AKADEMİK ARAMA SİSTEMİ

**Projenin Amacı:**

Bu projede bizden bir akademik arama motoru geliştirmemiz beklenmektedir. Akademik arama motoru, akademik çalışmalara ve araştırmacılara ulaşabileceğimiz bir web sayfasıdır. Bu proje ile web ve veri tabanı uygulamalarının birlikte kullanılması amaçlanmaktadır ve web tabanlı bir akademik arama sayfası yapmamız beklenmektedir. Projede veri tabanı olarak Neo4j graf veri tabanı kullanılacaktır. Yani kısaca amacımız Web tabanlı bir uygulama geliştirmek ve Neo4j’nin farklı programlama dillerinde bağlantısını ve kullanımını göstermektir.

**Programlama Dili:**

Programımızı Visual Studio Code da nodejs kullanarak geliştirdik ve Database kısmını ise bizden istenildiği üzere Neo4j kullanarak geliştirdik.

**Problem Tanımı:**

Proje bir web uygulaması olarak geliştirilecektir. Web uygulaması için iki farklı arayüz bulunmalıdır. Birincisi Yönetici Arayüzü: Veri tabanına veri girişi yapılacak olan kısım. İkincisi ise Kullanıcı Arayüzü: Kullanıcın içerik araması yapacağı ve çizgenin (graf) görüntüleneceği kısım. Yönetici Arayüzünde Veri tabanı yöneticisi, e-mail adresini ve şifresini kullanarak giriş yapmalıdır. Veri tabanına araştırmacılara ait bilgiler (ArastirmaciID, ArastirmaciAdi, ArastirmaciSoyadi, YayinAdi, YayinYili, YayinYeri (Konferans, çalıştay, dergi adı vb.), YayinTurBilgisi vb.), veri tabanı yöneticisi tarafından web üzerinden eklenmelidir (Verilerin web sayfası üzerinden veri tabanına eklenmesi gerekmektedir. Veri tabanına manuel giriş yapılmamalıdır.). Kullanıcı Arayüzünde ise Kullanıcı arayüzü genel bir web sayfası ve arama motoru yapısında olmalıdır. Kullanıcılar bu sayfadan araştırmacı adı-soyadı, yayın adı, yayın yılına göre içerik araması yapabilmelidir.

**Yapılan Araştırmalar:**

İlk olarak programalama dili olarak ASP.NET kullanmaya karar verdik. Fakat biraz ilerledikten sonra dinamik graf yapısını oluşturmka için neovis.js kütüphanesinin kullanılacağını öğrendik. Bu yüzden ASP.NET de ilerleyemedik ve baştan başlayarak node.js kullanmaya karar verdik. Login ekranımızı yaptıktan sonra database kısmına geçiş yaptık ve Neo4j ile ilgili kapsamlı araştırmalar yaptık. Bizden istenen Veri tabanında 3 adet tablo olması, Primary Key ve Foreign Key alanlarının bulunması ve Veri tabanında oluşturulması gereken Araştırmacı Tablosu, Yayınlar Tablosu ve Tür Tablosunu oluşturduk. Veri tabanında araştırmacılar ve araştırmacılara ait akademik yayınlar bulunacak. Kullanıcılar geliştirilen web sayfası üzerinden araştırmacı ve yayın araması yapabilecek. Arama yapıldığında ekrana yayın arama sonuçları, araştırmacıların çalıştığı kişiler, yayın bilgileri, yıllara göre yayın bilgisinin gösterilmesi gerekmektedir. Araştırmacının çalıştığı kişilerin üzerine tıklandığında tıklanılan araştırmacı içinde aynı bilgilerin elde edilip gösterilmesi gerekmektedir. Bunlar için gerekli kod parçalarını yazdık. Bizi en çok zorlayan şeylerden birisi ise birlikte yayın yaptığı diğer araştırmacılar, yayın türleri, yayın yerleri ve yıl bilgisi Neo4j çizge yapısı kullanılarak görselleştirilmesiydi. Bu sekmede sadece bu bilgilerin çizge olarak görsel bir şekilde bulunması gerekmektedir. Bunun için de gerekli çalışmaları yapıp kodlarımızı yazdık.

**Yalancı Kod:**

1.Register sayfasından gelen bilgileri al.

