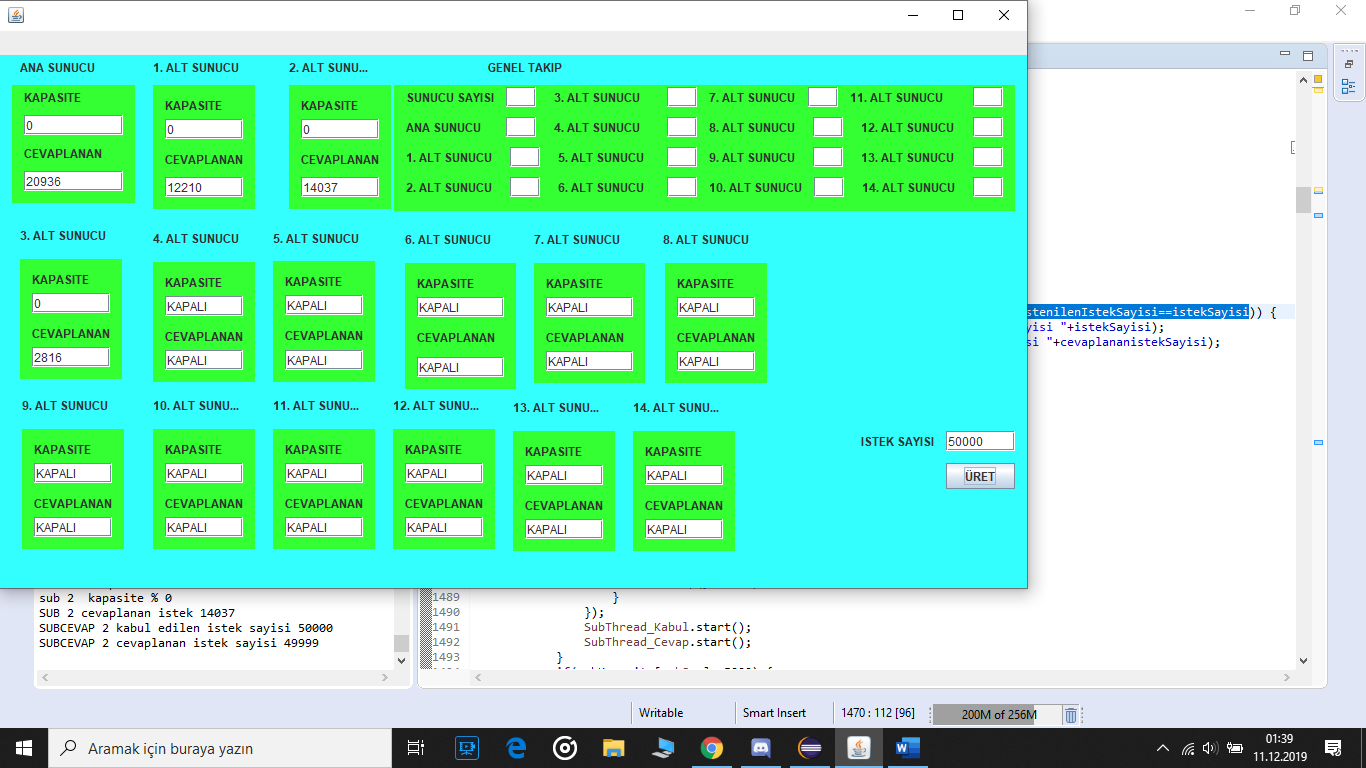
Resim 6: Toplam Kabul edilen istek sayısı ve cevaplanan istek sayısının konsolda görünümü.

