YODA İHA Yazılım Birimi - 2. Hafta Ödevi

Hazırlayan: Aydın Alizade
Kaynak: METU ACM Student Chapter | Git & GitHub Eğitimi

Genel Bakış

Bu ödev sürecinde Git ve GitHub'ın temellerini öğrendim. Yazılım geliştirme süreçlerinde sürüm kontrol sistemlerinin neden bu kadar önemli olduğunu fark ettim. Özellikle ekip çalışmasında yapılan değişikliklerin düzenli ve takip edilebilir olması açısından Git'in güçlü bir araç olduğunu gördüm.

Öğrendiğim Temel Kavramlar

Git

- Kod üzerinde yapılan değişiklikleri kaydeden bir versiyon kontrol sistemidir.
- Geliştiricilere, kodun geçmiş sürümlerine dönme ve ekip olarak aynı proje üzerinde çalışabilme imkânı sunar.

GitHub

- Git ile oluşturulan projelerin çevrimiçi ortamda saklandığı ve paylaşıldığı platformdur.
- Takım çalışması, açık kaynak katkısı ve proje yönetimi için kullanılır.

Temel Git Kavramları ve Kullanımları

- **Repository (Repo):**
  Proje dosyalarının, geçmiş sürümlerin ve değişiklik kayıtlarının tutulduğu depodur. Geliştirici bir proje üzerinde çalışmaya başlarken genellikle `git init` komutu ile yeni bir repo oluşturur veya var olan bir GitHub reposunu `git clone` komutuyla bilgisayarına indirir.
- **Commit:**
  Projede yapılan değişikliklerin yerel depoya kaydedilmesini sağlar.
  Bir nevi "kaydetme işlemi" gibidir ancak bu işlem yalnızca kendi bilgisayarında gerçekleşir.
  Her commit'e açıklayıcı bir mesaj eklenir, bu sayede değişikliklerin nedeni ve içeriği kolayca takip edilebilir.

- **Stage (Staging Area):**
  Commit işleminden önce, hangi dosyaların kaydedileceğinin seçildiği hazırlık alanıdır.
  Yani dosyalar önce stage'e alınır (`git add`), ardından commit edilir (`git commit`).
  Bu sayede sadece istenen değişikliklerin kayıt altına alınması sağlanır.
- **Branch:**
  Projenin ana sürümünden bağımsız olarak yeni bir özellik veya geliştirme üzerinde çalışmak için oluşturulan yan dalıdır.
  Örneğin, ana kodu bozmadan yeni bir deneme yapmak veya hata gidermek için branch kullanılır.
  Çalışma tamamlandıktan sonra bu branch, ana projeye birleştirilebilir.
- **Merge:**
  Farklı branch'lerde yapılan çalışmaların ana projeye veya başka bir branch'e eklenmesini sağlar.
  Örneğin, `odev_2` branch'inde yapılan değişiklikler tamamlandığında `main` branch'i ile birleştirilir.
  Bu işlem, projedeki farklı katkıların tek bir bütün haline gelmesini sağlar.
- **Push:**
  Yerelde yapılan commit'lerin GitHub'daki uzak depoya gönderilmesini sağlar.
  Commit işlemi yalnızca bilgisayarımıza kayıt yaparken, **push** işlemi değişiklikleri buluta (GitHub'a) aktarır.
  Bu sayede ekip arkadaşları güncel kodlara erişebilir.
- **Pull:**
  GitHub'daki (uzak depodaki) en son değişiklikleri yerel ortama indirir.
  Özellikle ekip projelerinde, başkalarının yaptığı commit'leri almak için kullanılır.
  Böylece herkesin kod tabanı güncel kalır.

Yaşadığım Bir Sorun ve Çözümü

Git'i bilgisayarıma kurduktan sonra **VS Code üzerinden commit** yapmak istediğimde hata aldım.
Sistemin benden kullanıcı adı ve e-posta bilgisi girmemi istediğini fark ettim.

Bu hatayı çözmek için terminalde aşağıdaki komutları girerek sorunu çözdüm:

```
git config --global user.name "İsim Soyisim"
git config --global user.email "mail.adresi@gmail.com"
```

Bu problem her ne kadar basit bir sorun gibi görünse de, ilk kez karşılaştığım için çözüm süreci beklediğimden uzun sürdü. Kaynak videoda bu konuya değinilmediğinden, çözümü internet araştırmaları yaparak kendi başıma bulmam gerekti. Bu süreç, zaman kaybettirse de karşılaşılan hataları bağımsız şekilde çözme becerimi geliştirdiği için benim açımdan değerli bir deneyim oldu.

Sonuç

Bu süreçte:

- Git ve GitHub’ın mantığını öğrendim,
- Temel komutların ne işe yaradığını kavradım,
- Yaşadığım küçük bir hatayı terminal komutlarıyla çözmeyi başardım.

Bu ödev bana, yazılım geliştirme sürecinde sürüm kontrolünün ne kadar kritik olduğunu ve pratik yapmanın öğrenmeyi ne kadar kolaylaştırdığını gösterdi.