2.VeriTabanıKaydet() fonksiyonuna parametleri ver (YazarAdi, YazarSoyAdi,YayinAdi,YayinYili,YayinTürü,YayinYeri)

2.1 Session.Start ile veri tabanına bağlan.

2.2 CreateYazaridirRelationShip() çalıştır.( Bu fonksiyon yayin ve yazar arasındaki yazaridir bağlantısını oluşturur.)

2.3 CreateTurudurRelationShip() çalıştır.(Bu fonksiyon yayin ve tür arasındaki türüdür bağlantısını oluşturur.)

2.4 CreateYeridirRelationShip()çalıştır.(Bu fonksiyon yayin ve yeri arasındaki yeridir bağlantısını oluşturur.)

2.5 Session.close ile sessionu kapat.

3.SearchTable.html sayfasında SearchNameButton a bas.

3.1Ajax Post methodu ile url /SearchName post et

3.2 Post isteğini login.js de al.

3.3 req.name ve req.surname parametlerini oku

3.4 SearchName(name,surname,res) fonksiyonunu çalıştır.

3.4.1 var liste =[] ve readQuery oluştur.

3.4.2 var veriNew değişkeni oluştur ve dönen verileri içine aktar.

3.4.3 liste.push(veriNew) ile listeye veriyi ekle

3.4.4 listeFriendEkle(liste,res) çalıştır.

3.4.5 Bir readQuery oluştur.

3.4.6 Dönen isimleri liste[i].isimler.push(friends2) ile isimler listesine at.

(Aynı projede ortak çalışmış kişileri aynı listeye atan fonksiyon)

3.5 res.send(liste) ile veriyi SearchTable.html e gönder

3.6 success: function (response) çalıştırç

3.7 Tablo elemanlarını create et ve içine gelen liste verisinedki

4.SearchTable.html sayfasında nameYayinYearButon a bas.

4.1 Ajax Post methodu ile url /SearchYayinName post et

4.2 Post isteğini login.js de al.

4.3 req.Yayinname ve req.YayinYear parametlerini oku

4.4 SearchYayinName(YayinName,res) çalıştır.

4.5 var liste =[] ve readQuery oluştur.

4.6 var veriNew değişkeni oluştur ve dönen verileri içine aktar.

4.7 liste.push(veriNew) ile listeye veriyi ekle

4.8 listeFriendEkle(liste,res) çalıştır.

4.9Bir readQuery oluştur.

4.10 Dönen isimleri liste[i].isimler.push(friends2) ile isimler listesine at.

4.11res.send(liste) ile veriyi SearchTable.html e gönder

5.success: function (response) çalıştır ve tabloya elemanları bas.

