



WEB CBS TEKNOLOJİSİ, MİMARİSİ VE AÇIK KAYNAK KODLU UYGULAMA GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

MEHMET ALPER ŞAHİN

**MERSİN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**UZAKTAN ALGILAMA VE COĞRAFİ BİLGİ SİSTEMLERİ
ANABİLİM DALI**

**MERSİN
KASIM - 2022**

WEB CBS TEKNOLOJİSİ, MİMARİSİ VE AÇIK KAYNAK KODLU UYGULAMA GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

MEHMET ALPER ŞAHİN
ORCID ID: 0000-0003-0133-5278

MERSİN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

UZAKTAN ALGILAMA VE COĞRAFİ BİLGİ SİSTEMLERİ
ANABİLİM DALI

DANIŞMAN
PROF. DR. MURAT YAKAR
ORCID ID: 0000-0002-2664-6251

MERSİN
KASIM - 2022

ÖZET

WEB CBS TEKNOLOJİSİ, MİMARİSİ VE AÇIK KAYNAK KODLU UYGULAMA GELİŞTİRİLMESİ

Teknolojinin hızla geliştiği günümüzde, insanların üzerinde yaşadıkları dünyayı anlamaları ve dünya üzerindeki yerlerini bilmeleri çok önemlidir. Hızla gelişen teknolojinin de desteği ile insanların konumsal bilgilere ulaşmaları geçmiş zamanlara göre çok daha kolay olmaktadır. Günümüzde pek de pahalı olmayan mobil cihazlar üzerinde coğrafi bilgi sistemleri (CBS) uygulamaları kullanılabilmektedir. Bu cihazlar GPS aletleri, cep bilgisayarları (PDA), tabletler hatta cep telefonları olabilirler. İnternetin de yayılmasıyla coğrafi bilgi sistemleri daha da yaygın olarak kullanılmaya başlanmıştır. Bugün internet üzerinden çok güçlü coğrafi bilgi sistemleri uygulamaları (Yandex Haritalar, Bing Haritalar, Google Haritalar, OpenStreet Haritalar vb.) kullanılabilmektedir.

İnternet tabanlı CBS uygulamalarının hızla yayılmaya başladığı, ticari ve açık kaynak kodlu CBS altyapılarının rahatça bulunabildiği günümüzde, GeoServer kullanılarak tarihi mekan ve binaların bilgilerinin tutulduğu temel CBS fonksiyonlarına sahip bir uygulama gerçekleştirilebilmektedir. Benzeri uygulamanın dokümantasyonunu kapsayan bu çalışma bir CBS sisteminin nasıl olduğu ve nasıl çalıştığı hakkında genel ve temel bilgiler verecektir.

Çalışmanın bir parçası olan web uygulaması Phyton, GeoServer, PostgreSQL Server kullanılarak geliştirilmiştir ve tarihi bir mekanın coğrafi ve metinsel verilerini tutabilecek şekilde tasarlanmış ve hayata getirilmiştir. Uygulama iki parçada ele alınabilir, ilk parça metinsel verileri tutan bölümdür. Metinsel veriler PostgreSQL üzerinde saklanır. İkinci parça ise coğrafi verileri tutan bölümdür. Coğrafi veriler harita kaynak dosyalarında tutulurlar ve GeoServer üzerinden yayınlanır. Python uygulaması ile yayınlanan bu haritalar internet sayfası üzerinde görüntülenebilmektedir. İnternet sayfası üzerinden bazı temel CBS işlemleri yapılabilir.

Çalışmanın dokümantasyonunda CBS ile ilgili ayrıntılı bilgilerin yanı sıra internet teknolojileri ve internet üzerinde CBS uygulamaları geliştirmek için gereken teknolojiler hakkında da geniş bilgiler yer almaktadır.

Anahtar Kelimeler: İnternet Tabanlı CBS, CBS, Açık Kaynak, Uygulama Geliştirme, GeoServer.

Danışman: Prof. Dr. Murat YAKAR, Mersin Üniversitesi, Uzaktan Algılama ve Coğrafi Bilgi Sistemleri Anabilim Dalı, Mersin.

ABSTRACT

WEBGIS TECHNOLOGY, ARCHITECTURE AND OPEN SOURCE APPLICATION DEVELOPMENT

In this day and age of technology advancing at a rapid pace, it is very important for people to understand the world they live in and know where in the world they are. With the support of rapidly developing technology, it is much easier for humans to access spatial information than it used to be. Geographic information systems (GIS) can now be used with inexpensive mobile devices. These devices can be GPS, PDA (handheld computers), Tablets or even cell phones. With the spread of the Internet, the importance of geospatial information systems became much better understood. Today, such powerful geographic information systems can be used via the Internet (Yandex Maps, Bing Maps, Google Maps, OpenStreet Maps, etc.).

In the present day when the Web based GIS applications rapidly spread, and commercial or free source coded GIS infrastructures can be found easily, a GIS application which has the basic GIS functions that hold the information of historic places and buildings' information by using a GeoServer can be realized. This study including the documentation of this application will supply basic and general information about how a GIS system is and how it works to whom concerned about it.

The web application that is a part of the study has been developed by Python, GeoServer, PostgreSQL Server and has been designed and realized as to be able to keep a historic place's geographic and written information. Application can be seen in two parts; first part is the part that keeps the written data. Written data is kept through PostgreSQL Server. And the second part is the part that keeps the geographic data. Geographic data is kept through map source files and are published through GeoServer. Those maps that are published through Python can be viewed through web page. Several basic GIS operations can be done via web page.

Besides the detailed information of the GIS, wide information about the technologies necessary for developing the GIS applications via internet and internet technologies are included within the documentation of the study.

Keywords: Internet Based GIS, GIS, Open Source, Application Development, GeoServer.

Advisor: Professor Doctor Murat YAKAR, Department of Remote Sensing and Geographic Information Systems, Mersin University, Mersin.

TEŞEKKÜR

Yüksek lisans tez danışmanlığımı üstlenen, engin bilgi ve tecrübeleriyle desteğini asla esirgemeyen, güçlü iş ahlakı ve dürüstlüğü ile bana ilham veren değerli danışman hocam Prof. Dr. Murat YAKAR'a sonsuz teşekkürlerimi sunarım.

Araştırmamda bana rehberlik etmek için önerilerde bulunarak tezimin tamamlanmasına katkı sağlayan değerli hocalarım Dr. Öğr. Üyesi Ali ULVI ve Arş. Gör. Abdurrahman Yasin YİĞİT'e, ayrıca tez çalışmamda bilgi ve tecrübelerini eksik etmeyen Harita Mühendisi Murat ÖZDEMİR, Şehir ve Bölge Plancısı Okay ADEM, Yüksek Harita Mühendisi Mustafa Ersin BAHAR, Harita Mühendisi Fatih DERNEK, Jeoloji Mühendisi Kemal GÜNEL'e ve Ziraat Mühendisi Ahmet YALÇIN'a teşekkürlerimi sunarım.

Sonuca ulaştırılan bu çalışmanın tüm evrelerinde, benden desteğini bir an olsun eksiltmeyen, her daim yanımda olan, hayatımın her döneminde yanımda görmek istediğim ve bunun için çabaladığım kıymetli dostum, abim Harita Mühendisi Serhat ÜNVER'e sonsuz teşekkürlerimi sunuyorum.

Şu an yanımda olmasalar da desteklerini her daim hissettiğim rahmetli annem Ayten ŞAHİN, rahmetli babam Bayram ŞAHİN, rahmetli teyzem Nurten GÜCEK, rahmetli dedem Münir PARLAT'a, çalışma azmini ve başarılarını örnek aldığım, kuzenim Dr. Öğr. Üyesi Gamze ÇAMLIK, dayım Dr. Kemal PARLAT, eniştem Mehmet GÜCEK, teyzem Gülten ÇAMLIK ve anneannem Naile PARLAT'a teşekkür ederim.

Bu tez çalışması, Mersin Üniversitesi Bilimsel Araştırma Projeleri tarafından 2022-1-TP2-4644 numaralı proje ile desteklenmiştir.

İÇİNDEKİLER

	Sayfa
İÇ KAPAK	i
ÖZET	ii
ABSTRACT	iii
TEŞEKKÜR	iv
İÇİNDEKİLER	v
TABLolar DİZİNİ	vii
ŞEKİLLER DİZİNİ	viii
SİMGELER VE KISALTMALAR	ix
1. GİRİŞ	1
2. KAYNAK ARAŞTIRMALARI	2
2.1. Coğrafi Bilgi Sisteminin Tanımı ve Kavramları	2
2.2. Coğrafi Bilgi Sisteminin Bileşenleri	3
2.2.1. Veri Bileşeni	3
2.2.2. Donanım Bileşeni	3
2.2.3. Yazılım Bileşeni	3
2.2.4. İnsan Kaynağı	4
2.2.5. Yönetim Bileşeni	4
2.3. Coğrafi Bilgi Sisteminin İşlevleri	4
2.3.1. Sayısal Verilerin Entegrasyonu	5
2.3.2. Konumsal Sorgulama	5
2.3.3. Otomasyon	5
2.3.4. Görüntüleme	5
2.3.5. Veri Manipülasyonu	5
2.3.6. Mekansal Analizler	6
2.3.7. Karar Analizleri	6
2.3.8. Model Analizleri	6
2.4. Coğrafi Bilgi Sistemindeki Veri Yapıları	6
2.4.1. Temel CBS Veri Tipleri	6
2.4.2. Coğrafi Veri Bileşenleri	8
2.4.3. Coğrafi Bilgi Sisteminde Veri Toplama Yöntemleri	8
2.4.4. Ölçme	9
2.4.5. Fotogrametri Ölçümleri	9
2.4.6. Uzaktan Algılama	9
2.4.7. Sayısallaştırma	10
2.4.8. Hazır Veri Kullanımı	10
3. MATERYAL VE YÖNTEM	11
3.1. İnternet Üzerinden CBS Uygulaması	11
3.2. İnternet Tarihi	11
3.3. Ağ Teknolojileri	12
3.4. İnternet Protokolü	14
3.5. Hiper Metin Transfer Protokolü (HTTP)	14
3.6. Köprü Metni İşaretleme Dili (HTML)	14
3.6.1. HTML'nin Doğuşu ve Gelişmesi	15
3.6.2. HTML'in Son Versiyonu	16
3.7. Standard Generalized Markup Language (SGML)	17
3.8. Extensible Markup Language (XML)	17
3.8.1. XML'in Avantajları	18
3.8.2. XML'in Dezavantajları	18
3.9. XHTML	19
3.10. Veritabanı Teknolojileri	20
3.10.1. Konumsal/Mekansal/Uzamsal Veritabanları	21
3.11. Python Programlama Dili	22

	Sayfa
3.12. Javascript	23
3.12.1. Javascript' in Yetenekleri	24
3.13. Asenkron Javascript ve XML (AJAX)	24
4. BULGULAR VE TARTIŞMA	26
4.1. Açık Kaynak Kodlu Web CBS Uygulaması	26
4.2. GeoServer	26
4.3. PostGRE SQL Server	27
4.3.1. PostGIS	29
4.4. Uygulama Tasarımı	29
4.4.1. Veri Erişim Katmanı	31
4.4.2. İş Katmanı	31
4.4.3. Kullanıcı Arayüzü Katmanı	31
4.5. Proje Kodları ve Arayüzleri	31
4.5.1. Kullanıcı Giriş Ekranı Backend ve Frontend	31
4.5.2. Uygulama Anasayfası	33
4.5.3. Uygulamanın Arka Ofis (Backend) Kodları	35
4.5.4. Uygulamanın GeoServer Kodları	37
4.5.5. Uygulama İçin Yazılan DLL Listesi	39
5. SONUÇLAR VE ÖNERİLER	42
KAYNAKLAR	43

