**Liana Dosmanbetova**

**Öğrenci NO: 98716435059**

**Ders: Algoritma ve Programlama**

**Alıcı: Dr. Öğr. Üyesi Hüseyin PARMAKSIZ**

**Googleye ubuntuda Python nın farklı farklı sürümleriyle kod yazılınca çalışmasını engelleyen veya kontrol eden sistem var mı?**

Evet, Ubuntu’da Python projelerinde kütüphane ve bağımlılıkların çakışmasını önlemek için birkaç yöntem kullanabilirsiniz. En yaygın çözümlerden biri virtual environment (sanallaştırılmış ortamlar) oluşturmaktır. Python’un venv modülü bu işlemi basit hale getirir. Her proje için izole edilmiş bir sanal ortam yaratarak, her projede farklı Python kütüphaneleri ve sürümlerini kullanabilirsiniz.

**Virtual Environment Kullanımı:**

1. Python Virtual Environment oluşturma:

Öncelikle projenizin kök dizinine gidin ve şu komutu çalıştırın:

python3 -m venv venv

Bu komut venv adında bir sanal ortam oluşturur. (İsim farklı olabilir, örneğin myenv).

2. Virtual Environment’ı aktif etme:

Sanal ortamı etkinleştirmek için:

source venv/bin/activate

Bunu yaptığınızda terminalde (venv) etiketi belirir ve artık tüm kütüphaneler bu sanal ortamda kurulacaktır.

3. Gerekli kütüphaneleri yükleme:

Sanal ortam aktifken ihtiyaç duyduğunuz kütüphaneleri normalde olduğu gibi yükleyebilirsiniz:

pip install <package\_name>

4. Sanal ortamı devre dışı bırakma:

Çalışmayı bitirdikten sonra sanal ortamı devre dışı bırakmak için şu komutu kullanabilirsiniz:

deactivate

Bu sayede her projeniz için bağımsız bir Python ortamı oluşturabilir ve kütüphane sürümlerinin çakışmasını önleyebilirsiniz.

**Alternatif Araçlar:**

• Conda: conda ile de sanal ortamlar oluşturabilirsiniz. Özellikle farklı Python sürümleri kullanmanız gerektiğinde faydalıdır.

• Docker: Daha izole ve tamamen bağımsız ortamlar oluşturmak için Docker kullanabilirsiniz. Her proje için bir Docker konteyneri oluşturarak bağımlılıkların tamamen ayrılmasını sağlayabilirsiniz.

Bu yöntemlerle çakışmaların önüne geçip projelerinizin daha düzenli ve yönetilebilir olmasını sağlayabilirsiniz.