**Kaynakça:**

<https://neo4j.com/docs/>

<https://www.youtube.com/c/neo4j>

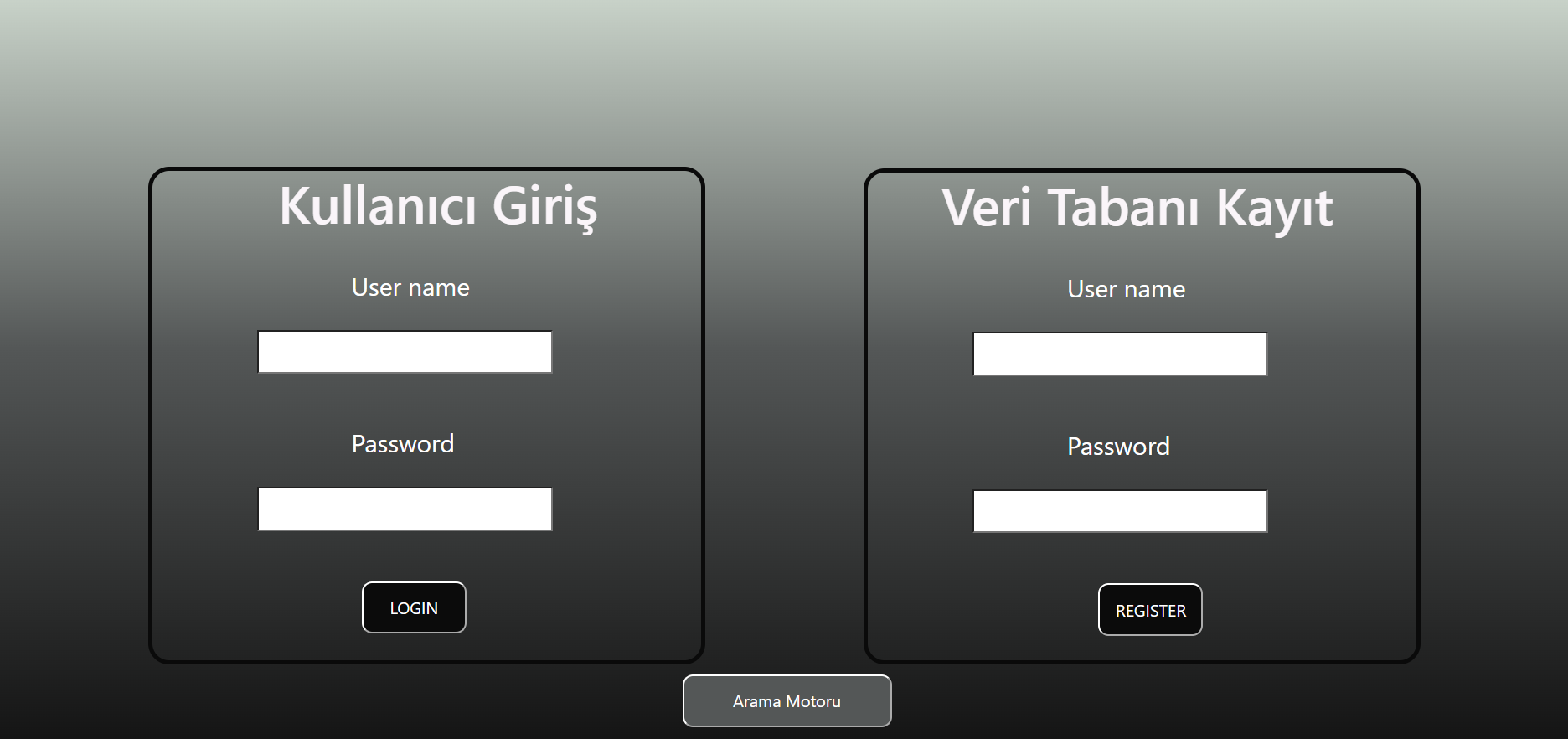
<https://www.youtube.com/watch?v=0-1A7f8993M>