TABLÖLAR DİZİNİ

	Sayfa
Tablo 2.1. Nokta, Çizgi ve Poligon.	7
Tablo 2.2. Veri Tipleri.	8



ŞEKİLLER DİZİNİ

	Sayfa
Şekil 3.1. Bir ağın soyut görünümü.	13
Şekil 3.2. SGML örneği.	17
Şekil 3.3. XML örneği.	18
Şekil 3.4. HTML/XHTML açılış ve kapanış etiketi.	19
Şekil 3.5. AJAX nasıl çalışır.	25
Şekil 4.1. Uygulamanın ana katmanları.	30
Şekil 4.2. Kullanıcı Giriş Ekranı Frontend.	32
Şekil 4.3. Kullanıcı Giriş Ekranı çıktısı.	32
Şekil 4.4. Kullanıcı Giriş Ekranı Backend.	33
Şekil 4.5. Uygulamanın anasayfa görseli.	34
Şekil 4.6. Uygulama anasayfasının ön ofis (Frontend) kodları.	34
Şekil 4.7. Uygulama anasayfasının ön ofis (Frontend) kodları.	35
Şekil 4.8. Uygulamanın arka ofis (Backend) kodları.	36
Şekil 4.9. Uygulamanın arka ofis (Backend) kodları.	37
Şekil 4.10. Uygulamanın GeoServer yani raster verilerimizin yüklendiği, cache edildiği ve tilelar oluşturulduğu sunucumuzun kodları.	38
Şekil 4.11. Uygulamanın GeoServer yani raster verilerimizin yüklendiği, cache edildiği ve tilelar oluşturulduğu sunucumuzun kodları.	39
Şekil 4.12. DLL listesi.	40
Şekil 4.13. DLL'lerin uygulamaya eklenmesi.	41

SİMGELER VE KISALTMALAR

Kısaltma/Simgesi	Tanım
ADO	ActiveX Data Objects
AJAX	Asynchronous J avascript And XML
ARPANET	Advanced Research Projects Agency Network
CAD	Computer-Aided Design
GIS	Geographic Information System
GPS	Global Positioning System
HTML	Hypertext Markup Language
IP	Internet Protocol
LAN	Local Area Network
Mb	Megabyte
OOP	Object-Oriented Programming
PC	Personal Computer
SDK	Software Development Kit
SQL	Structured Query Language
TCP	Transmission Control Protocol
W3C	World Wide Web Consortium
XML	Extensible Markup Language

1. GİRİŞ

Konum, insanlığın üzerinde yaşadığı dünyayı anlamının anahtarıdır. Konumu bilmek ve kullanmak hayatın ritmini arttırır. Böylece insanlar çevrelerini ve bulundukları yeri bilmek isterler. Elektronik alet üreticileri ve yazılım teknolojisi şirketleri (cep telefonları, pda'lar, el tipi GPS'ler, Google Haritalar, Google Earth, Microsoft Virtual Earth gibi) teknoloji ve bilimi insanların taleplerini karşılamak için kullanır. Bu teknolojiler, onları kullanan insanlar için alışkanlık haline gelir. Bilmedikleri yerlere internetten bakmadan gitmezler ya da telefonlarının gps özelliklerini kullanarak konumlarını veya buluşma noktalarını arkadaşlarına gönderirler. Günümüzde CBS teknolojileri veya ilgili teknolojiler artan bir hızla hayatımıza giriyor ve bu teknolojileri içtenlikle karşılıyoruz.

Bu çalışmanın amacı, bir web CBS yazılımının nasıl geliştirileceğini ve sıradan bir internet tarayıcısı ile nasıl kullanılacağını göstermektir. Sonraki bölümlerde, çalışma sırasında karşılaştığım problemler ve zorluklar ortaya konulacaktır. benim yaptığım çerçevede bir CBS yazılımı geliştiren herkes bu çözümleri kullanabilir.

Bu çalışma beş bölümden oluşmaktadır.

İlk bölüm; günlük hayatta CBS'yi nasıl kullandığımızı gösterir ve bu projenin amacından bahseder. İkinci bölüm; coğrafi bilgi sistemlerinin ne olduklarını ve neler yapabileceklerini açıklar. Üçüncü bölüm; internet teknolojilerinden bahseder. Üçüncü bölüm; internet teknolojilerinden bahseder. Bu bölüm bu konuları, mevcut veritabanı sistemlerini, mevcut programlama çerçevelerini ve internet uygulaması geliştirmeyi kapsar. Dördüncü bölüm; projenin aşamalarını ve üretilen sistemi açıklar. Beşinci bölüm olan bu son bölüm, bir web CBS uygulaması geliştirecek herkes için sonuç ve önerilerden oluşmaktadır.

2. KAYNAK ARAŞTIRMALARI

2.1. Coğrafi Bilgi Sisteminin Tanımı ve Kavramları

Coğrafi Bilgi Sistemleri (CBS), coğrafya, bilgi, coğrafi bilgi ve bilgi sistemi kavramlarından oluşur.

Coğrafya; dünyanın zeminini anlatan bilim dalıdır. Coğrafya, fiziksel ve insan kaynaklı faktörlerle ilgilenir (Antenucci, 1991). Bu faktörlerin doğru bir şekilde tanımlanabilmesi için oluşumların ve konumların doğru bir şekilde belirtilmesi gerekir. Beşeri coğrafya, sosyoloji, kültür ve ekonomi gibi demografik verilerden oluşur. Ancak fiziki coğrafya, jeolojik, biyolojik ve topografik verilerden oluşur.

Bilgi; bir iş veya konu hakkındaki veriler olarak tanımlanır (Taştan, 1998).

Veri; insanlar veya bilgisayarlar tarafından iletmeye veya manipüle edilmeye uygun bir grup gerçek, kavram veya ifadedir. Veri, bilginin ham maddesi olarak tanımlanabilir (Yomralıoğlu, 2000).

Coğrafi bilgi; yeryüzünde bir konum ve şekil ifade edebilen soyut ve somut nesneler olan coğrafi varlıklara ilişkin bilgilerdir. Yeryüzündeki veya yeraltındaki her türlü doğal veya yapay somut detay (ormanlar, nehirler, binalar, yollar, enerji dönüşüm hatları vb.), yüzey topoğrafyasına ve konumuna bağlı olarak soyut detaylar (nüfus, hava kirliliği, sivil ve idari sınırlar vb.)) bu coğrafi varlık kavramlarına dahildir.

Sistem; daha önce tanımlanan ve değerlendirilmesi gereken ortak bir amaç için çevreyle olduğu kadar birbirleriyle de ilişkili olan ve birlikte çalışan personel, donanım, süreç, belge, etkinlik, veri ve varlıklar grubudur (Erden, 2001).

Bilgi sistemi; karar vermeye yardımcı olacak bilgileri oluşturmak için çalışan, depolayan ve ham verileri temsil eden süreç grubudur. Bilgi sistemleri, verileri mantıksal süreçler yardımıyla işleyen ve analiz, sorgulama ve raporlar sonucunda bir sonuca varan ve bunları çeşitli ortamlarda temsil eden ve bu verileri depolayan yapılardır.

Coğrafi Bilgi Sistemleri (CBS); donanım, yazılım ve kullanıcılardan oluşan ve tüm bu bileşenlerin grafik ve grafik olmayan bilgilerin elde edilmesi, saklanması, analiz edilmesi ve bunlardan yeni bilgiler elde edilmesi ve temsil edilmesi için organizasyonu sağlayan sistemlerdir (Yomralıoğlu, 2000).

Bir başka kaynağa göre CBS, karmaşık planlama ve yönetim problemlerini çözmek için lokasyonda açıklanan verilerin toplanması, yönlendirilmesi, çalıştırılması, analiz edilmesi, modellenmesi ve görüntülenmesini içeren donanım, yazılım ve yöntemler sistemidir. CBS, yeryüzündeki alanları tanımlayan, verileri depolayan ve kullanan bir bilgisayar sistemi olarak tanımlanabilir. Artan kullanımla birlikte CBS, özellikle coğrafi verilerin ilişkilendirilmesi için kullanıldığında birçok farklı disiplinin kullanım alanı içine girmektedir (Bartelme, 1989).

2.2. Coğrafi Bilgi Sisteminin Bileşenleri

Temel işlevleri yerine getirebilmesi için coğrafi bilgi sistemleri, şekilde gösterildiği gibi en az beş bileşen içermelidir. Bunlar:

- Veri
- Donanım
- Yazılım
- İnsan Kaynağı

2.2.1. Veri Bileşeni

Coğrafi varlıkların sahip olduğu özellikler farklı sunumlarla ifade edilebilir. CBS temel olarak grafik ve grafik olmayan verilerden oluşur. Coğrafi verilerin, doğal sınırları (nehirler, ormanlar, vadiler vb.) ve yapay sınırları (yollar, binalar vb.) olabileceği gibi, alanların (ülkeler, şehirler, parseller vb.) sınırlanmasıyla da oluşabilir.

2.2.2. Donanım Bileşeni

Donanım bileşenleri, programları çalıştıran işlem birimlerini, veri girişlerini, programları, verileri saklayan yardımcı depolama birimleri ile haritaları ve raporlar görüntüleyen monitörleri, sayısallaştırıcıları, yazıcıları ve çizicileri kapsar (Antenucci, 1991; Burrough, 1986; Davis, 1996).

Bilgisayar donanımı, merkezi işlem birimi (CPU) ve çevresel bileşenler olmak üzere iki kategoriye ayrılır. Merkezi işlem birimi, işlemleri gerçekleştirir ve yönetir. Monitör, yazıcı, çizici, tarayıcı ve bellek gibi fiziksel olarak ayrı çevre birimleri, veri depolanması, bilgi gösterilmesi, veri girişi, çıktı alınması ve iletişim etkinlikleri için kullanılır (Antenucci, 1991; Taştan, 1998; Yomralıoğlu, 2000).

2.2.3. Yazılım Bileşeni

Yazılım; coğrafi bilgilerin saklanması, analiz edilmesi ve görüntülenmesi gibi ihtiyaç ve fonksiyonları üst düzey programlama dilleri ile kullanıcıya sağlayan algoritmalarlardır. Firmaların ticari amaçla geliştirip ürettiği yazılımların yanı sıra üniversiteler ve benzeri araştırma enstitüleri tarafından eğitim ve araştırma amaçlı geliştirilen yazılımlar da mevcuttur. Dünya CBS pazarının çoğu yazılım geliştirme şirketlerinin elindedir.

Coğrafi bilgi sistemi yazılım bileşenleri; CBS yazılım paketi, uygulama yazılımı ve işletim sistemi yazılımı olmak üzere üç ana gruba ayrılır. Bilgisayarlarda işletim sistemi olarak Windows kullanılır; ancak gün geçtikçe Linux da yaygınlaşıyor.

CBS yazılımının temelleri şunlardır:

- Coğrafi veri/bilgi girişi ve süreci için gerekli araçlar

- Veritabanı Yönetim sistemi
- Konumsal sorgulama, analiz ve görüntüleme
- Ek donanım bağlantıları için arayüz desteği

ArcGIS, QGIS, SagaGIS, Intergraph, Mapinfo, Geomedia, Geographies bu tür yazılımlara örnektir.

Arayüz yazılımları, Python, C#, PHP, Perl, Visual Basic ve Java gibi programlama dilleri ile yazılmış, CBS program paketleri tarafından desteklenen ve bu paketler altında ilerleyen programlardır. Aynı zamanda, bazı CBS paket programlarında, yeni programlar yazmak için uygun APIler de bulunur.

Ayrıca CBS programlarının verilerini tutan veritabanı programları da çözüm olarak kullanılmaktadır. Bu tür programlara örnek olarak PostGRE SQL, Microsoft SQL Server, MySQL, Oracle ve Microsoft Access verilebilir.

