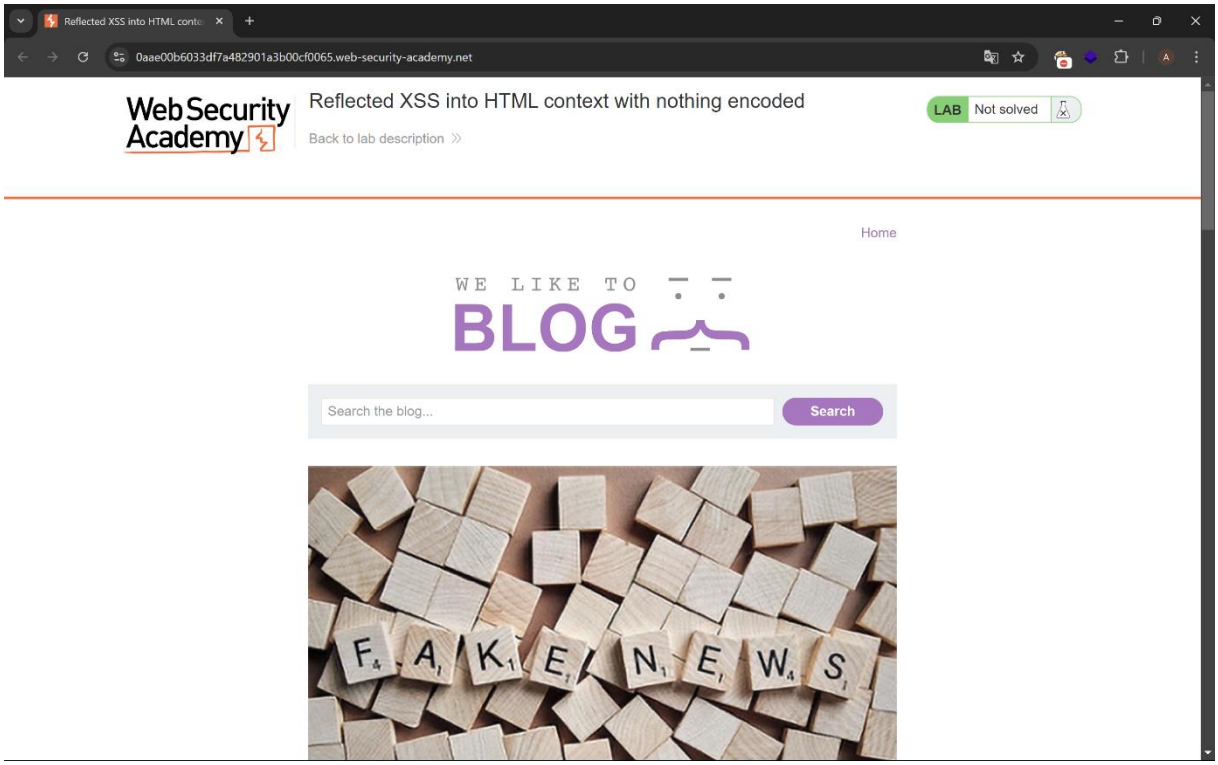
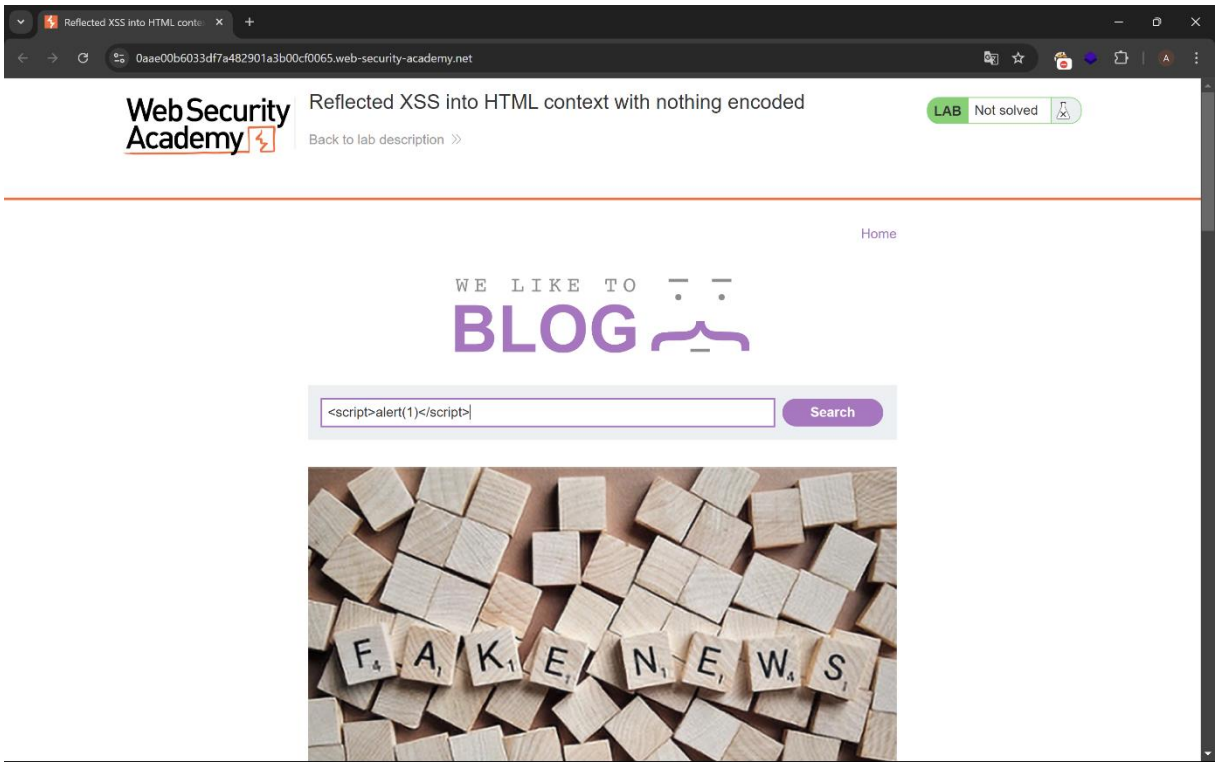
Yavuzlar Ödev

Ali izzet alp ikbal

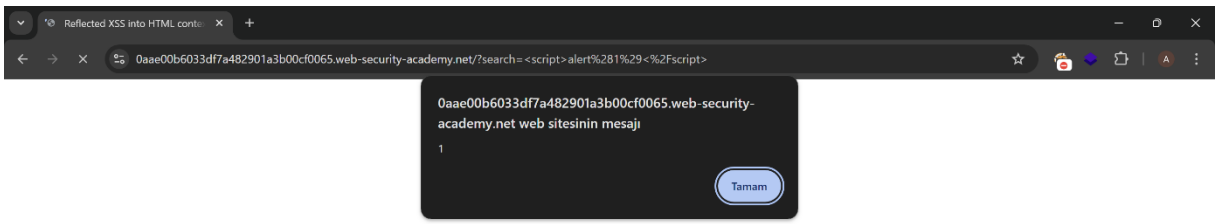
1. **Injection**
   Bu konuda bir çok zafiyet türü var ve ben de injection konusunda seçtiğim konu XSS. Portswigger labları çok öğretici olduğu için oradan bir lab seçtim.