<https://www.youtube.com/watch?v=jbsf4F_PpNQ>

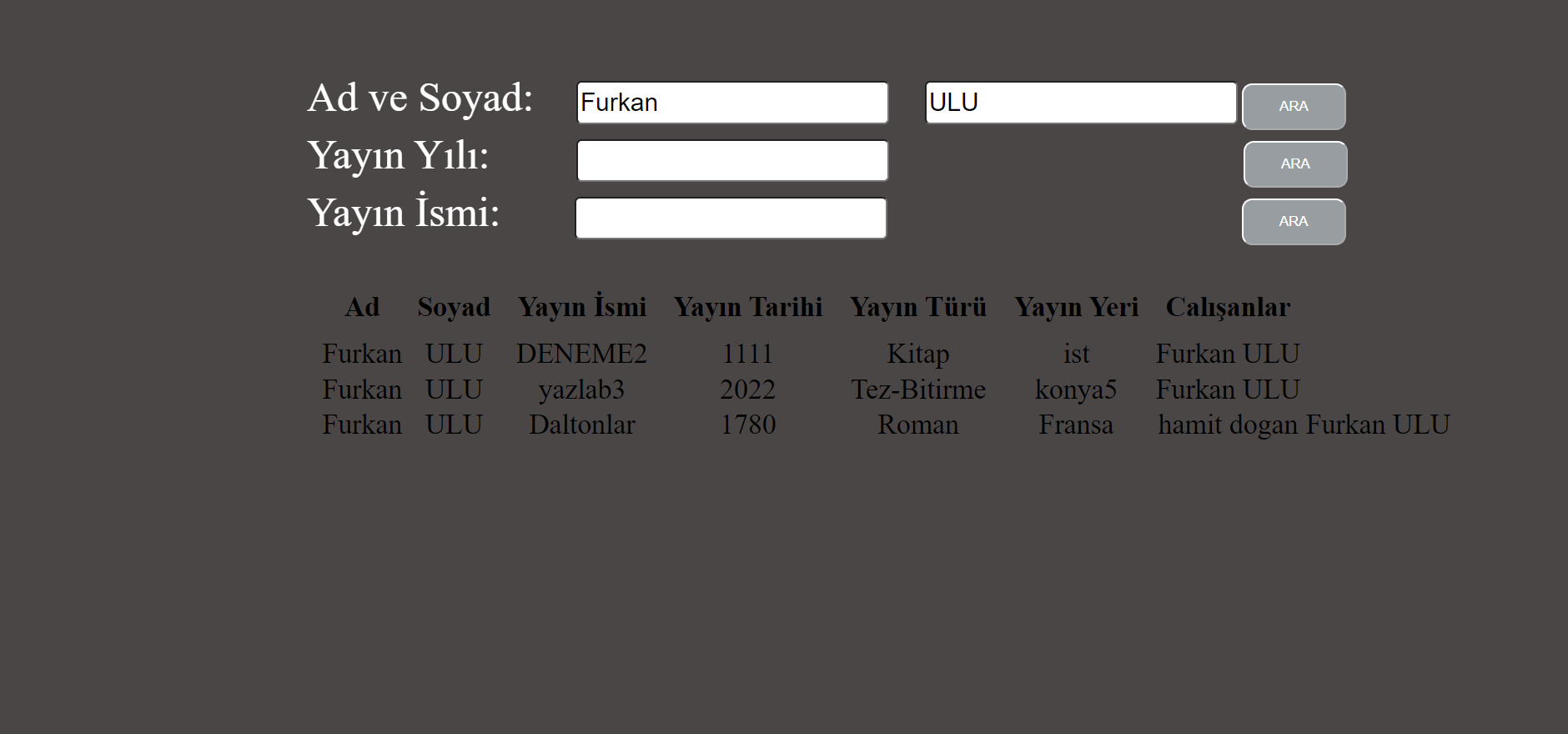
<https://github.com/neo4j-contrib/neovis.js/>