2.2.4. İnsan Kaynağı

CBS personeli; sistem geliştirme ekibi, sistem ve veritabanı yöneticileri ile CBS kullanıcılarından oluşur. CBS kullanıcıları, bilgi ürünü kullanıcıları (karar vericiler vb.), operatörler ve bilgi toplamak için veri toplayıcılar olup sürekliliği sağlayanlardır (Batuk, 1995).

2.2.5. Yönetim Bileşeni

Başarılı bir CBS, iyi tasarlanmış planlara ve çalışma kurallarına göre çalışır. Bu tür işlevler her kuruma ve projeye özel uygulamalar şeklindedir. CBS'nin kurumlar arası etkileşimi veya kurumlar arası konumsal bilgi aktarımı etkin bir şekilde sağlanmalıdır. Standartların ve kuralların belirlenmesi şeklindeki uygulamalar, enstitünün organizasyon yapısı ile doğrudan ilgilidir. Yönetim Bileşeni, her bir CBS bileşeninin belirlenen standart ve kurallar dahilinde yönetilmesini ve denetlenmesini içerir (Yomralıoğlu, 2000).

2.3. Coğrafi Bilgi Sisteminin İşlevleri

CBS, zemin yapılarını ve zeminde meydana gelen olayları haritalamak ve analiz etmek için gerekli olan bilgisayar destekli araçlardan oluşan bir sistemdir. CBS teknolojisi, ilişkili veritabanlarını birleştirme yeteneğine sahiptir. Haritaların sağladığı görsel ve coğrafi analizler, sorgulama ve istatistiksel olarak kullanıcıya sunulmaktadır. Bunlar CBS'nin diğer sistemlerden farklı olarak kullandığı fonksiyonlardır (Yomralıoğlu, 2000).

2.3.1. Sayısal Verilerin Entegrasyonu

CBS, farklı ortamlarda geliştirilen sayısal ve grafik olmayan verilerle entegre çalışabilme özelliğine sahiptir. Örneğin, CAD yazılımları tarafından oluşturulan grafik veriler, fotoğraflar ve benzeri görsel veriler, veri tabanında bulunan çizelge ve çizelge envanteri CBS tarafından veri girişi olarak kullanılabileceği gibi, CBS tarafından üretilen veriler de diğer sistemler tarafından veri girişi olarak kullanılabilir.

2.3.2. Konumsal Sorgulama

Daha sonra ihtiyaç duyulması halinde toplanan coğrafi verilere ulaşmak için genellikle veri tabanı yönetim sistemleri kullanılmaktadır. Ancak hem grafik hem de grafik olmayan bilgileri bir arada görmek ancak CBS ile mümkündür. Bu sayede sayısal verilerden sözel verilere ve grafik olmayan bilgilerden grafik bilgilere hızlı bir şekilde geçiş yapılabilmektedir.

2.3.3. Otomasyon

İşin hassasiyeti CAD ve haritalama işlerinde ölçümlere bağlıdır ve bu işlemlerin klasik yöntemlerle yapılması genellikle hatalara neden olur. CBS, grafik özelliği ile kullanıcılara bilgisayar destekli kullanım imkanı, ölçme ve hesaplama gerektiren işlemlerde otomasyon kolaylığı sağlamakta ve bu sayede hem hesaplama işlemleri hem de grafik çizimleri aynı ortamda hızlı ve doğru bir şekilde yapılabilmektedir.

2.3.4. Görüntüleme

CBS'nin önemli özelliklerinden biri de görüntüleme özelliğine sahip olmasıdır. Daha önce veritabanlarında sadece sayısal olmayan bilgilerin sunumu mevcutken, şimdi CBS ile sayısal bilgileri, sözel verileri, videoları, sesleri, fotoğrafları, istatistiksel projeksiyonları ve diğer birçok benzer görünümü görüntülemek mümkündür.

2.3.5. Veri Manipülasyonu

Konumsal veriler kullanılarak yapılan projelerde en büyük problem verilerin güncellenmesi, tasnif edilmesi, eklenmesi, aktarılması gibi yapılamayan manipülasyonlardır. CBS ile konumsal verileri hızlı ve doğru bir şekilde manipüle etmek mümkündür. Bu sayede mevcut bilgilerden yeni bilgiler toplanarak istenilen formatta bilgi üretilmesi ve farklı sistemlere bilgi aktarımı gerçekleştirilebilmektedir.

2.3.6. Mekansal Analizler

Sayısal ve sözel verilerin bir amaç için açıklanması ve yorumlanması gibi tüm işlemler konum analizi olarak bilinir. Mekansal analiz süreçlerinde, mevcut veri kümelerini kullanarak yeni kümeler oluşturma, coğrafi özellik gösteren alanların potansiyel kullanımlarının değerlendirilmesi, mekansal olayların çevre üzerindeki etkilerinin tahmin edilmesi ve bu olayların yorumlanarak anlaşılır hale getirilmesi gibi tüm uygulamalar mekansal analiz kapsamına girer.

2.3.7. Karar Analizleri

Konuma bağlı olarak coğrafi verilerin toplanmasının yanı sıra zamana veya aynı konuma ait farklı özelliklere bağlı bilgilerin sağlanması büyük kapasiteli verilere neden olmaktadır. Bu durum, özellikle envanter ve istatistik süreçlerine altlık oluşturmak için ortaya çıkar. Temel istatistiksel analizlere ek olarak, mevcut verilere göre tahminler, ticari alanların belirlenmesi, planlama için gerekli donatıların en iyi alanlara yerleştirilmesi, operasyonel analizler, istatistiksel açıklama gibi birçok "neden" ve "ne için" sorularına cevap veren karakterize karar analizleri toplu verilerin toplanması ve zamana göre konum değişikliklerinin izlenmesi CBS ile çok daha dinamiktir. CBS bu tür verileri toplayarak ve önceden belirlenen niteliklere göre sınıflandırarak konumsal bilgilerin grafik desteği ile daha iyi anlaşılmasında önemli bir işlev görmektedir.

2.3.8. Model Analizleri

Önceden planlanmış bazı projelerin veya gerçek olaylar gibi doğal olayların gerçekleşmesi durumunda meydana gelen koşulların ortaya çıkmadan önce gözlemlenmesi işlemine simülasyon denir. CBS, coğrafi varlıkların çevreleriyle olan ilişkisini göz önünde bulundurarak gerçek modeller kullanarak simülasyon yapabilir. Örneğin deprem, sel, taşkın, erozyon gibi olayların simülasyonları veya yol, demiryolu, boru hattı güzergahı gibi projelerin hazırlanması, koordinatlara bağlı olarak yeni yerleşim alanlarının planlanması gibi.

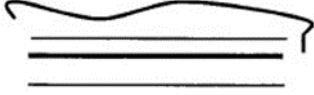
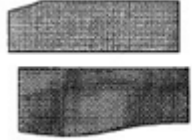
2.4. Coğrafi Bilgi Sistemindeki Veri Yapıları

2.4.1. Temel CBS Veri Tipleri

- Vektör Veri
- Raster Veri
- Öznitelik

Vektör verileri, bir koordinat sistemini referans olarak benimser ve sistem uzayındaki koordinatlarda yer alır. Örneğin; Bir A noktasının uzaydaki konumu, Kartezyen koordinat değerleri (x, y, z) veya coğrafi enlem ve boylam koordinat değerleri (f, λ) ile kesin olarak tanımlanır. Söz konusu detay bir "nokta" ise bir koordinat değeri yeterlidir, ancak bir çizgi veya alan için Tablo 2.1'de gösterildiği gibi bir koordinat satırına ihtiyaç vardır (Davis, 1996).

Tablo 2.1. Nokta, Çizgi ve Poligon.

		
1) Nokta	2) Çizgi	3) Poligon
X1,Y1	X1,Y1;X2,Y2;X3,Y3	X1,Y1;X2,Y2;X3,Y3;Xi,Yi
Sıfır boyutlu, uzunluk ve alan bilgisi olmayan ve tek bir koordinat değeri (x, y) ile gösterilen coğrafi detaylar.	Başlangıç ve bitiş noktası olan ancak alan bilgisi olmayan coğrafi detaylar koordinat satırından oluşur.	Başlangıç ve bitiş noktası aynı, uzunluk ve alan bilgilerini bir arada bulunduran koordinat satırından oluşur.
Ağaçlar, direkler vb.	Yollar, nehirler vb.	İnşaat alanları, meydanlar vb.

Vektör bilgileri, koordinatlarla ifade edildiğinden detayların boyutu hakkında bilgi verir. Bu grafik bilgi, farklı ortamlarda, örneğin kağıt üzerinde, formlar açısından aynı görünüyor. Haritalar bunlara açık bir şekilde örneklenebilir. Haritalar genellikle coğrafi bilgi sistemlerinde grafik bilgi olarak adlandırılır.

Raster veriler daha çok süreklilik özelliği taşıyan coğrafi varlıkları ifade etmek için kullanılmaktadır. Raster görünüm, komşu grid yapılı aynı büyüklükteki hücrelerin bir araya toplanmasıyla oluşur. Her hücre piksel olarak tanımlanır. Fotoğrafik görünüme sahip raster modeller, genellikle fotoğraf veya harita taranarak toplanır. Koordinat bilgilerini içermezler. Ancak matrisi oluşturan dikdörtgenin enine iki kenarının veya dört kenarının koordinatları hesaplanarak koordinat görüntüsü (Georeferanslı Görüntü) oluşturulabilir. Böylece koordinatlı raster üzerinde ölçümler yapmak mümkün hale gelir. Raster klasörler grafik dışı bilgiler içermediği için sorgulama işlemi yapılamaz. Ancak uzaktan algılama yöntemi ile renk değerlerinden sonuçlar alınabilir. Varlıkların koordinat bilgilerinin yanı sıra, bu varlıklar arasındaki ilişkiler ve bunların özellikleri de gereklidir. Bunlar genellikle grafik bilgileri tanımlayan ve coğrafi varlıkların niteliklerini içeren yazılı bilgilerdir. Nitelik bilgisi, özelliklerin grafiksel olarak ifade edilemeyen ancak yazılı ve şekilsiz ifadeleridir. Örneğin bir nokta gerçekte elektrik direği gösteriyorsa cins, yükseklik, kuruluş tarihi, son bakım tarihi, uzunluk ve renk gibi bilgiler grafiksel olmayan bilgilerdir. Grafik olmayan bilgiler genellikle yazılı tablolar belgesi olarak toplanan verilerden oluşur.

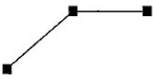
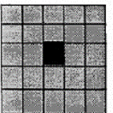
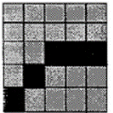
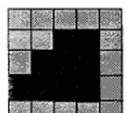
2.4.2. Coğrafi Veri Bileşenleri

Coğrafi veriler haritalarda grafik olarak gösterildiğinden, yoğunlukları göz önüne alındığında birçok karmaşık şekil ortaya çıkar. Bu karmaşık yapı, özellikle grafik verilerin sınıflandırılması gerektiğini göstermektedir. Bunun nedeni, grafik verilerin geometrik olarak tanımlanmasını sağlayarak bilgisayar ortamında saklanmasını sağlamaktır. Coğrafi veriler üç ana unsurdan oluşur. Bunlar;

- Nokta, düğüm
- Çizgi, çoklu çizgi, yay
- Alan, poligon

olarak bilinen coğrafi veri bileşenleridir. İncelendiğinde gerçek dünyadaki varlıkların Tablo 2.2'de gösterilen sadece üç temel yapıda olduğu görülmektedir. Örneğin, noktalarla ifade edilen ağaçlar, elektrik direkleri, otobüs durakları vb. coğrafi detaylar; nehirler, yollar, boru hatları vb. çizgilerle ifade edilir; binalar, parseller, göller vb. alanlarla ifade edilir. Veri bileşenleri bir koordinat değeri veya bir dizi koordinat ile ifade edilirken, grafik olmayan bilgiler veri tabanında öznitelikler şeklinde tutulur.