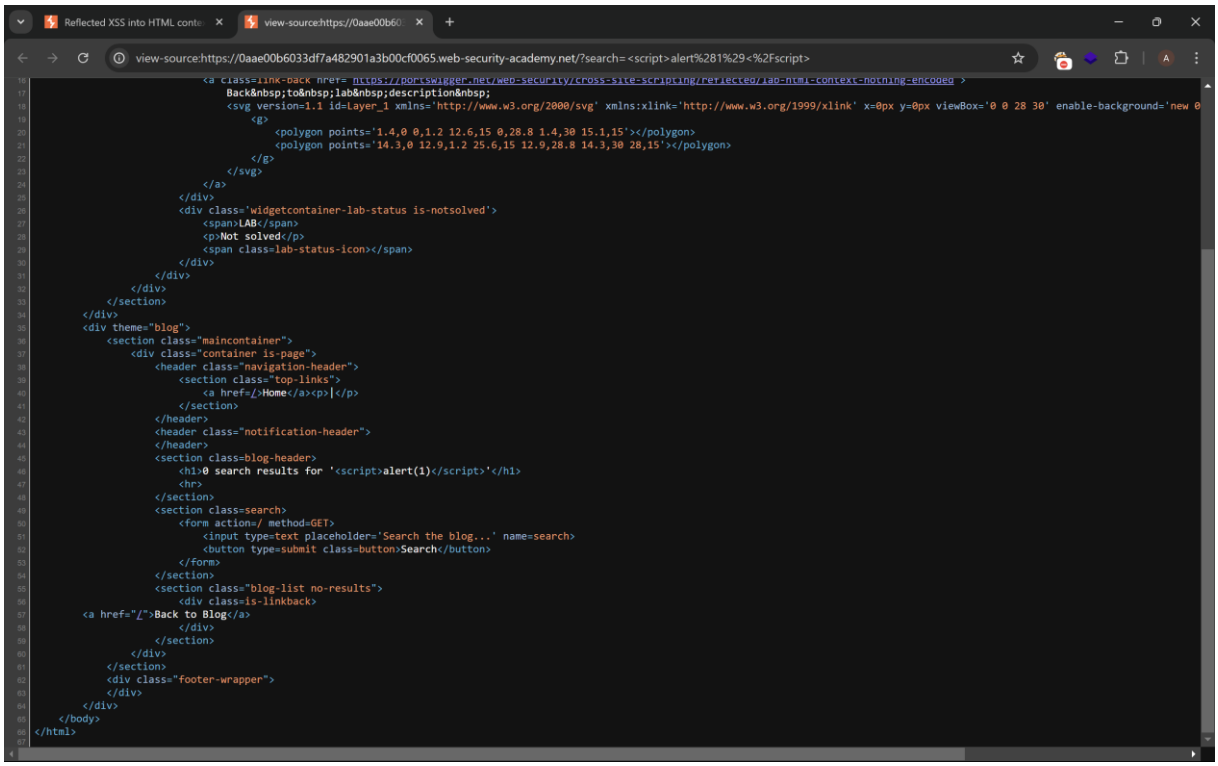
Bu tarz arama ve yorum yapma gibi bir çok alanda yani input verebildiğimiz her alanda potansiyel injection zafiyeti bulunabilmektedir. O sebeple açığın olup olmadığını kontrol etmek için zararlı js kodumuzu yazıyoruz.



Kodumuzu yazdıktan sonra çalıştırıyoruz.

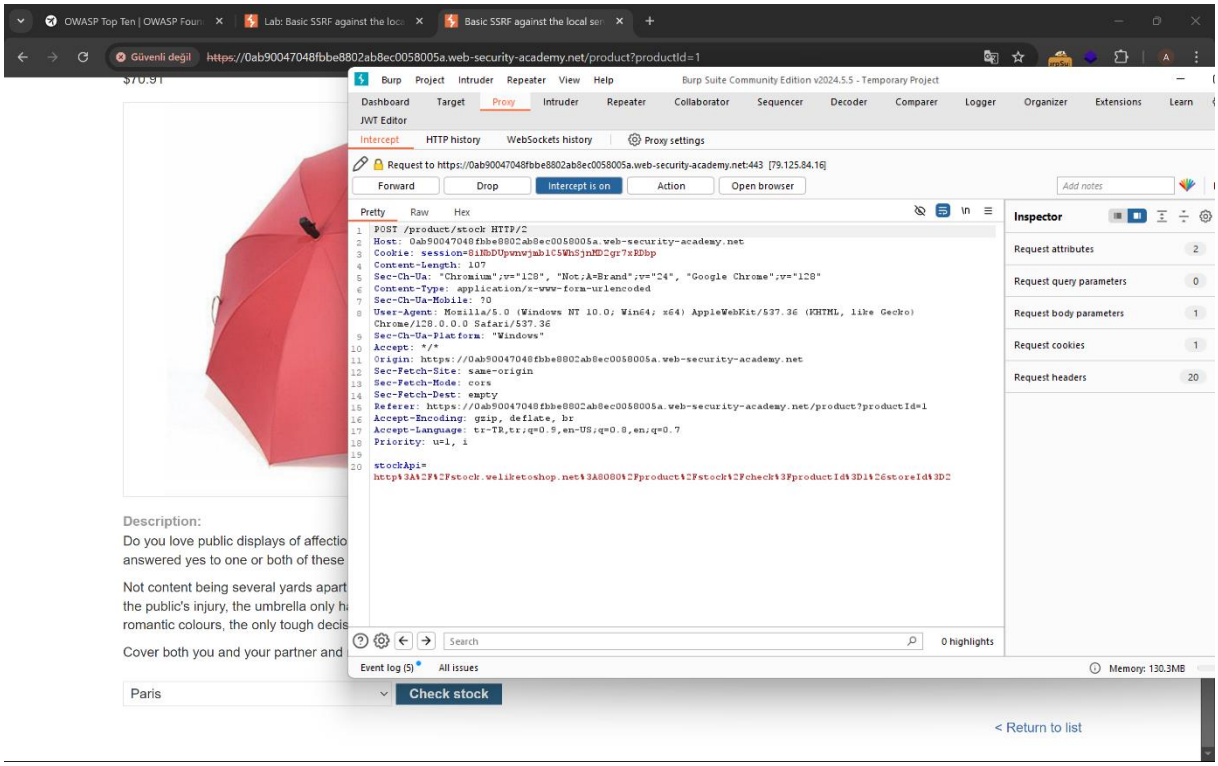


Ve bu şekilde zafiyetin varlığını kanıtlamış oluyoruz.