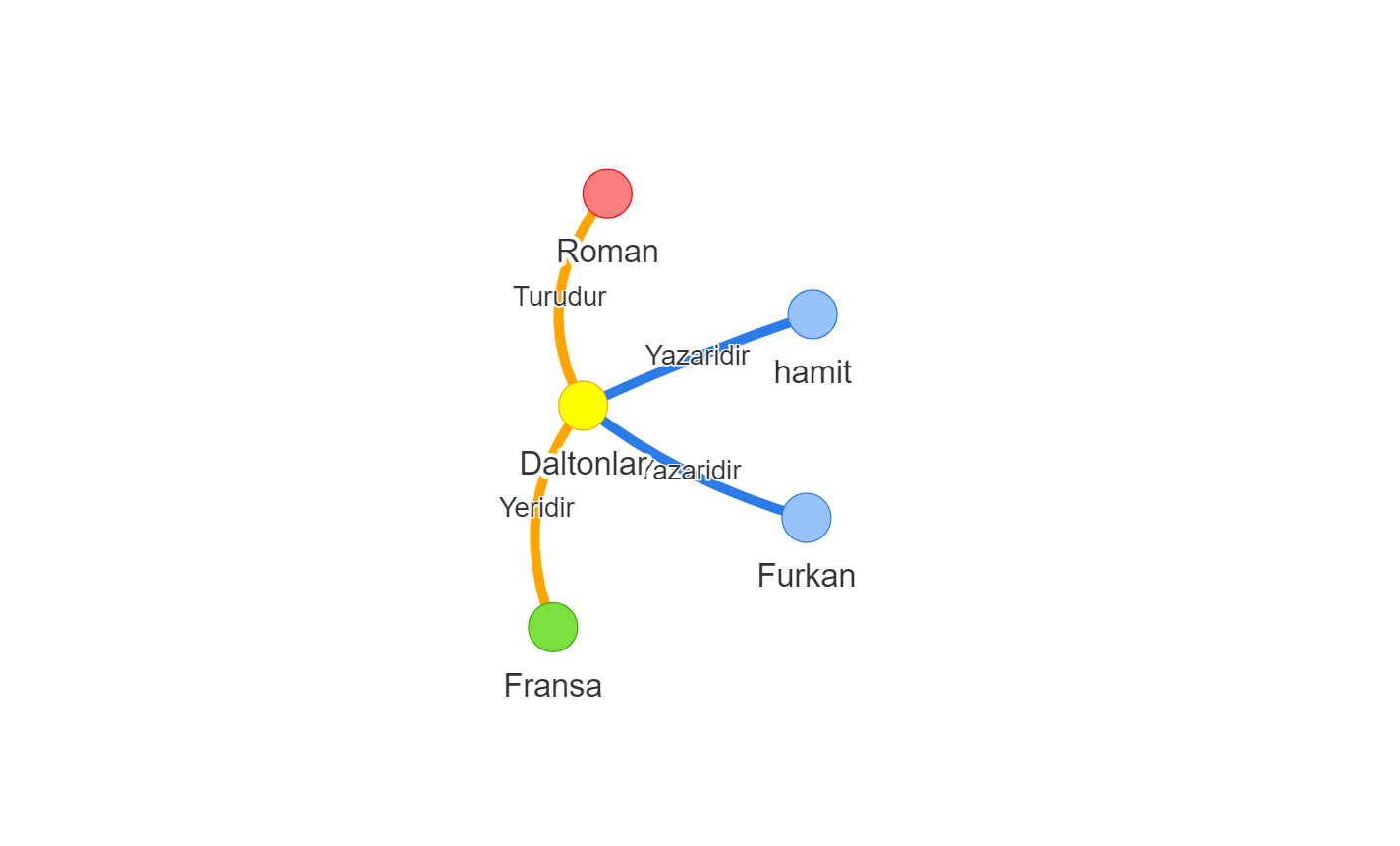
<https://stackoverflow.com/questions/72107909/how-can-i-get-properties-of-clicked-node-with-neovis>