Tablo 2.2. Veri Tipleri.

Detay Türü		
Nokta	Çizgi	Alan
		
X_1, Y_1	$X_1, Y_1; X_2, Y_2; X_3, Y_3$	$X_1, Y_1; X_2, Y_2; X_3, Y_3; X_4, Y_4$
		
<pre> 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 </pre>	<pre> 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 0 </pre>	<pre> 0 0 0 0 0 0 0 1 1 0 0 1 1 1 0 1 1 1 1 0 </pre>

2.4.3. Coğrafi Bilgi Sisteminde Veri Toplama Yöntemleri

Coğrafi bilgi sistemlerinde bilgi toplanması, uzun zaman alan, birçok zorluğu ve para yükünü beraberinde getiren ilerlemelerden biridir. Coğrafi bilgi sistemlerinde veri toplama, konum analizlerinin potansiyelini artırmak için birçok klasörü birbirine bağlamaktadır. İdeal koşullarda bağlanan tüm veriler aynı tarihte toplanacak, tanımlanacak, ölçülecek ve birbirini takip ederek aynı yöntemle coğrafi bilgi sistemine dahil edilecektir. Kullanıcılar, klasörleri bağlamak için aynı yöntemleri kullanırlarsa

sonuçların doğruluğundan emin olabilirler. Uygulamada, veri toplamak genellikle ideal olmaktan uzaktır.

Coğrafi bilgi sistemlerinde veri toplama aşağıdakiler gibi sıralanabilir:

- Ölçme
- Fotogrametri
- Uzaktan Algılama
- Sayısallaştırma
- Hazır Veri Kullanımı

2.4.4. Ölçme

Yüzey ölçümleri ile büyük ölçekli topografik veriler toplanabilir. Yüzey ölçüm cihazlarının teknolojiye paralel olarak gelişmesiyle yüzey verilerinin coğrafi bilgi sistemlerine entegrasyonu daha kolay ve hızlıdır (Erden, 2001). GPS teknolojisi ile daha büyük alanlar daha kısa sürede ve daha doğru ölçülür.

2.4.5. Fotogrametri Ölçümleri

Fotogrametri, nesnelerin boyut, konum ve şeklini fotoğraflarına göre tanımlayan bilim dalıdır. Fotogrametrik ölçme ve değerlendirme sonucunda üç tip çıktı elde edilebilir. Bunlar aşağıda listelenmiştir.:

- Nesnelerin koordinatlarının belirlenmesi (x, y, z)
 - Topografik harita ve planların yapılması
 - Bu fotoğraflardan dikey fotoğraflar elde etmek veya harita üretmek
- (Burrough, 1989; Herbert, 2005; Yomralıoğlu, 2000).

2.4.6. Uzaktan Algılama

Uzaktan algılama, varlıklara herhangi bir fiziksel temas olmaksızın sensörler tarafından alınan görüntülerden bilgi toplanması olarak tanımlanabilir. Dijital olarak işletilmesi, hızlı ve ekonomik olması, küresel olması, güncellenebilmesi ve diğer verilerle entegre olabilmesi, uydu verilerinin başka amaçlar için kullanılmasını sağlar. Çok zamanlı, kolay erişilebilir ve geniş alanların dahil edilmesi, bu verilerin durum analizi, denetim, planlama ve kontrol için kullanılabilirliğini artırdı.

2.4.7. Sayısallaştırma

Analog veriler, dijital olmayan kağıt ortamdaki verilerdir. Bunlar genellikle kağıda çizilen haritalar, planlar ve istatistiki tablo verileri, defter ve benzeri ortamlarda yazılı veriler ve basılı fotoğraflardan oluşur. Analog verilerin coğrafi bilgi sistemlerinde kullanılabilmesi için sayısallaştırılması gerekmektedir. Bu amaçla kullanılan yöntemler şunlardır:

- Klasik Sayısallaştırma
- Otomatik Sayısallaştırma
- Tarama Sayısallaştırma
- Monitörde Dijitalleştirme

(Yomralıoğlu, 2000).

2.4.8. Hazır Veri Kullanımı

Coğrafi bilgi toplamak pahalı ve zor olduğu için, başka bir kurum tarafından üretilen veriler, bilgilerin güncel durumu ve doğruluğu doğrulandıktan sonra kullanılabilir. Sürekli güncel tutulması gerektiğinden, tüm verilerin tek bir kurum tarafından üretilmesi zordur. Günümüzde birçok kurum sayısal verileri paralı veya ücretsiz olarak üretmekte ve temsil etmektedir. Ancak içerdikleri yapı, doğruluk, duyarlılık ve bilgiler dikkate alınmalı ve analizlerden sonra kullanılmalıdır.

3. MATERYAL VE YÖNTEM

3.1. İnternet Üzerinden CBS Uygulaması

İnternet, sonsuz etkileşimli bir ortam ve bilgi kaynağıdır. Birilerinin bu bilgiyi üretmesi ve internet üzerinden sunması gerekiyor ki, diğerleri ona ulaşabilsin ve kendi amaçları için kullanabilsin. Son yıllarda birkaç çevrimiçi CBS yazılımı ortaya çıkmıştır (Google Haritalar, Yandex Haritalar, Bing Haritalar, OpenStreet Haritalar vb.). Bu çevrimiçi uygulamalar gerçekten de günlük hayatımızda oldukça kullanışlıdır. Bu güçlü CBS uygulamaları, CBS verilerini temel görüntülere ve HTML koduna dönüştürür ve sıradan bir internet tarayıcısının kodu kolayca yorumlayabilmesi ve çıktı haritalarını ve grafiklerini gösterebilmesi için bu formda hizmet verir. İnternet teknolojisi kendini yeniden icat etti ve inanılmaz derecede farklı bir seviyeye geldi ama tüm bunların altında programlama dilleri 5 veya 6 yıl öncesinden gerçekten farklı değil. O zamanlar da HTML ve Javascript kullanıyorduk, günümüzde de hala HTML ve Javascript kullanıyoruz. Aynı programlama dillerini kullanmamıza rağmen kullanım biçimlerimiz yıllar içinde oldukça değişti. Bu bölümde internet teknolojileri ve internet programlama dilleri olabildiğince geniş bir şekilde anlatılacaktır.

3.2. İnternet Tarihi

İnternet, bireysel ağların oldukça gevşek bir topluluğudur; genel yönetim yolunda çok az şey var. Bireysel ağlar, çok sayıda bağımsız operatöre aittir. Bunlardan bazıları, büyük, yüksek kapasiteli ağlara sahip büyük şirketlerdir; diğerleri evlerinde iki veya üç bilgisayardan oluşan küçük ağları işleten özel kişilerdir. Bu ağlar icat edilmiş bütün teknolojileri kullanır. İnternetin en büyük gücü, bu farklı ağların tek bir küresel ağ hizmeti sağlamak için birlikte hareket etmesine izin vermesidir. Bir ağ ve komşuları arasındaki etkileşimler özünde hem basit hem de sağlamdır. Bu, kolay genişletilebilirlik sağlar ve İnternet'in erken büyümesini hızlandırır. Yeni katılımcıların yalnızca mevcut bir operatörle bir anlaşmaya varması ve tam oyuncu olmak için oldukça basit bir ekipman kurması gerekiyordu. Bu, operatörlerin çoğunlukla büyük ve bürokratik olduğu ve yeni ara bağlantıların eklenmesinin karmaşık müzakere ve konfigürasyon ve muhtemelen uluslararası anlaşmalar gerektirdiği telefon ağları dünyasındaki durumla büyük bir tezat oluştuyordu (Bidgoli, 2003).

İnternet hikayesi 1960'ların başında, verileri sürekli bit akışları yerine ayrı kümeler ("paketler" olarak adlandırılır) halinde bilgisayar ağları aracılığıyla gönderme fikrinin icadıyla başladı. Genel olarak, bilgisayar dosyaları gibi büyük nesnelerin birkaç pakete bölünmesi ve hedefte yeniden birleştirilmesi gerekir. Paket anahtarlama büyük bir erken deney, Leonard Kleinrock, Lawrence G. Roberts ve diğerleri tarafından başlatılan ve ABD Savunma Bakanlığı tarafından desteklenen ARPANET'ti. 1969'un sonunda, ARPANET çalışır durumdaydı ve dört bilgisayar birbirine bağlıydı. ARPANET önümüzdeki birkaç yıl içinde büyüdü ve dosya aktarımı ve elektronik posta gibi

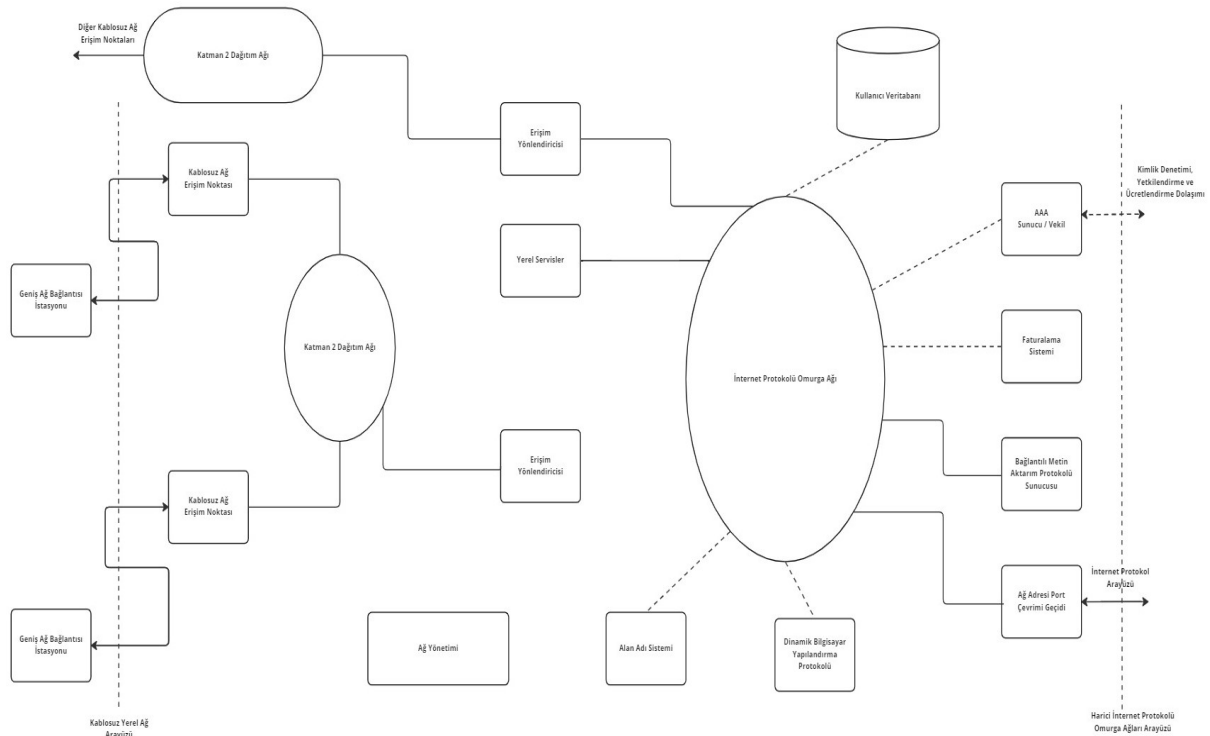
uygulamaları desteklemeye başladı. 1972'de ARPANET tasarımında etkili olan Bob Kahn, dikkatini ağ ara bağlantısına veya "internetlemeye (interneting)" çevirmeye başladı. Kahn'ın temel kavrayışlarından biri, gelecekte her biri farklı bir teknolojiye dayanan birçok bağımsız ağ olacağıydı. Tüm bu ağların ARPANET'in işleri yapma şekliyle uyumlu olmasını gerektirmenin imkansız olacağını fark etti. Bunun yerine açık bir "İnternet Çalışması Mimarisi" tasarladı. Esasen, ağlar dahili olarak istediklerini yapabilirler, ancak diğer ağlarla birlikte çalışmak istiyorlarsa, birlikte çalışma mimarisini de uygulamaları gerekir. Kahn, Vint Cerf ile işbirliği içinde bu temel fikirleri detaylandırmaya devam etti. Ağlar arasında güvenilir veri iletimi sağlamak için iki bilgisayarın nasıl işbirliği yapması gerektiğini belirleyen "iletim kontrol protokolü" (TCP) adı verilen bir dizi kural geliştirdiler. Kısa bir süre sonra, tüm uygulamaların aynı düzeyde güvenilirlik garantisi istemeyeceği anlaşıldı, bu nedenle TCP ikiye bölündü: bir parça güvenilirlik sağladı ve TCP adını korudu; diğer kısım ise paketlerin teslimi ile ilgiliydi ve "İnternet Protokolü" (IP) olarak adlandırılıyordu. TCP ve IP, 1970'lerin başlarında tasarımcılarına bir övgü olarak bugüne kadar en önemli açılardan değişmeden hayatta kaldı. Kahn ve Cerf'in fikirleri kısa süre sonra ARPANET'in kendisinden, bir paket radyo ağı ve bir paket uydu ağından oluşan üç ağı bir "internet"te uygulamaya kondu. İkincisi, Peter Kirstein tarafından yönetilen Birleşik Krallık'taki University College London'da bir düğümle uluslararası bir erişim olan ağlara verdi. Başlangıçta bir gün belki 100 katılımcı ağ olabileceği düşünülüyordu. 1980'lerde yerel alan ağlarının (LAN'lar) icadı her şeyi değiştirdi ve bugün yüzbinlerce olan katılımcı ağların sayısında patlayıcı bir büyüme getirdi. 1986'da ABD Ulusal Bilim Vakfı, beş süper bilgi işlem merkezini birbirine bağlayan bir omurga ağını finanse etti. Bu ağ (NSFNET), dünyanın birçok ülkesine bağlantıları olan ilk gerçek İnternet omurgası haline geldi. 1995 yılına gelindiğinde, İnternet'in uzun yıllar boyunca araştırma ve akademik ağ olarak önemli bir ticari geleceği olduğu ve NSFNET'in omurga rolünün bir dizi bağlantılı ticari ağ sağlayıcısına devredildiği açıkta (Bidgoli, 2003).