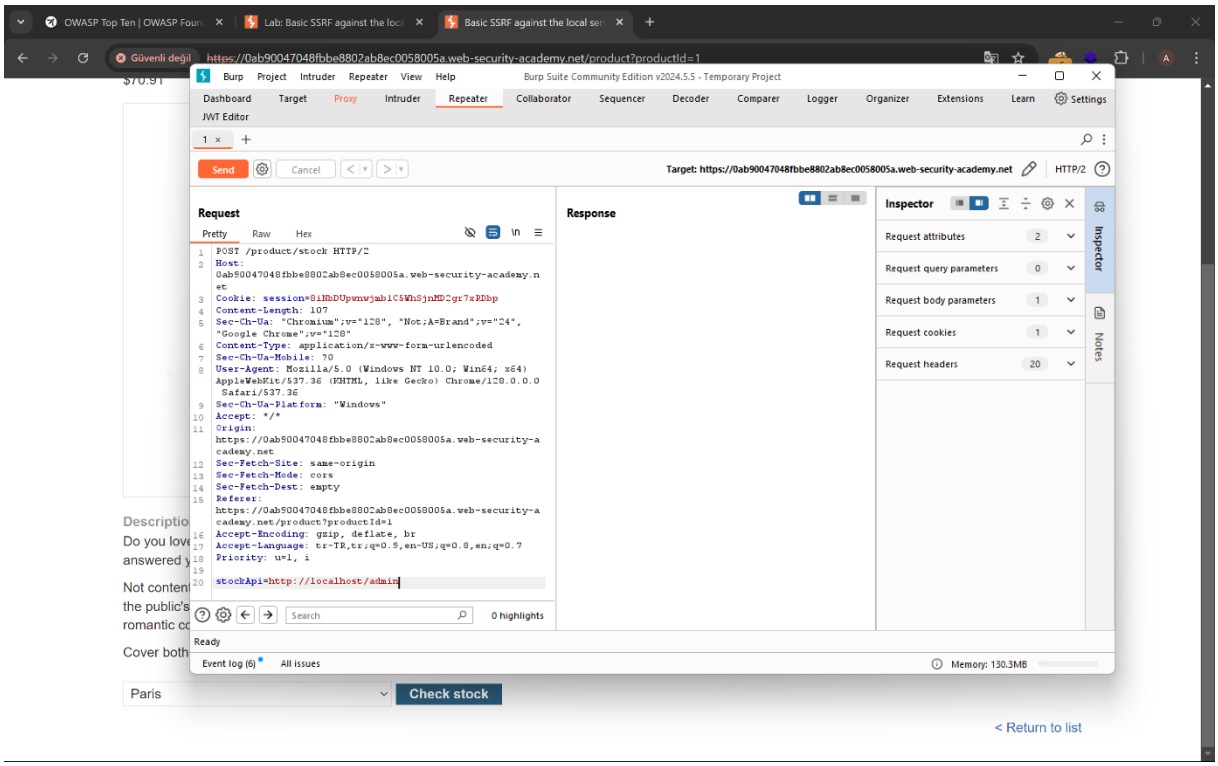


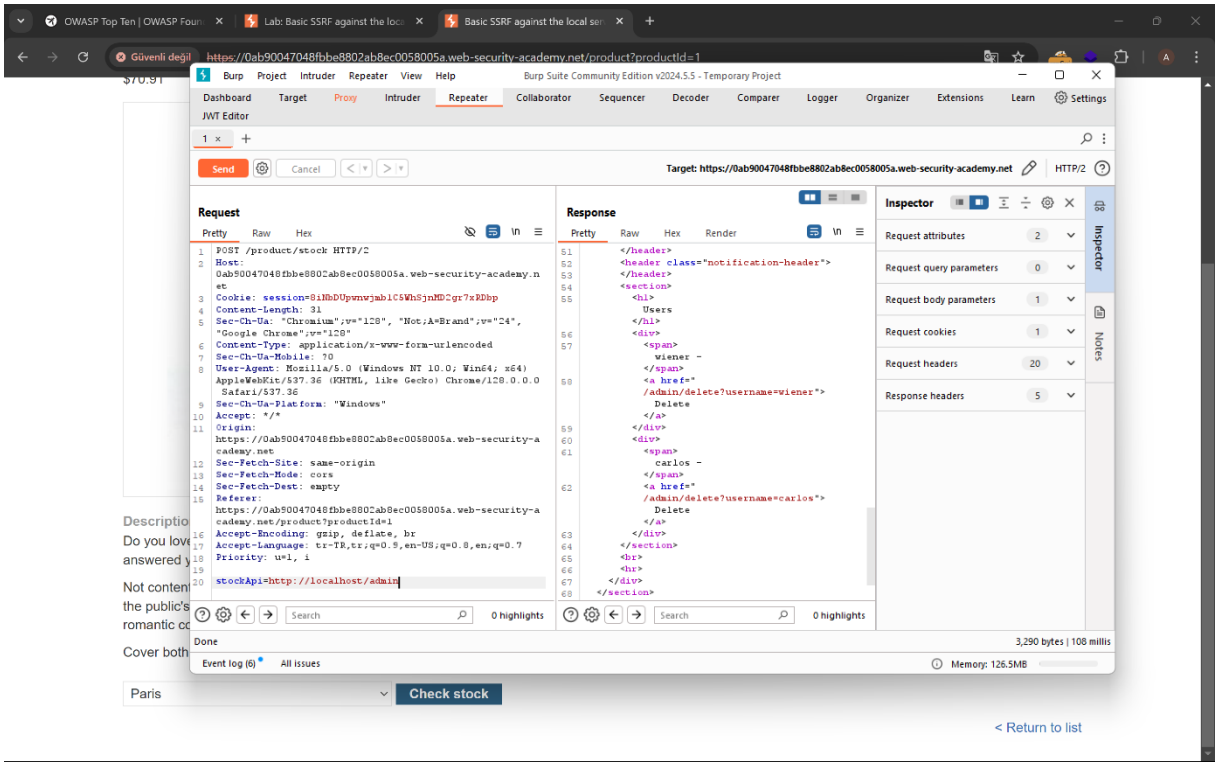
Sayfanın kaynağına baktığımızda kodumuzu görebiliyoruz.

2. **Server Side Request Forgery (SSRF)**
   Bu konu isminden de anlaşıldığı gibi server taraflı istek aldatmacası gibi bir durum söz konusudur. Server tarafından gelen isteğin kontrol edilmemesinden kaynaklanır genellikle. Kontrol edildiği durumlarda da black list - white list durumlarına göre url encodingler ve farklı yöntemlerle de exploit edilebiliniyor.

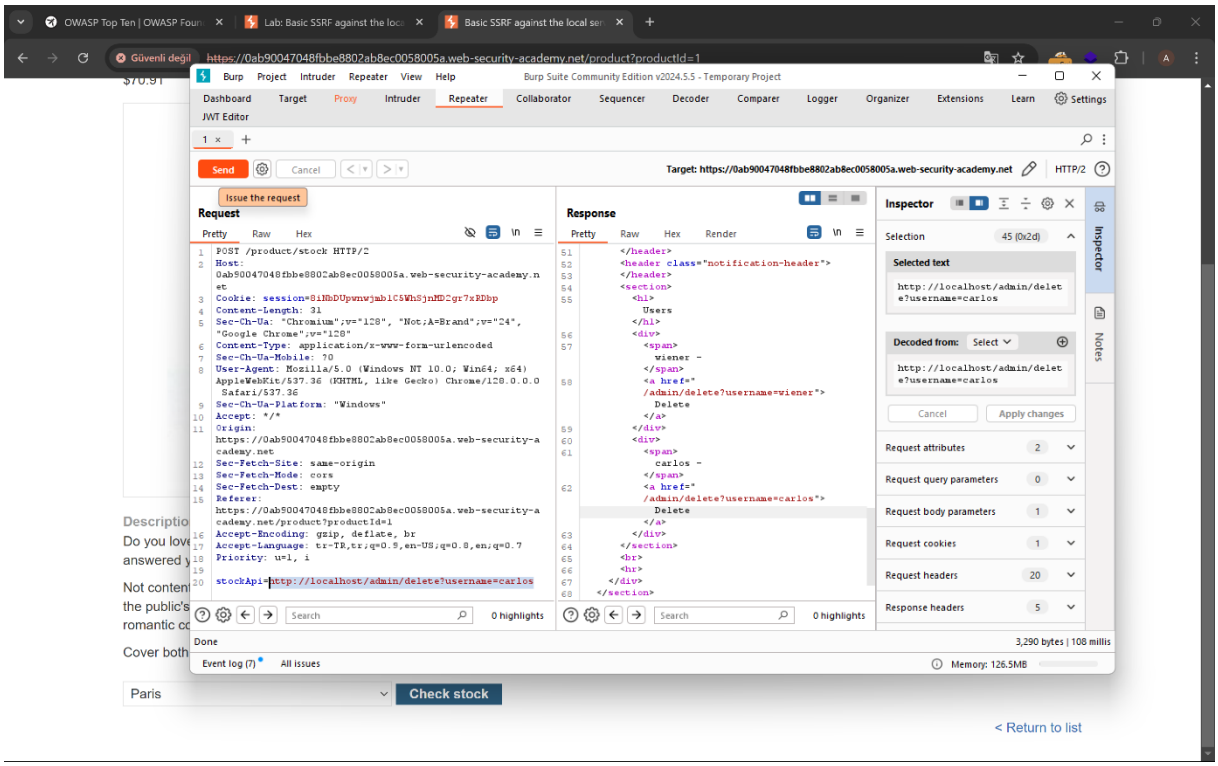


Bu lab içerisinde stock check sistemi var ve bu isteği yakalıyoruz. Gördüğümüz gibi bir apiye giden istek üzerindeki linki değiştirebilme şansımız var. Hemen bu isteği repeater kısmına atıyoruz ve linki http://localhost/admin olarak değiştiriyoruz.