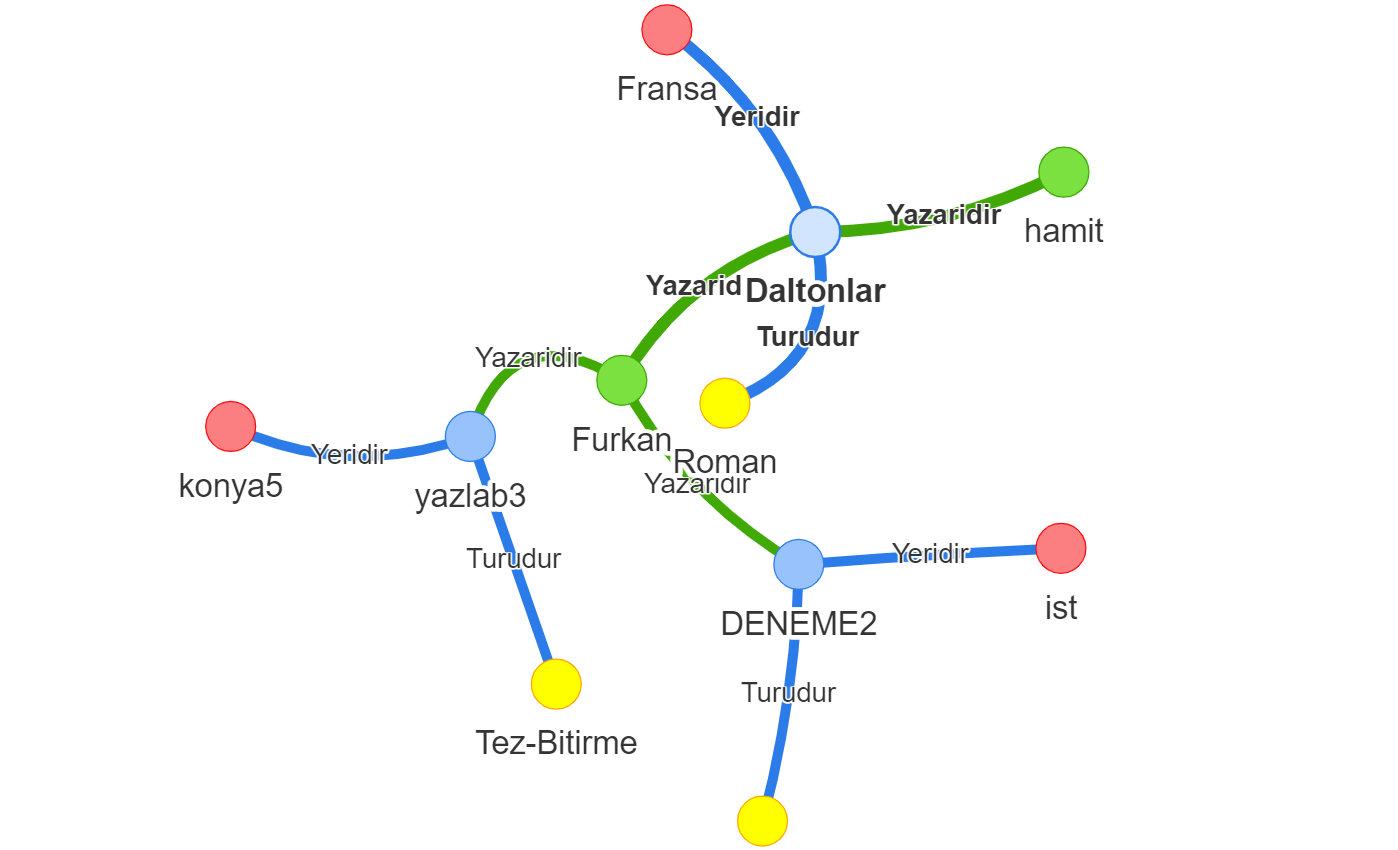
**Kod İle İlgili Ekran Görüntüleri:**

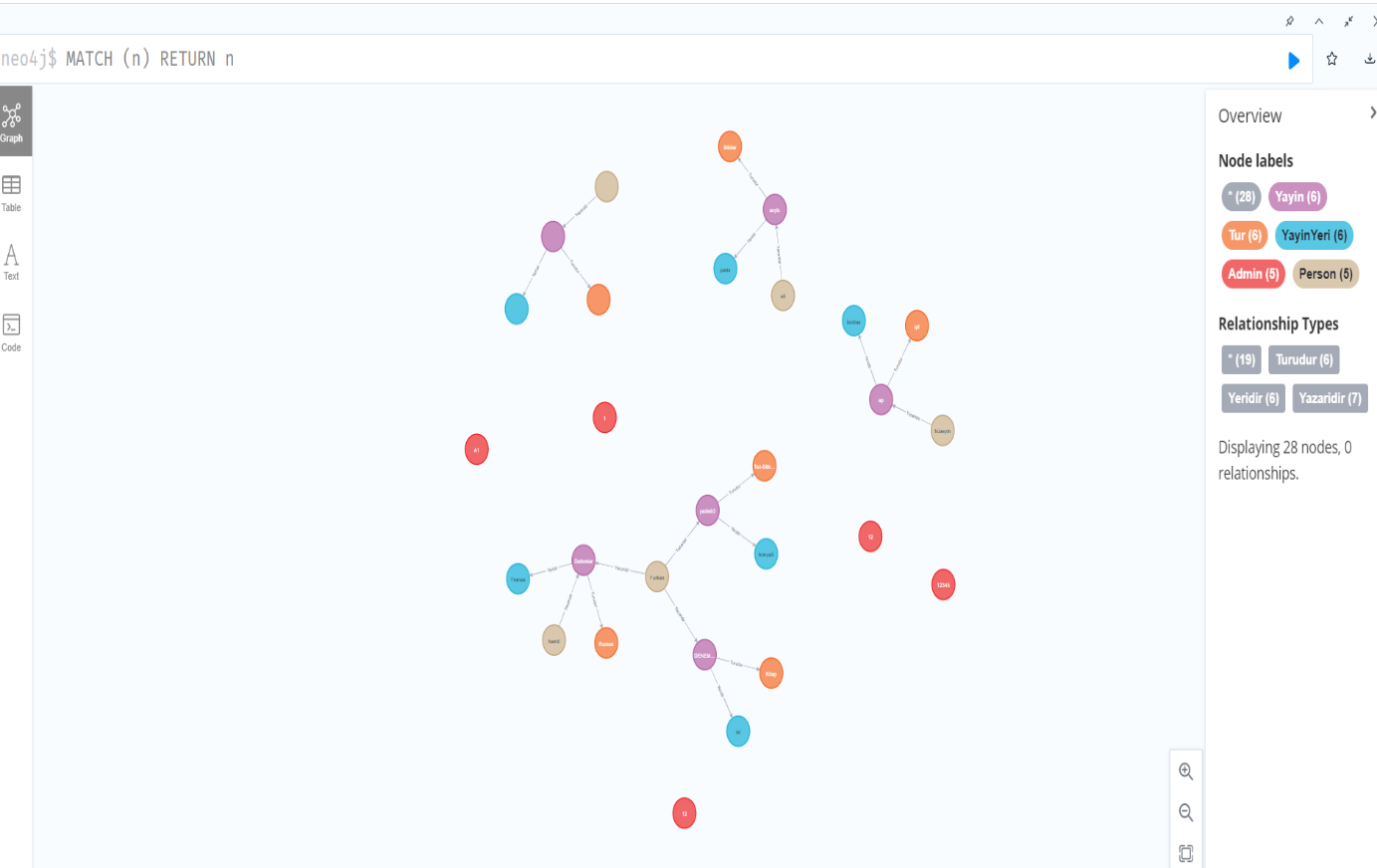
****

****

****

****

****

****