3.3. Ağ Teknolojileri

İnternetin temel bileşenleri "paket ağları"dır (bu bölümün geri kalanında kısaca "ağlar" olarak adlandırılır). Bu terim, veri paketlerini bir bilgisayardan diğerine aktarabilen herhangi bir teknolojiyi belirtir; Şekil 3.1, böyle bir ağın soyut bir görünümünü göstermektedir. Genellikle, birçok bilgisayar ağa bağlıdır, bu da onlara paketlerin kaynağını ve hedefini belirtmek için kullanılabilecek "adresler" sağlar. Ağlara bağlı bilgisayarların çoğu "ana bilgisayar" olarak sınıflandırılır. Bunlar, çalışan programlardan çok bilgisayarlardır; masaüstü bilgisayarlar ve sunucular bu kategoriye girer. Birkaç bilgisayar, yalnızca ağ işlemiyle ilgili işlevler sağlar. Örneğin, "ağ geçitleri", "yönlendiriciler" ve "köprüler" ağlar arasında paketleri aktarır. Bu genel modele uyan çok çeşitli teknolojiler vardır. Örneğin, Ethernet (Metcalfe & Boggs, 1976) gibi LAN'lar, orta erişim kontrolü (MAC) adreslerini kullanır. Bunlar, ana bilgisayarlara takılan arabirim kartlarının üreticileri tarafından tahsis edilen 48 bitlik adreslerdir. Bir LAN tarafından sağlanan "ağ hizmeti" basittir: bir bilgisayar, hedef MAC adresi içeren bir paket oluşturur ve ağdan bunu

tıpkı bir mektup gönderir gibi teslim etmesini ister. Buna "bağlantısız" (CL) hizmeti denir. Örneğin, asenkron transfer modu (ATM) (Vetter, 1995) ağları tarafından oldukça farklı bir hizmet tarzı sağlanmaktadır. Bu ağlarda, bilgisayarın ağa ne tür veri göndermek istediğini ve hedefin adresini söylediği bir ilk görüşme aşaması vardır. Bu, bilgisayara ihtiyaç duyduğu hizmet kalitesini (QoS) (iş hacmi sınırları, aktarım gecikmesi vb.) belirleme fırsatı verir. Ağ, talebi karşılayıp karşılayamayacağına karar vermelidir. Yapabiliyorsa, bilgisayara hedefle bir "bağlantı" kurulduğunu ve veri aktarımının başlayabileceğini söyler. Bu, kişinin bir numarayı çevirdiği ve konuşma başlamadan önce bir arama yaptığı bir telefon görüşmesi yapmaya benzer ve "bağlantı yönelimli" (CO) hizmet olarak adlandırılır. CO ağlarındaki adresler yalnızca bağlantı kurulum aşamasında kullanılır (Farrel, 2004).

Hemen hemen tüm ağ teknolojileri, hizmeti uygulamak için kullandıkları mekanizmalar kökten farklı olsa da, az önce açıklananlara benzer hizmetler sağlar. Bununla birlikte, bu geniş kategoriler içinde pek çok garip farklılık vardır: Her teknoloji, gönderilebilecek bir paketin boyutuna kendi sınırını getirir, her birinin kendine özgü adresleme şeması vardır, bazı teknolojiler QoS'nin ince bir şekilde kontrolüne izin verir, diğerleri vermez, vb. . Bu farklılıklar ağ ara bağlantısı için zorluklar yaratır. Bir ağdan paket alıp diğerine iletemeyiz; her türlü ince dönüşüm gerekli olacaktır. Çözüm "yakınsama"dır. Soyut bir ağ hizmeti icat ediyoruz ve her ağ teknolojisi için soyut hizmeti uygulamak için "yakınsama işlevleri" sağlıyoruz. Bu sayede tüm ağların aynı görünmesini sağlıyoruz ve ara bağlantı sorununu yönetilebilir hale geliyor (Anderberg, 2002).



Şekil 3.1. Bir ağın soyut görünümü.

3.4. İnternet Protokolü

IP bir yakınsama işlevi sağlar. Protokolün 4. versiyonu (hala en yaygın kullanılanı) ve uyguladığı soyut ağ hizmeti, 1981'de yayınlanan RFC 791'de (Postel, 1981a) belirtilmiştir. Hizmet, tek paketlerin (IP datagramları olarak adlandırılır) "IP adresleri" ile tanımlanan hedeflere gönderilmesine izin veren basit bir CL hizmetidir. IP adresi, iki bölüme ayrılmış 32 bitlik bir miktardır. İlk kısım, "ağ öneki", bir "IP ağını" tanımlar ("IP alt ağı" olarak adlandırılır). İkinci kısım, o ağdaki bir arayüzü tanımlar. 32 bit normalde noktalarla ayrılmış dört ondalık bayt olarak yazılır (örn. 128.16.5.2). IP adresinin ağ öneki bölümünün 16 bit olduğunu varsaydık. Bu, her ağdaki ayrı arabirimleri tanımlamak için toplam 216 (= 65,536) olmak üzere 16 bit'e sahip olabileceğimiz anlamına gelir; bu, bir ağda sahip olunabilecek en çok arabirim sayısıdır. Bu nedenle, IP tasarımcıları küçük, orta ve büyük ağların gereksinimlerine uygun üç sınıf IP adresi olması gerektiğine karar verdiler. Küçük ağlar için, ağ öneki 24 bit ve ana bilgisayar bölümü 8 bittir ve 256 ana bilgisayara izin verir (aslında, biri "bu ana bilgisayar" için ve diğeri yerel yayın için kullanıldığından 254'tür) (Farrel, 2004).

3.5. Hiper Metin Transfer Protokolü (HTTP)

Hiper Metin Transfer Protokolü (HTTP), en yaygın kullanılan İnternet uygulama protokollerinden biridir. HTTP, web tarayıcıları ve web sunucuları arasında iletişim kurmak, web sayfalarından bilgi okumak veya örneğin doldurulmuş formlar veya işaretli kutular gibi yanıtlar göndermek için kullanılır. İnternetin birçok kullanıcısı HTTP kullanan uygulamalara aşinadır, ancak bu uygulamalar web tarayıcılarının grafik arayüzlerinin arkasına çok iyi gizlendiğinden, herhangi birinin protokole aşina olması (hatta farkında olması) yaygın değildir. Dünya Çapında Ağ (WWW veya Web), nesnelerin evrensel olarak yorumlanabilmeleri için nasıl biçimlendirildiklerini, konumlarının kolayca bulunabilmeleri için nasıl ifade edilebileceğini ve İnternet içinde nasıl hareket ettirildiklerini açıklayan bir dizi standart tarafından basitleştirilmiştir (Farrel, 2004). Hiper Metin İşaretleme Dili (HTML), internet üzerinden en yaygın şekilde yayılan iletişim yoludur.

3.6. Köprü Metni İşaretleme Dili (HTML)

İşaretleme dili, belgeleri metinden kolayca ayırt edilebilen etiketler kullanarak metne gömülü biçimlendirme talimatlarıyla ortak bir karakter kümesinde (ASCII) sunma ve uygulamaya belgeyi nasıl göstereceğini veya yazdıracağını gösteren bir yöntemdir. C dilindeki dizelere aşina olan kodlayıcılar, "/" kaçış karakterinin ve "%" parametre biçimi karakterinin etiket biçimleri olduğunu anlayacaktır. Tüm kelime işlemciler, ürettikleri belgelerdeki biçimlendirmeyi temsil etmek için bir tür etiketleme kullanır. Girinti, yazı tipi ve metin rengi gibi şeyleri belirten etiketleme, genellikle tescilli kodlamaları kullanır,

böylece belge diske kaydedildiğinde, diğer uygulamaların yorumlaması zor veya imkansız olan ikili bir biçimde olur. Köprü metni, birden çok çapraz referansı veya anahtar tanımları olan ve bir kullanıcı tarafından elektronik olarak okunan belgelerde son derece popülerdir. Kullanıcı, fare veya benzeri bir cihaz kullanarak şemalardaki anahtar kelimeleri veya konumları işaret edebilir ve tıklayabilir böylelikle uygulama yeni bir belge yükleyecek veya metnin başka bir bölümüne atlayacaktır. Geleneksel olarak, bu tür köprü metni bağlantılarını oluşturan anahtar sözcükler veya tümcecikler, metinde özel renklendirme kullanılarak belirtilir ve belki de altı çizilir. Kaynak belgede, köprü metni bağlantıları, bağlanılacak sözcüğü veya tümceyi tanımlayan ve bağlanılacak konumu (dosya adı veya etiket) sağlayan özel etiketler aracılığıyla desteklenir. Köprü Metni Biçimlendirme Dili (HTML), İnternet'te yaygın olarak kullanılan, ortak ve iyi anlaşılmış anlamlara sahip standart bir etiketler kümesidir. Standart bir etiket kümesinin avantajı, herhangi bir uygulamanın dosyayı okuyabilmesi ve biçimlendirme talimatlarından aynı anlamı alabilmesidir. HTML bir adım daha ileri gider ve etiketler için kullanılabilen karakter kümesini yalnızca yazdırılabilir karakterlerle sınırlar, böylece HTML belgeleri tamamen aptal metin düzenleyicileri tarafından bile okunabilir. Bir HTML belgesinin gösterildiği veya yazdırıldığı zamanki tam düzeni, belgeyi okuyan uygulamanın sorumluluğundadır. Belirli etiketler için desteği sınırlamayı ve diğerlerini aynı çıktı stiliyle eşleştirmeyi seçebilir. Etiketlerin standart anlamları, uygulamanın tutarlı ve net çıktılar üretmesini kolaylaştırır (Michael, 2003).