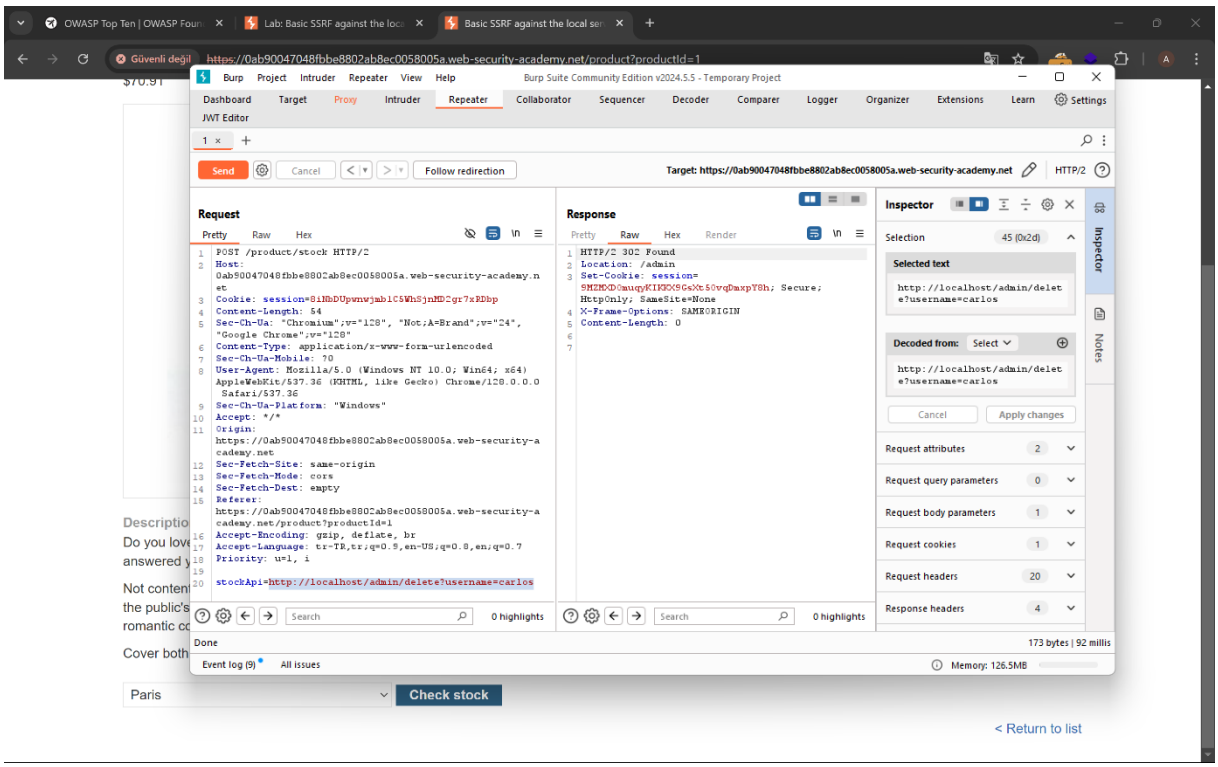




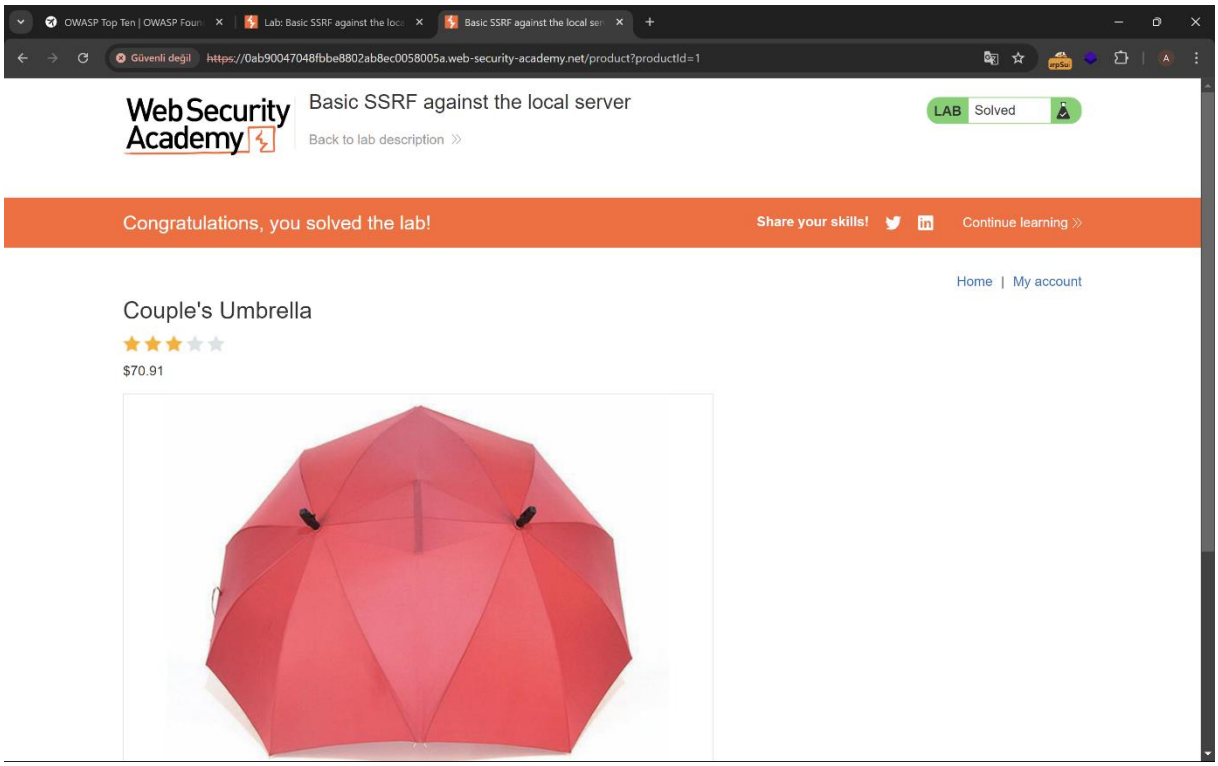
Gelen response içerisindeki bilgilere bakarak user silebilmeyi görebiliyoruz ve linki ona göre tekrar ayarlıyoruz.



Ve isteği gönderiyoruz.

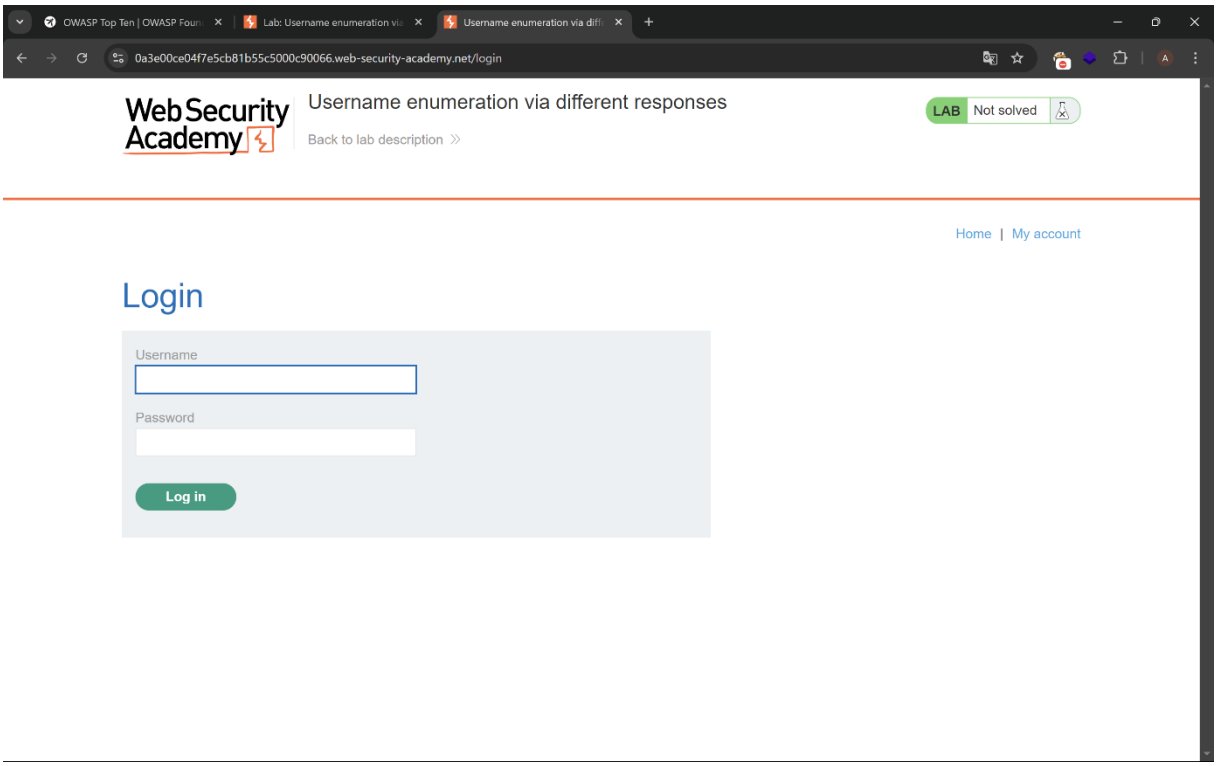


Gelen cevapta 302 status kodu var bunun bizi başka bir sayfaya yönlendirdiğini görüyoruz ve tekrar kontrol ediyoruz kullanıcı silindi mi diye.