TOPLAM kabul edilen istek sayisi 500

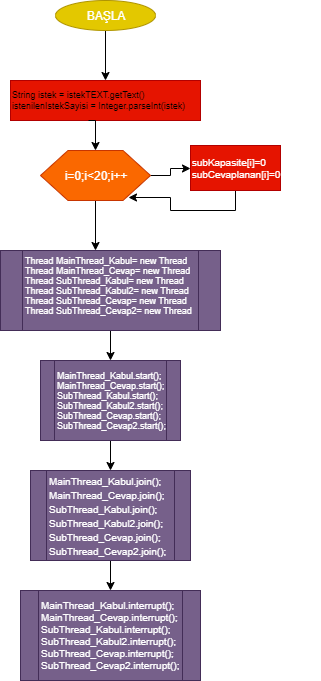
TOPLAM cevaplanan istek sayisi 500

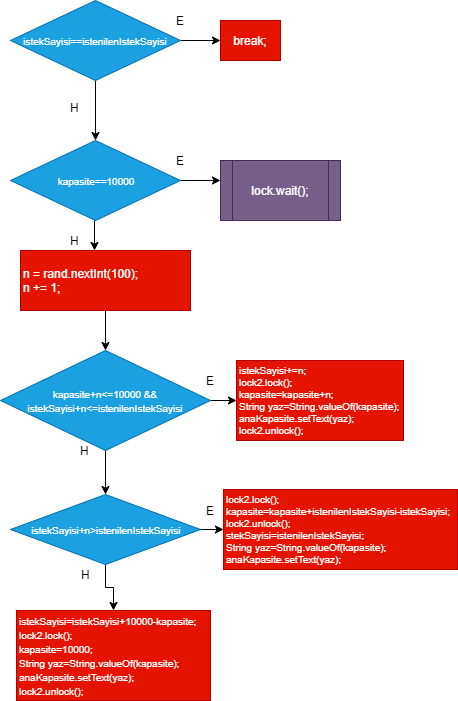
Resim 6: Toplam Kabul edilen istek sayısı ve cevaplanan istek sayısının pencerede görünümü.

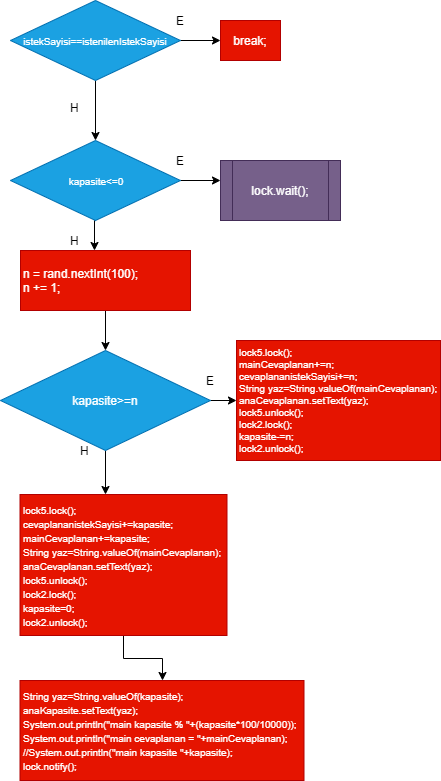


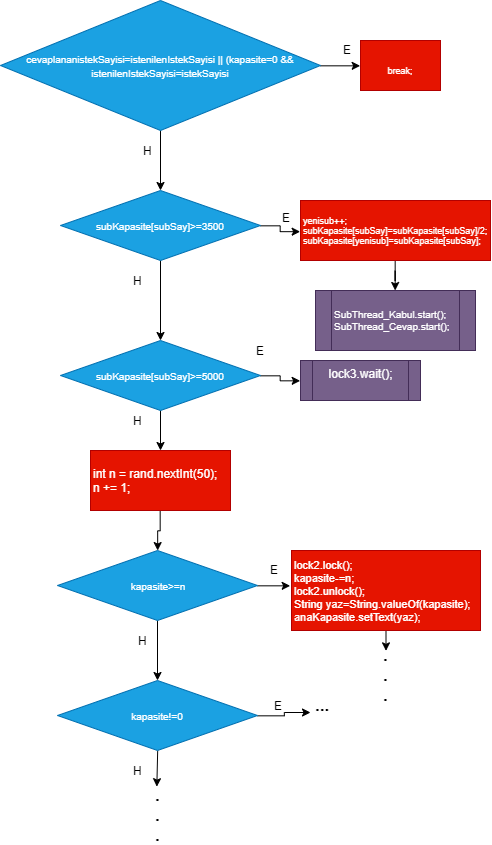


IV. akış diyagramı









KAYNAKLAR

1. https://ufukuzun.wordpress.com/2015/02/14/javada-multithreading-bolum-1-merhaba-thread/ (Access date: 14 ŞUBAT 2015)
2. https://www.youtube.com/watch?v=Xj1uYKa8rIw (Access date: 24 Kas 2016)
3. https://www.youtube.com/watch?v=vCDrGJWqR8w (19 Şub 2015)
4. https://medium.com/gokhanyavas/javada-multithreading-bbc6a9181772(Nisan 20, 2017)
5. <https://www.udemy.com/course/sifirdan-ileri-seviyeye-komple-java-gelistirici-kursu/learn/lecture/8523436#overview>
6. <https://www.udemy.com/course/sifirdan-ileri-seviyeye-komple-java-gelistirici-kursu/learn/lecture/8523460#overview>
7. https://www.udemy.com/course/sifirdan-ileri-seviyeye-komple-java-gelistirici-kursu/learn/lecture/8523472#overview