3.6.1. HTML'nin Doğuşu ve Gelişmesi

HTML için ilham kaynağı, 1986'dan beri elektronik belgelerin yapısını kodlamak için kullanılan Standartlaştırılmış Genel İşaretleme Dili'nde (SGML) bulunabilir. Tim Berners-Lee "Conseil Europ'een pour la Recherche Nucl'eaire" (CERN) de köprüleri de dahil edebilen daha kolay bir HTML şeması geliştirmek için çalışmaya öncülük etti. HTML belgelerini İnternet üzerinden göndermek için gereken HTTP protokolü eş zamanlı olarak geliştirildi. CERN, işaretleme dilini ve protokolü 1991'de başlattı ve böylece World Wide Web'i yarattı. HTML 1.0, salt metin tarayıcılarda yalnızca metin tabanlı belgeleri oluşturmak için tasarlanmıştır. Desteklenen öğeler şunları içerir: sekme başlığı, paragraf (p), başlıklar (h1'den h6'ya), tanım listeleri, sırasız listeler ve en önemlisi belgeler arasında bağlantı kurmak için bağlantı (a). <, > ve & karakterleri de dilin alfabesinin bir parçasıydı. World Wide Web Konsorsiyumu, web ile ilgili konularda rehberlik sağlamak için 1994 yılında kuruldu. Konsorsiyumun ilk başta koyduğu kurallar HTML etiketlerinin derleyiciler tarafından yavaş yorumlanması sebebiyle HTML 1.0 topluluk tarafından kabul görmedi. 1996 yılına gelindiğinde konsorsiyum, html, head, body, base, link, meta, sıralı listeler, satır içi etiketler, form öğeleri, sıralı listeler, satır içi öğeler (b, kod, em, samp, strong, u), blok öğeler (blockquote ve pre), aralar (br ve hr) karakter varlıkları (© vb.) ve özel karakterleri (€, © vb.) ortaya çıkardı.

Bununla birlikte derleyicilerin daha hızlı yorumlama yapabilmesi için yeni bir algoritmayla beraber HTML 2.0 yayınladı. Böylelikle toplulukta referans kaynak olma özelliğini kazandı. Tüm

bunların yanında belki de en büyük gelişme, grafik dosyalarının görüntü ögesinin yardımıyla Mosaic ve Cello gibi tarayıcılar tarafından oluşturulabilmesiydi. Bu sürüm, statik web sayfalarının gerektirdiği işlevlerin çoğuna sahipti. HTML 2.0'ın gecikmiş (gecikmeli) onayı aslında "tarayıcı savaşlarının" en şiddetli döneminin (1993-1996) sonundaydı. Yeni gelen Microsoft Internet Explorer ve baskın tarayıcı olarak Mosaic'in yerini alan Netscape Navigator (daha sonra Communicator) arasındaki rekabet, hem kaos hem de bir dizi yenilik yarattı. Bunun neden olduğu çıkmaza rağmen, web geliştiricileri yine de genellikle kodlarını seçtikleri bir tarayıcı için "optimize etmeyi" tercih ettiler. Bu nedenle, HTML belgeleri yalnızca bir tarayıcı tarafından desteklenen öğeler, nitelikler veya değerler içeriyordu. Geliştiriciler tarafından diğer bir tarayıcı için optimize edilmemiş HTML dökümanları ne yazık ki istenildiği gibi gözüküyor ve işlevlerini yerine getiremiyordu.

Bir sonraki resmi W3C önerisi olan HTML 3.2, Ocak 1997'de geldi (HTML 3.0, W3C'ye sunulan ancak onaylanmayan bir teklifti). Bu sürümde div elementi ile birlikte stil ve komut dosyaları ekleme ve/veya çalışma özellikleri eklendi.

HTML 4.0, Aralık 1997'de W3C tarafından başlatıldı. HTML 3.2'den farklı olarak gelen en önemli özelliği JavaScript kodunu çağırabilen, obje olaylarını tetikleyebilen niteliklerin gelmesiydi. Bu nitelikleri dört kategoride sınıflandırabiliriz:

- Gövde ve Çerçeve Kümesi Elemanları İçin Pencere Olayları; onload ve onunload aksiyonları
- Form Elemanları ve Form Elemanlarının Çocukları İçin; onchange, onsubmit, onreset ve onselect aksiyonları
- Blok ve Satır İçi Elemanları İçin Klavye Olayları; onkeydown, onkeypress ve onkeyup
- Blok ve Satır İçi Elemanları İçin Fare Olayları; onclick, ondblclick, onmousedown, onmousemove, onmouseover, onmouseout ve onmouseup

3.6.2. HTML'in Son Versiyonu

Köprü Metni Biçimlendirme Dili revizyon 5 (HTML5), World Wide Web içeriklerinin yapısı ve sunumu için bir biçimlendirme dilidir. HTML5, biçimlendirme, Yeni API'ler, XHTML ve hata işlemede geleneksel HTML ve XHTML tarzı sözdizimini ve diğer yeni özellikleri destekler.

Şu anda HTML5 spesifikasyonundan üç kuruluş sorumludur:

- Web Hypertext Application Technology Working Group (WHATWG)
- World Wide Web Consortium (W3C)
- Internet Engineering Task Force (IETF)

HTML5, ortak uygulamaları düzenleyerek ve çeşitli tarayıcılardan uygulamaları sorunsuz şekilde çalıştırarak web geliştirme kaosa düzen getirme çalışmasıdır. HTML5 spesifikasyonlarının bir parçası olarak 100'den fazla özellik ve aksiyona sahiptir. HTML5, yeni nesil web uygulamaları ve işlevselliğin daha iyi biçimlendirme (HTML), daha iyi stil (CSS) ve daha iyi etkileşim (JavaScript) ile nasıl

geliştirileceğini kuralları ile birlikte anlatmaya yarar ve en basit haliyle böyle açıklanabilir. HTML5'in yeni özellikleri şunları içerir:

3.7. Standard Generalized Markup Language (SGML)

Standard Generalized Markup Language(Standart Genelleştirilmiş İşaretleme Dili) doküman işaretleme dillerinin tanımlanması için belirlenen standartlardır. SGML, 1960'larda geliştirilen IBM'in Genelleştirilmiş İşaretleme Dili'nin (GML) soyundan gelmektedir. Bilişim dünyasında yaygın olarak kullanılan HTML ve XML işaretleme dilleri SGML baz alınarak tanımlanmış olmakla beraber, SGML kurallarına tamamen uymadıkları için bir SGML uygulaması sayılmazlar. Tipik bir SGML örneği Şekil 3.2'de gösterilmiştir.

```
<LIST TYPE="Alisveris">  
  <ITEM>Makarna</ITEM>  
  <ITEM>Peynir</ITEM>  
</LIST>
```

Şekil 3.2. SGML örneği.

3.8. Extensible Markup Language (XML)

XML, SGML' nin bir alt kümesi olup SGML'ye ek olarak mevcut karmaşıklığın en aza indirilmesini amaçlar. XML, SGML ile tam uyumlu olacak şekilde tasarlanmıştır; bu, XML'in sözdizimi kurallarını izleyen herhangi bir belgenin, aynı zamanda SGML'nin sözdizimi kurallarını da takip ettiği ve bu nedenle mevcut SGML araçları tarafından okunabileceği anlamına gelir. Fakat bu ilişki iki yönlü çalışmaz. Yani bir XML belgesi SGML araçları tarafından okunabilirken bir SGML belgesi her zaman XML araçları tarafından okunamaz. Bununla birlikte, XML'in gerçekten bir "dil" olmadığını, ancak XML kriterlerini karşılayan diller oluşturmak için bir standart olduğunu anlamak önemlidir. Başka bir deyişle, XML, kendi dillerinizi oluşturmak için kullandığınız bir sözdizimini tanımlar. Örneğin:

```
<name>  
  <first>Mehmet</first>  
  <second>Alper</second>  
  <last>ŞAHİN</last>  
</name>
```

Şekil 3.3. XML örneği.

Bu basit örnekten bile, SGML ve XML gibi biçimlendirme dillerinin neden "kendi kendini tanımlayan" olarak adlandırıldığını görebilirsiniz. Verileri kontrol ederek, bunun bir <isim> hakkında bilgi olduğunu söyleyebilirsiniz ve <first>, <second> ve <last> adında verileri betimlediğini görebilirsiniz.

3.8.1. XML'in Avantajları

- Aynı anda insan ve makine tarafından okunabilen bir formattır.
- Unicode'u destekler ve herhangi bir yazılı insan dilindeki hemen hemen her türlü bilginin iletilmesine izin verir.
- En yaygın veri yapılarına uygundur: kayıtlar, listeler ve ağaçlar.
- Kendi kendini belgeleyen formatı, yapı ve alan adlarının yanı sıra belirli değerleri tanımlar.
- Esnek sözdizimi sayesinde veri birleştirme işlemlerini son derece basit, verimli ve tutarlı şekilde gerçekleştirir.
- XML, hem çevrimiçi hem de çevrimdışı olarak belge depolama ve işleme için kullanılır.
- Uluslararası standartlara bağlıdır.
- Etkin birim testi, güvenlik duvarları, kabul testi, sözleşme belirtimi ve yazılım yapımını kolaylaştıran XSD ve Schematron gibi şema dillerini kullanarak doğrulamayı onaylar.
- Hiyerarşik yapı, çoğu belge türü için uygundur.
- Diğer tescilli belge formatlarından daha az sınırlı olan düz metin dosyaları olarak kullanılabilir.
- Platform bağımsızdır ve bu nedenle dize tabanlı yapısı nedeniyle teknolojideki değişikliklere nispeten dirençlidir.
- Belge Türü Tanımı veya Şema'daki değişikliklere rağmen ileriye ve geriye dönük uyumlulukların bakımı nispeten kolaydır.
- Selef SGML, 1986'dan beri kullanılmaktadır, bu nedenle kapsamlı deneyim ve örnek yazılım kullanımları mevcuttur (XML, t.y.).

3.8.2. XML'in Dezavantajları

- XML sözdizimi, benzer teknolojilerin (JSON vb.) ikili temsillerine göre daha fazla yer kaplar.
- Yedeklilik, daha yüksek depolama, iletim ve işletim maliyetleri yoluyla uygulama verimliliğini etkileyebilir.
- XML sözdizimi, diğer alternatif 'metin tabanlı' veri iletim biçimlerine göre çok ayrıntılıdır.
- Veri türü desteği yoktur. Yani XML, belirli bir tamsayı, dize, mantıksal (boole), tarih vb. notasyon desteği sağlamaz.

- Veri sunum için kullandığı hiyerarşik model yapısı, ilişkisel modele veya nesne yönelimli modele kıyasla sınırlıdır.
- Örtüşen (hiyerarşik olmayan) düğüm ilişkilerini ifade etmek ekstra çaba gerektirir.
- XML ad alanlarının kullanımı sorunludur ve ad alanı desteğinin bir XML ayrıştırıcısında doğru şekilde uygulanması zor olabilir.
- XML yaygın olarak "kendi kendini belgeleyen" olarak tanımlanır, ancak bu açıklama kritik belirsizlikleri göz ardı eder (XML, T.y.).