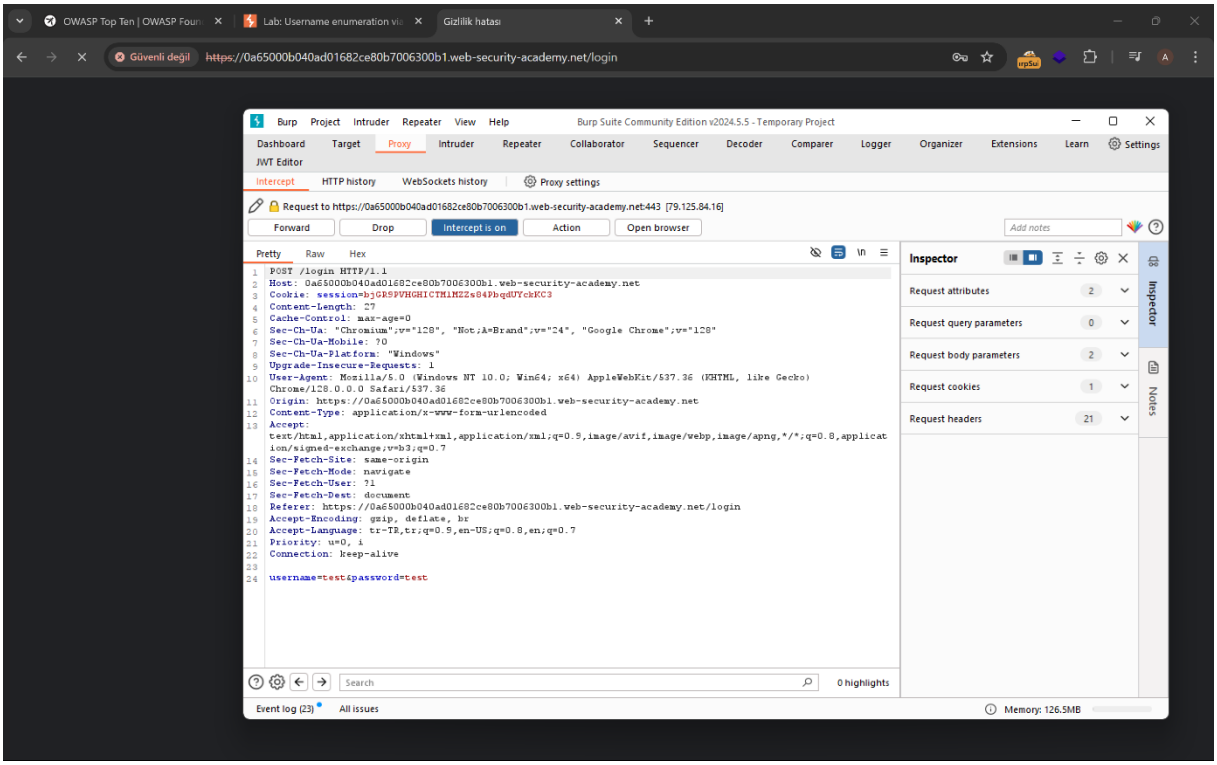


Ve başarılı bir şekilde kullanıcıyı silmiş bulunmaktayız.

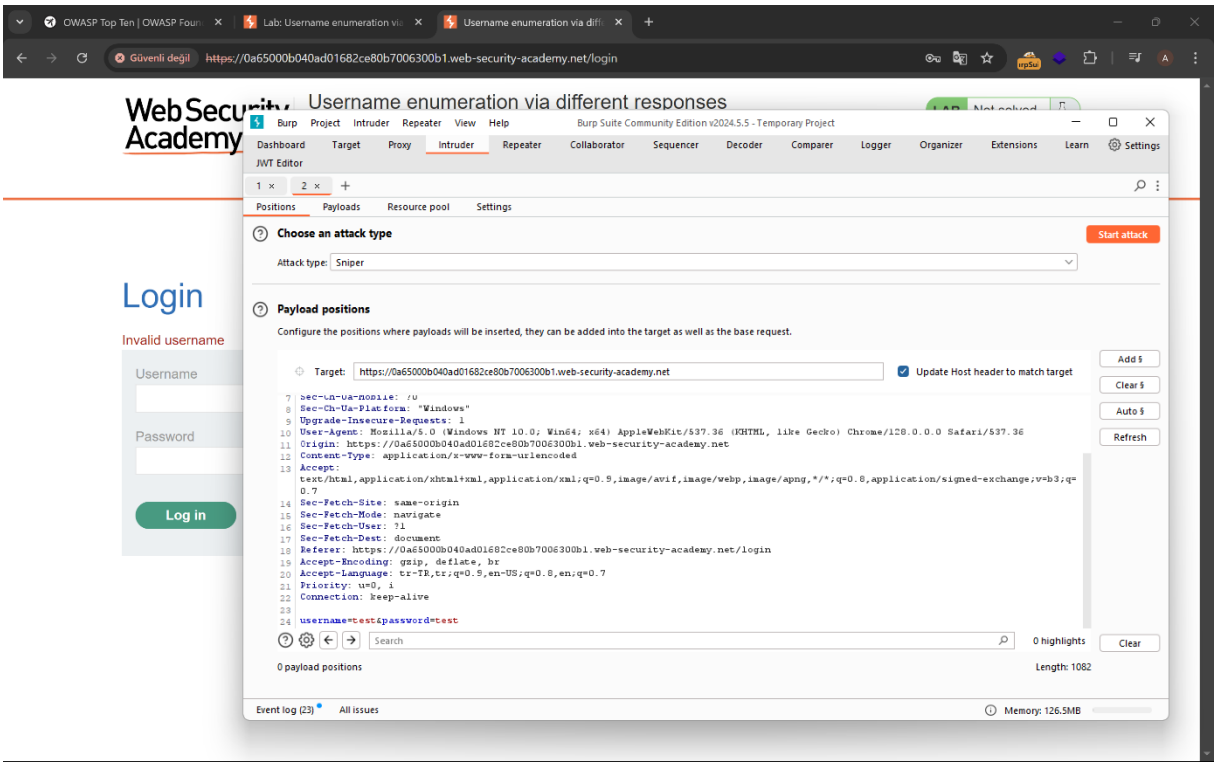
3. **Identification and Authentication Failures**
   Bu konu için portswigger lablarından authentication başlığı altındaki **Lab: Username enumeration via different responses** labını seçtim. Buradaki açık aslında authorization ve authentication arasındaki farkı bize güzel gösteriyor. Authentication kullanıcıyı doğrulamak için Authorization ise kullanıcının bir şey yapma veya yapamama yetkisiyle alakalı. Hemen çözümüne geçelim.