3.9. XHTML

Yapısal olarak, XML, bir programlama dili gibi, mükemmel dilbilgisi(yazım kuralları) gerektirir. Tarayıcılar, genellikle yazılımcının aklını okumaya çalışmak gibi bir geleneğe sahipti ve bu yüzden kusurlu HTML sayfaları oluşturmaktaydılar. Örneğin kimi tarayıcı eksik noktalı virgüle rağmen son kullanıcıya doğru HTML çıktısını gösterirken başka bir tarayıcı alakasız bir HTML çıktısı göstermekteydi. Aslında bu durum XML'in avantajı olarak kabul ettiğimiz esnek olmasından meydana gelmekteydi.

XHTML 1.0 ile birlikte tarayıcıların yazılımcıların aklını okumaya çalışması yerine eksik olan noktlaama işaretleri ve/veya yazım hataları neticesinde son kullanıcıya bir HTML çıktısı vermek yerine bu talebi reddetme özelliği getirildi.

XML büyük/küçük harf duyarlı olduğundan ve XHTML bir XML uygulaması olduğundan, tüm XHTML tanımlayıcılarının tümü küçük harflerle görünmelidir. Bu, HTML kodu yazımında kullanılan tamamının büyük harf olması geleneğinin tersidir. Diğer bir kural ise ve ampersand (&), küçüktür (<) ve büyüktür (>) işaretlerinin karakter verisi olarak kullanılamamasıdır. Bunlar işaretleme karakterleri olarak tasarlanmadıklarında, & < ve > yazım kuralları ile karakter olarak kullanılabilirler ancak. Yine benzer şekilde kesme işareti/tek tırnak/apostrophe (') ve çift tırnak (") işaretleri de ' şeklinde kodlanmıştır. Bunlar gibi daha çok çeşitli işaretlerin kullanılabilmesi için ASCII karşılıkları kullanılmalıdır. En faydalı olanlardan biri özel sembolüdür. Boşluk yani space karakterine denk gelir ve içi boş olan bir hücre ile boş olmayan bir hücreyi aynı şekilde çizilmesine yarar (Michael, 2003).

Bir HTML/XHTML ögesinin büyük küçük semboller (< >) içine alınmış açılış etiketi ögesi vardır ve her açılış etiketi mutlaka Şekil 3.4'te gösterildiği şekilde bir kapanış etiketi (</>) ile kapatılmalıdır.

<p>Profesör Doktor Murat YAKAR.</p>

Şekil 3.4. HTML/XHTML açılış ve kapanış etiketi.

Bu kuralın tek bir istisnası vardır o da boş karakter bırakarak bir alt satıra inmeye yarayan
 şeklinde kodlanmış karakterdir. Bu öğeyi
</br> şeklinde bir açılış ve bir kapanış etiketi yerine tek bir
 etiketi ile ifade ederiz. Bunlar, XHTML için temel sözdizimleridir. Gördüğünüz gibi XHTML, HTML ve XML yapılarının bir kombinasyonudur. XHTML, HTML'nin halefi olsa da, tamamen yeni bir dil değildir. XHTML, yalnızca yeni modern internet ve yeni teknolojiler için bir kurallar dizisi tanımlar.

3.10. Veritabanı Teknolojileri

Bilgi çağında yaşıyoruz. Ürettiğimiz veya öğrendiğimiz tüm bilgileri depolamak zorundayız. Bilgisayar bilimcileri, bilgileri depolamak için bir veritabanı fikri buldular. Ücretsiz Çevrimiçi Bilgi İşlem Sözlüğü olan FOLDOC (<http://foldoc.org/>) bir veritabanını "genellikle verileri saklamak, güncellemek ve sorgulamak için yazılımla ilgili olan bir veya daha fazla olan büyük yapılandırılmış kalıcı veri kümesidir. Basit bir veritabanı, her biri, her alanın belirli bir sabit genişlikte olduğu aynı alan kümesini içeren birçok kaydı içeren tek bir dosya olabilir" şeklinde tanımlamıştır.

Modern veritabanları, 1960'larda diğer şirketlerin yanı sıra IBM'de yapılan araştırmaların sonucunda ortaya çıktı. Araştırma, daha önce çok fazla insan gücü gerektiren veri depolama ve dizin oluşturma görevlerinin otomatikleştirilmesi etrafında odaklandı. Böylelikle gerekli olan insan kaynağı azalarak, veri işlem gücü, veri depolama ve veri indeksleme bilgisayar kullanımı ile çok daha ucuz hale geldi. Veritabanı alanındaki öncülerden biri, veritabanı teknolojisindeki öncü çalışmaları nedeniyle 1973'te Turing Ödülü'nü alan Charles W. Bachman'dı.

1970 yılında, Ted Codd adlı bir IBM araştırmacısı, ilişkisel veritabanları üzerine ilk makaleyi yayınladı. IBM, veritabanı araştırmasında lider olmasına rağmen, Honeywell Information Systems, Inc., 1976'da IBM bilgi sistemiyle aynı ilkelere dayanan ticari bir ürün piyasaya sürdü, ancak bu, IBM'in çalışmasından ayrı olarak tasarlanmış ve uygulanmıştı. 1980'lerde, SQL standardı üzerine inşa edilen ilk veritabanı sistemleri Oracle Şirketinin geliştirdiği Oracle Versiyon 2 ve IBM'in geliştirdiği SQL/DS oldu.

1990'ların başında, daha az sayıda şirketin giderek daha karmaşık ürünleri daha yüksek fiyatlarla sunmasıyla bir endüstri sarsıntısı başlar. Bu dönemdeki geliştirmelerin çoğu, PowerBuilder (Sybase), Oracle Developer, VB (Microsoft), vb. gibi uygulama geliştirme için istemci araçlarına odaklanır. İstemci-sunucu bilgi işlem modeli, gelecekteki iş kararları için norm haline gelir. Bu bilgi işlem modelinin ilk uygulamaları Excel/Access (MS) ve ODBC gibi kişisel üretkenlik araçlarıdır. Bu aynı zamanda Nesne Veritabanı Yönetim Sistemleri (ODBMS) prototiplerinin başlangıcını da işaret ediyordu.

1990'ların sonlarında, internet üzerine çalışan şirketlere yapılan büyük yatırım, özellikle veritabanı konektörleri için uygulama geliştirme pazarındaki patlamayı körükledi. Active Server Pages, Front Page, Java Servlet, JDBC, Enterprise Java Beans, ColdFusion, Dream Weaver, Oracle Developer 2000 vb. bu tür uygulamalara örnektir. Bu uygulamalar ücretli olup teknolojinin gelişim hızındaki

negatif etkisi ortaya çıktıkça alternatif uygulamalar gönüllü kişi/kişiler ve/veya şirketler tarafından geliştirilip ücretsiz olarak pazara sürülmeye başladı. Bu uygulamalara GEE, CGI, Apache, MySQL vb. örnek olarak verilebilir. 2000'lerin başında, veritabanı uygulamalarının büyümesi devam etti. Cep bilgisayarlarının ve el terminallerinin kullanımı ile, POS işlemleri, satıcıların konsolidasyonu vb. işlemlerle daha etkileşimli uygulamalar ortaya çıktı. Bu büyük pazara üç ana şirket hakimdi; IBM (Informix satın alımı sonrası), Microsoft ve Oracle.

Günümüzde verinin önemi insanlık için suyun önemi ile eşdeğer hale gelmiştir. Verilerin boyutları petabyte seviyelerine çıkmış bu verilerin saklanması, işlenmesi, ilişkilendirilmesi ve akıllı hale getirilerek geleceğe yönelik yüksek yüzdeli öngörülerin gerçekleşmesi için Veri Ambarı, Veri Madenciliği vb. iş ve işlemleri gerçekleştirmek için çok daha komplike sistemler ortaya çıkmıştır. Bir süre sonra bu sistemlerinde yetersiz gelmeye başlaması neticesinde veri de bir ayrışma gidilmiştir. Bu ayrımın en güncel örneği Spatial Data yani Konumsal/Mekansal/Uzamsal Veri'dir.

3.10.1. Konumsal/Mekansal/Uzamsal Veritabanları

Son yıllarda, geleneksel veritabanları coğrafi verileri içerecek şekilde genişletildi. Coğrafi Veritabanı Yönetimi (SDBM) sistemleri, alanla ilgili verileri etkin bir şekilde manipüle etmek için modeller sağlamayı amaçlamaktadır. Bu tür bir manipülasyon, örneğin bilgisayar destekli tasarım (CAD), çok büyük ölçekli entegrasyon, robotik, navigasyon sistemleri ve görüntü işleme gibi büyük uzamsal veri kümelerine dayanan her türlü uygulama için yararlıdır. Mekansal veri uygulamaları (özellikle jeo-uzamsal uygulamalar), aşağıdaki nedenlerle geleneksel veri uygulamalarından farklıdır (Voisard ve David, 2002).

- Uzamsal bilgi, uzamsal ve uzamsal olmayan verilerle ilgilenir; bu, uzamsal veri türlerinin tanımının kendilerine uygulanabilir işlemler altında tamamlanması gerektiği anlamına gelir;
- Birçok uzamsal veri, doğal olarak tutarsız veya belirsizdir, bu da çelişkili verilere yol açabilir (örneğin, bir gölün veya kirlilik alanının tam sınırı genellikle belirsizdir);
- Topolojik ve diğer coğrafi ilişkiler çok önemlidir ve genellikle dolaylı olarak temsil edilir;
- Veriler, nesne toplama kavramıyla yüksek düzeyde yapılandırılmıştır;
- Kullanıcı tanımlı süreçler, genişletilebilir bir temel modele ihtiyaç duyar;
- İşlevler hem düşük düzeyde bir soyutlamada (örneğin, noktalar, çizgiler, çoklu çizgiler) hem de yüksek düzeyde bir soyutlamada (örneğin, haritalar, tematik katmanlar, konfigürasyonlar) meydana gelir. (Voisard ve David, 2002).

Bu nedenler, bilim insanlarını coğrafi uygulamalar için uzmanlaşmış veri tabanları geliştirmeye itmektedir. Coğrafi veritabanları çoğunlukla, belirsizlik, belirsizlik, eksik ve tutarsızlık tutarsızlığı olarak sınıflandırılabilen farklı türdeki veri kusurlarıyla ilgilenir. Tutarsızlık, belirsizlik ve eksiklik genellikle veri toplamadaki sorunlardan kaynaklanan farklı veri yanlışlıkları olarak görülürken,

tutarsızlık tutarsız verilerin varlığından kaynaklanan veri kusuru türüdür. Bütünlük kısıtlamaları, coğrafi veritabanlarında tutarlılığı zorunlu kılar. Bir veri tabanını güncellerken, verilerin anlambilimi ve kalitesi korunacak şekilde bütünlük sınırları dikkate alınmalıdır. Coğrafi veritabanlarında tutarlılığın tedavisi, esas olarak bütünlük sınırlarını belirlemeye ve bunları veri manipülasyonu için zorlamaya odaklanmıştır. Farklı coğrafi veri işleme süreçlerindeki tutarsızlığı tespit etmek ve açıklamak için resmi modellerle ilgili bu sorunlar için çözümler, örneğin çoklu temsiller ve veri entegrasyonu olsa da, çabaların çoğu hala kavramsal düzeyde veya izole uygulamalardadır.

3.11. Python Programlama Dili

Bilgisayarlar üzerinde çok çalışırsanız, sonunda otomatikleştirmek istediğiniz bir görev olduğunu görürsünüz. Örneğin, çok sayıda metin dosyası üzerinde bir arama ve değiştirme işlemi gerçekleştirmek veya bir grup fotoğraf dosyasını karmaşık bir şekilde yeniden adlandırıp yeniden düzenlemek isteyebilirsiniz. Belki küçük bir özel veritabanı, özel bir GUI uygulaması veya basit bir oyun yazmak istersiniz.