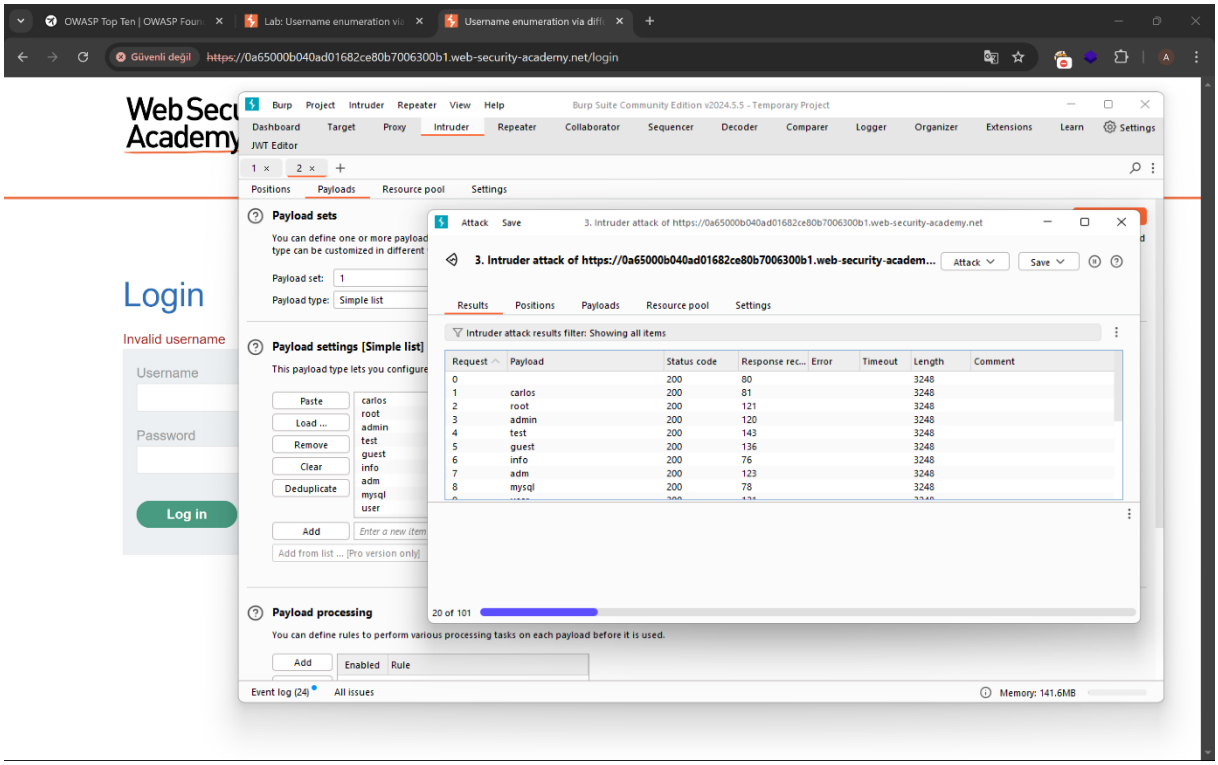
Öncelikle buradaki login kısmındaki isteği yakalıyoruz.



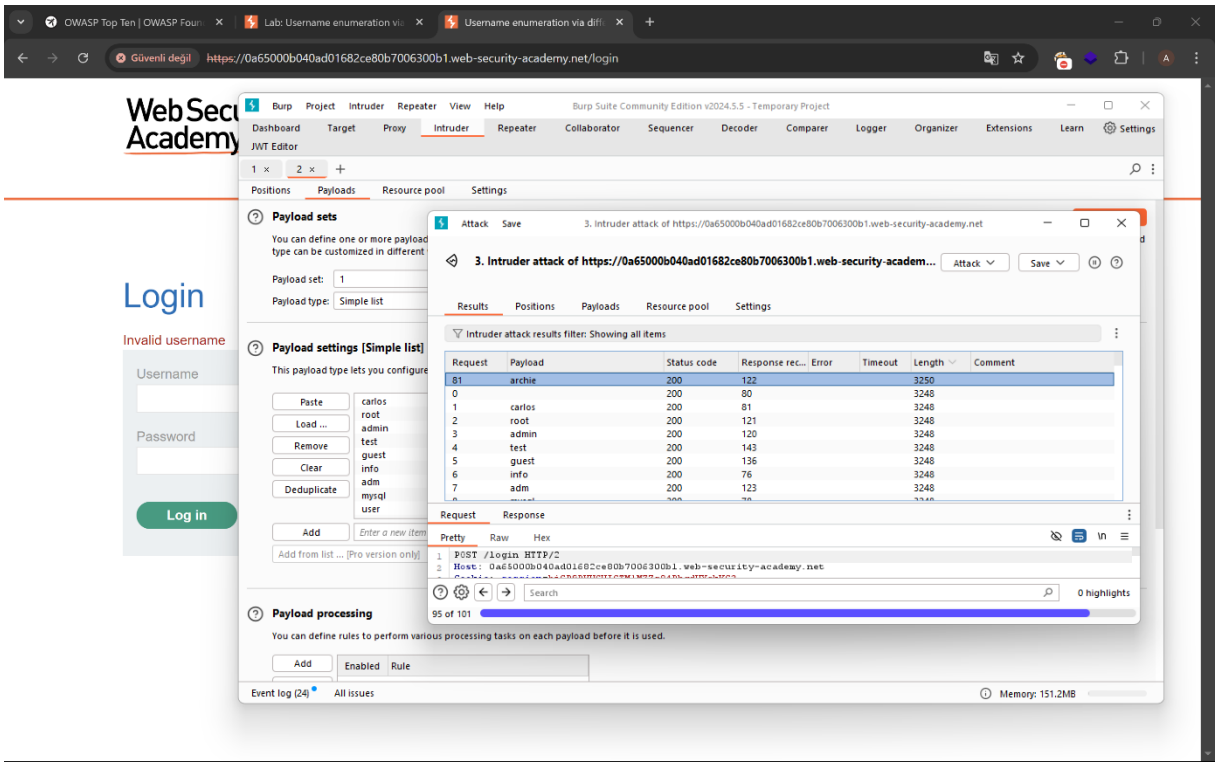
Yakaladığımız isteği intrudera gönderiyoruz.



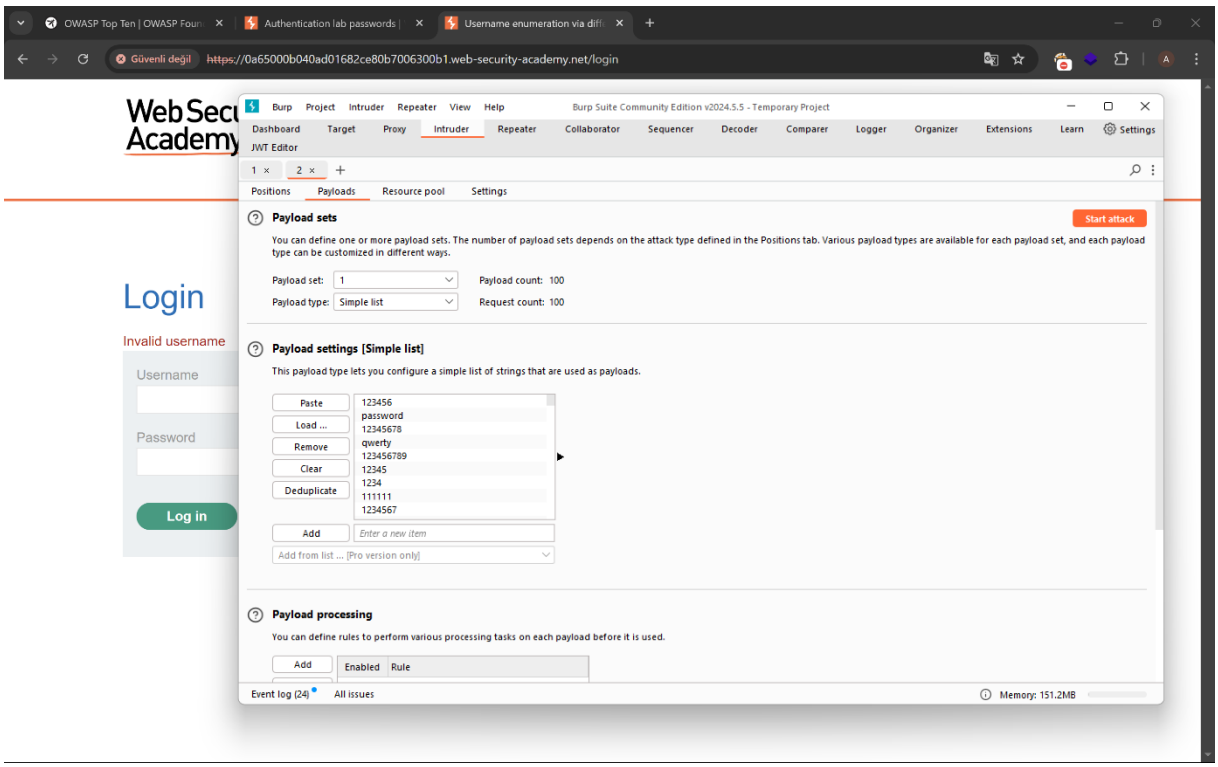
Burada username kısmını seçerek add diyoruz ve payloads kısmına geçiyoruz.

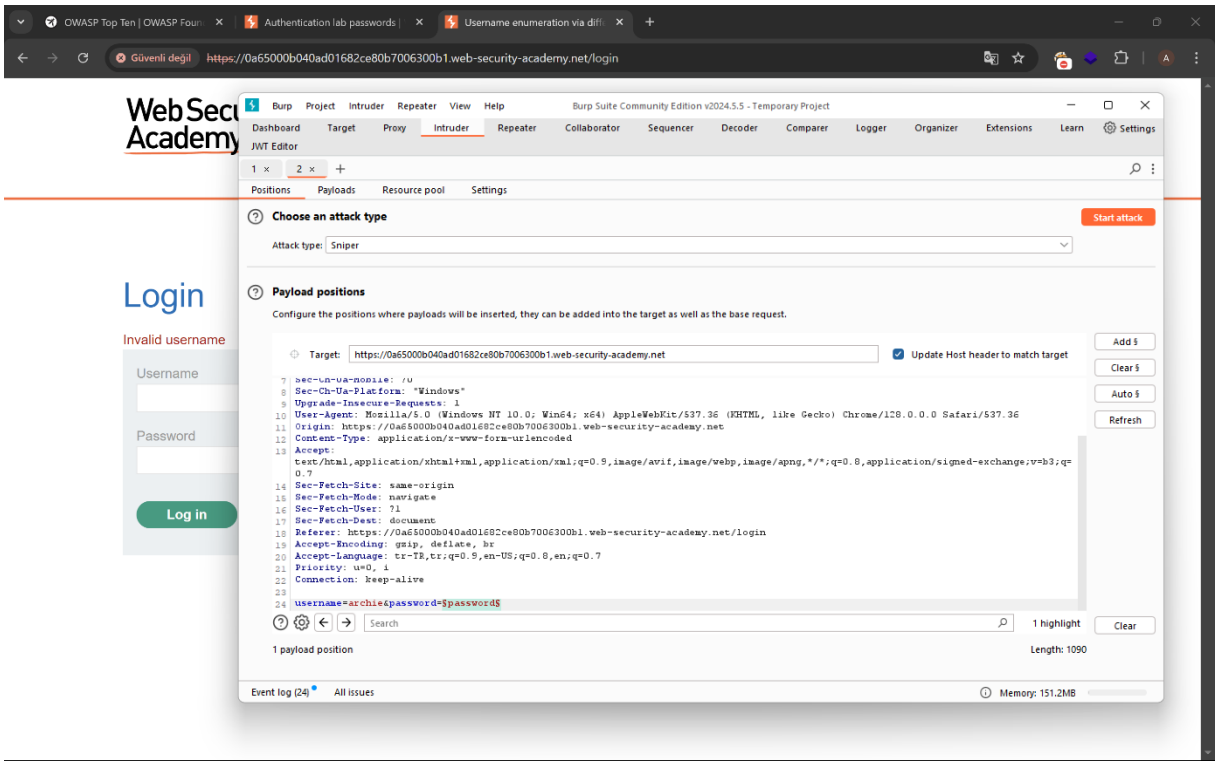


Username listimizi payload kısmına ekledikten sonra start attack diyoruz. Buradan sonra dönen cevapların uzunluğuna göre inceleyeceğiz.

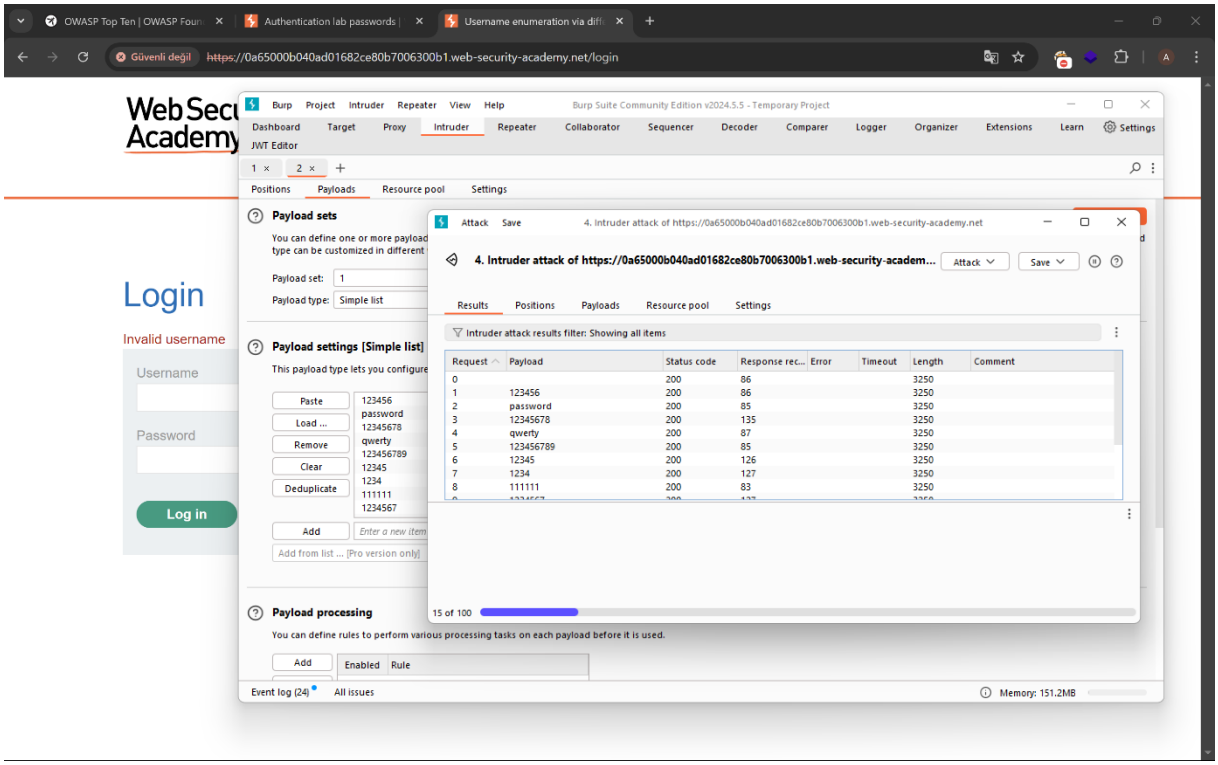


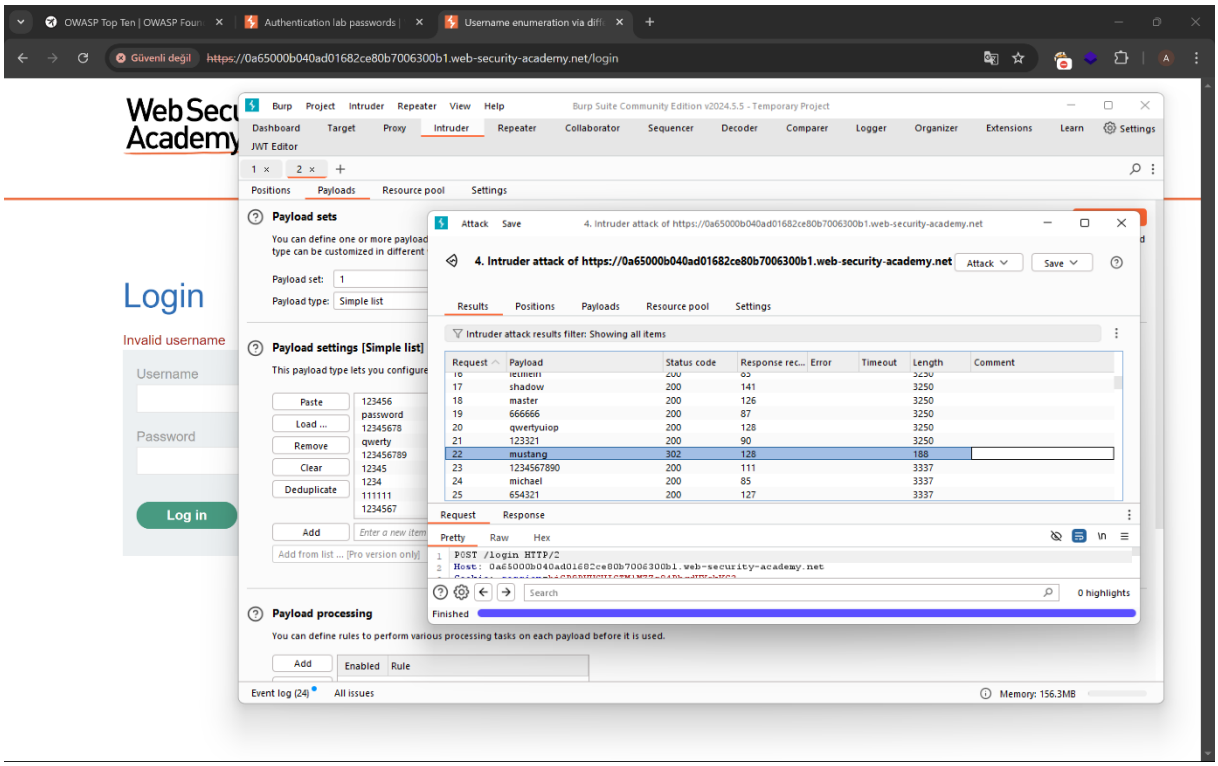
Dönen cevaplar içinde tek farklı olan archie kullanıcı adıydı. Şimdi username bilgisine sahibiz, sırada password listimizi denememiz lazım. Username kısmına archie yazıyoruz ve aynı işlemi password kısmı için de yapıyoruz.



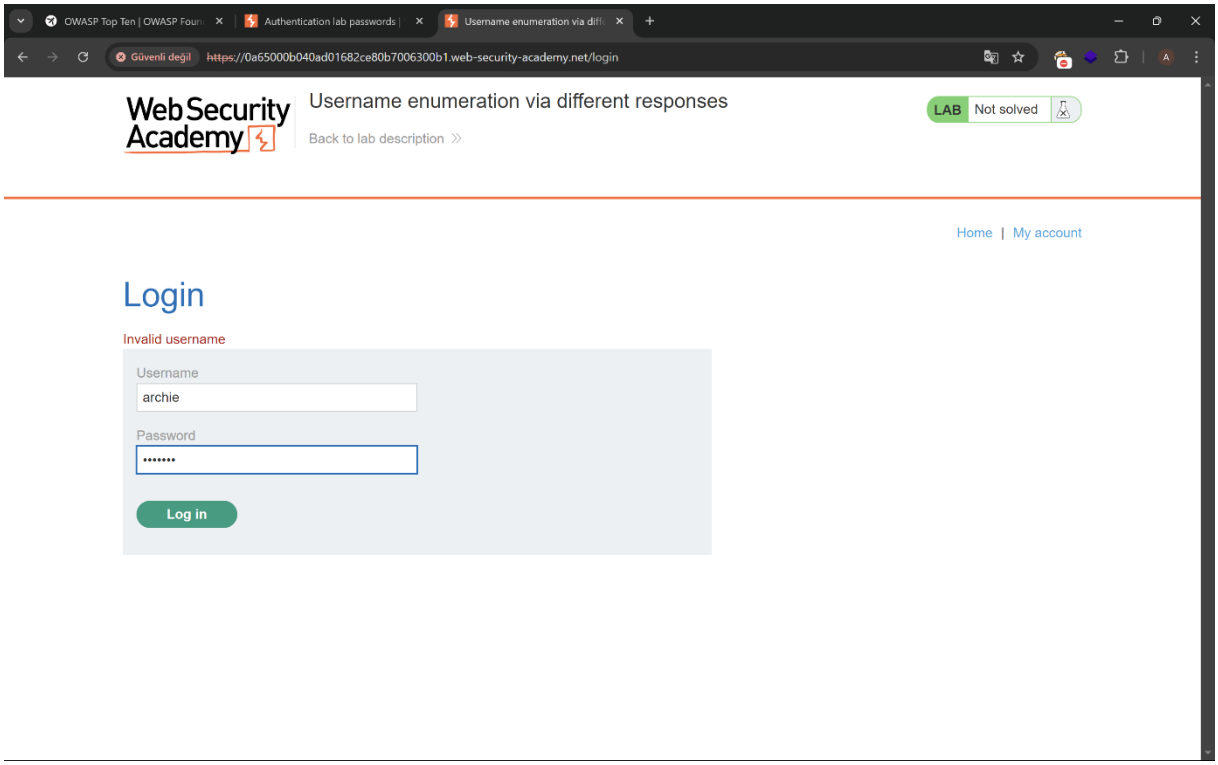


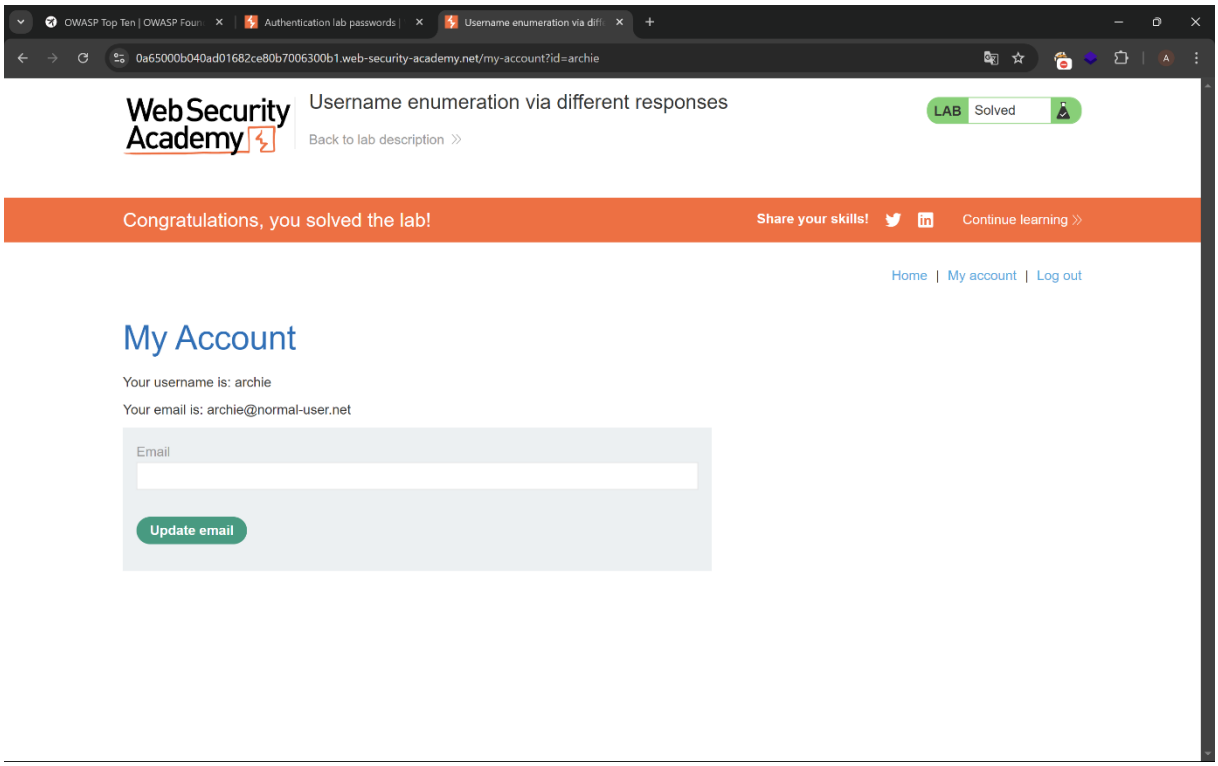
Ve start attack diyoruz





Buradan da görebileceğimiz gibi tek farklı uzunluğa sahip cevabı bulduk ve şimdi şifreyi de elde etmiş olduk. Artık manuel olarak denememiz lazım.





Ve bu labı da başarıyla çözmüş oluyoruz. Buradaki önemli nokta en başta belirttiğim gibi authantication yönetimiyle alakalıydı authorizationdan farklı olarak. Kimlik doğrulama açıklarının etkisi ciddi olabilir. Bir saldırgan kimlik doğrulamayı atlatırsa veya başka bir kulanıcının hesabına kaba kuvvetle girerse, tehlikeye atılan hesabın sahip olduğu tüm verilere erişebilir ve kontrol edeblr. Sistem yöneticisi gibi tam yetkili bir hesabı tehlikeye atabilrlerse, tüm uygulama üzerinde tam kontrole sahip olabilir ve altyapıya erişim sağlarlar.