Profesyonel bir yazılım geliştiriciyseniz, birkaç C/C++/Java kitaplığıyla çalışmanız gerekebilir, ancak olağan yazma/derleme/test/yeniden derleme döngüsünün çok yavaş olduğunu görebilirsiniz. Belki de böyle bir kitaplık için bir test takımı yazıyorsunuz ve test kodunu yazmayı sıkıcı bir iş olarak görüyorsunuz. Veya bir uzantı dili kullanabilecek bir program yazdınız ve uygulamanız için tamamen yeni bir dil tasarlamak ve uygulamak istemiyorsunuz. Python tam size göre bir dildir.

Bu görevlerin bazıları için bir Unix kabuk komut dosyası veya Windows toplu iş dosyaları yazabilirsiniz, ancak kabuk komut dosyaları, GUI uygulamaları veya oyunlar için çok uygun olmadığından, dosyalar arasında hareket etme ve metin verilerini değiştirme konusunda en iyisidir. Bir C/C++/Java programı yazabilirsiniz, ancak ilk taslağı elde etmek bile çok fazla geliştirme süresi alabilir. Python'un kullanımı daha basittir, Windows, macOS ve Unix işletim sistemlerinde bulunur ve işi daha hızlı yapmanıza yardımcı olur.

Python'un kullanımı basittir, ancak gerçek bir programlama dilidir ve büyük programlar için kabuk komut dosyaları veya toplu iş dosyalarının sunabileceğinden çok daha fazla yapı ve destek sunar. Öte yandan Python, C'den çok daha fazla hata kontrolü sunar ve çok yüksek seviyeli bir dil olduğundan, esnek diziler ve sözlükler gibi yerleşik yüksek seviyeli veri türlerine sahiptir. Daha genel veri türleri nedeniyle Python, Awk ve hatta Perl'den çok daha büyük bir sorun alanına uygulanabilir, ancak Python'da birçok şey en azından bu dillerde olduğu kadar kolaydır.

Python, programınızı diğer Python programlarında yeniden kullanılabilecek modüllere ayırmanıza olanak tanır. Programlarınızın temeli olarak veya Python'da programlamayı öğrenmeye başlamak için örnek olarak kullanılabileceğiniz geniş bir standart modül koleksiyonuyla birlikte gelir. Bu modüllerden bazıları, dosya G/Ç, sistem çağrıları, yuvalar ve hatta Tk gibi grafiksel kullanıcı arabirimi araç takımlarına arabirimler gibi şeyler sağlar.

Python, derleme ve bağlama gerekmediğinden program geliştirme sırasında size önemli ölçüde zaman kazandırabilecek yorumlanmış bir dildir. Tercüman etkileşimli olarak kullanılabilir, bu da dilin özelliklerini denemeyi, kullanılıp atılan programlar yazmayı veya aşağıdan yukarıya program geliştirme sırasında işlevleri test etmeyi kolaylaştırır. Aynı zamanda kullanışlı bir masa hesap makinesidir.

Python, programların kompakt ve okunabilir bir şekilde yazılmasını sağlar. Python'da yazılan programlar, çeşitli nedenlerle tipik olarak eşdeğer C, C++ veya Java programlarından çok daha kısadır:

- Üst düzey veri türleri, karmaşık işlemleri tek bir ifadede ifade etmenize olanak tanır;
- Deyim gruplandırması, başlangıç ve bitiş parantezleri yerine girinti ile yapılır;
- Değişken veya bağımsız değişken bildirimleri gerekli değildir.

Python genişletilebilirdir : C'de nasıl programlanacağını biliyorsanız, kritik işlemleri maksimum hızda gerçekleştirmek veya Python programlarını yalnızca mevcut olabilecek kitaplıklara bağlamak için yorumlayıcıya yeni bir yerleşik işlev veya modül eklemek kolaydır. ikili biçimde (satıcıya özel grafik kitaplığı gibi). Gerçekten bağımlı olduğunuzda, Python yorumlayıcısını C ile yazılmış bir uygulamaya bağlayabilir ve bu uygulama için bir uzantı veya komut dili olarak kullanabilirsiniz.

Bu arada, dil adını BBC'nin "Monty Python's Flying Circus" programından alıyor ve sürüngelele hiçbir ilgisi yok. Belgelerde Monty Python skeçlerine atıfta bulunmak sadece izin verilmez, teşvik edilir!

3.12. Javascript

JavaScript, web sayfalarının yeteneğini artırmak için en yaygın olarak kullanılandır. Web geliştiricisinin kullanıcı eylemlerine bir tür dinamik yanıt oluşturmalarını sağlar. Bu yanıtlar, değişen renkleri, hareketli görüntüleri, hesaplamaları gerçekleştirmeyi ve bileşenlerin görünmesini ve kaybolmasını içerir. JavaScript dili, World Wide Web ile birlikte gelişti ve kendisini popüler ve önemli bir teknoloji olarak kabul ettirdi. Daha fazla web işlevselliği için artan kullanıcı talebine yanıt olarak gelişmeye devam edecektir.

1995'ten önce HTML belgeleri genellikle paragraf biçiminde düz metindi. Kullanıcıların web sayfalarını göstermede daha fazla işlevsellik talebine yanıt olarak, Nisan 1995'te Netscape Sürüm 1. 1, tablolar ve daha birçok öğe ve nitelik ekledi. Ancak kullanıcılar, kişisel bilgisayarlarında kendileriyle dinamik yollarla etkileşime giren programlara alışmışlardı. web sayfaları bunu ancak en minimal biçimde yapabiliirdi. Sorunun özünde HTML vardı. Ticari bir işaret biçimlendirme paradigmasına dayalı olarak, bir programlama dili olarak tasarlanmamıştır. HTML "etiketleri", metin ve grafik resimlerin nereye yerleştirileceğini, ne kadar büyük olması gerektiğini ve bir belgedeki diğer öğelerle ilişkilerinin ne olduğunu belirtir. HTML, günün karmaşık uygulamalarını oluşturmak için kullanılan programlama dillerinin özelliklerine sahip değildi. Bir tür gömülü programlama dilinin gerektirdiği daha dinamik etkileşimli web sayfalarına olan talebi karşılanması gerekiyordu (Bidgoli, 2003).

1995'in başında Netscape, böyle yeni bir dilin tasarımı ve uygulanmasından sorumlu olmak için Brendan Eich'i görevlendirdi. Eich, gevşek yazılmış bir betik dilinin çevre ve izleyici için uygun olduğuna karar verdi. Dilin hedefi statik web sayfalarına hayat vermek olduğu için "LiveScript" olarak adlandırıldı. Programlama dilinin genel kabul görmesi nedeniyle, Live Script'in C++ sözdizimini kullanması şaşırtıcı değildi. Pazarlama düşüncelerine göre, dil kısa süre sonra JavaScript olarak yeniden adlandırıldı. 4 Aralık 1995'te Netscape ve Sun, Netscape Navigator 2.0B3 sürümündeki yeni dili ortaklaşa duyurdular ve onu hem HTML hem de Java için bir tamamlayıcı haline getirdiler. Java ve Javascript isimleri arasındaki benzerlik, iki dil arasında güçlü bir bağlantı olduğu algısı nedeniyle hala kafa karıştırıyor. Ancak böyle bir bağlantı kesinlikle yok. İki dil tarafından paylaşılan ortak C++ sözdizimi, amaç, temel bilgiler ve uygulamadaki önemli farklılıkları gizler (Bidgoli, 2003).

3.12.1. Javascript' in Yetenekleri

JavaScript, HTML belgelerine gömülü bir betik dili olarak, web içeriğini dinamik olarak kontrol etmenin bir yoludur. JavaScript, web tarayıcısının bir nesnesinin veya ögesinin hemen hemen her özelliğini değiştirmek için kullanılabilir. Örneğin JavaScript, fareyi hareket ettirmek veya bir tuşa basmak gibi kullanıcı olaylarına yanıt olarak HTML öğelerinin rengini, boyutunu ve konumunu değiştirmek için kullanılabilir. Bu öğeler resimler, tablolar, giriş kutuları ve metin blokları içerebilir. Öğeleri görünür veya görünmez yapmak için kullanılabilir. Tarayıcının başka bir web sayfasına gitmesini veya geçmiş listesinde ileri veya geri hareket etmesini sağlayabilir. JavaScript'in tanımladığı içerikle yeni tarayıcı pencereleri bile oluşturabilir. Yani JavaScript, tarayıcı tarafından yürütülecek HTML'yi dinamik olarak oluşturabilir. Javascript ayrıca sunucu ile iletişim kurabilir. JavaScript dinamik olarak bilgi oluşturabilir ve web sunucusunda bulunan programlara bilgi aktarabilir. Bunu, formlar ve sorgu dizeleri kullanarak yapar. Formlar içeren web sayfaları, form öğelerinin içeriğinin sunucudaki bir programa geçirilmesine neden olmak için JavaScript kullanabilir. Öte yandan, Javascript kullanarak bir web sayfası, formlarını göndermeden sunucuyla iletişim kurabilir. Bu tür iletişime AJAX (Asenkron Javascript ve XML) denir (Bidgoli, 2003).

3.13. Asenkron Javascript ve XML (AJAX)

Ajax, etkileşimli web uygulamaları yapmak için kullanılan bir web geliştirme yöntemidir. Amaç, sunucuyla küçük miktarlarda veri alışverişi yaparak web sayfalarının daha duyarlı görünmesini sağlamak, bu arada sayfa kullanıcı tarafından deneyimlenebilirken, kullanıcı her değişiklik istediğinde web sayfasının tamamen yeniden yüklenmesine gerek kalmaz. Bu, web sayfasının etkileşimini, hızını, işlevselliğini ve kullanılabilirliğini geliştirmeyi amaçlamaktadır (AJAX, t.y.). Geliştiriciler, Ajax kullanarak, web'in geleneksel geri gönderme modelinin dışına çıkan, kullanıcılara gelişmiş bir kullanım

arayüzü sağlayan ve geliştiricilerin çok daha kullanıcı dostu uygulamalar geliştirmesine izin veren uygulamalar oluşturabilir.



Şekil 3.5. AJAX nasıl çalışır.

Ajax, bir web sunucusuna asenkron aramalar yapma olanağı sağlar. Bu, istemci tarayıcısının, kullanıcının bir kez daha harekete geçmesine izin vermeden önce tüm verilerin gelmesini beklemekten kaçınmasına olanak tanır. Tam bir geri gönderme yapılmaması ve tüm form verilerinin sunucuya gönderilmesiyle ağ kullanımı en aza indirilir ve işlemler daha hızlı gerçekleşir. Şekil 3.5, bir web sayfasının gönderilmesini ve sunucudan geri dönüş yanıtını gösterir. Veri aktarımı için kısıtlı borulara sahip sitelerde ve konumlarda bu, ağ performansını büyük ölçüde artırabilir. AJAX uygulamaları istemcide asenkron olduğundan, çok duyarlı oldukları algılanır. Bugün en büyük teknoloji şirketlerinden bazıları, daha kullanıcı dostu, sorunsuz çalışan uygulamalar geliştirmek için AJAX sistemini kullanıyor, bu şirketler Google, Microsoft, Developer Express, Telerik vb (AJAX, t.y.).

4. BULGULAR VE TARTIŞMA

4.1. Açık Kaynak Kodlu Web CBS Uygulaması

Bu projenin amacı, Python, GeoServer, PostGRE SQL Server ve PostGIS kullanarak geçmiş mülklerin verilerini yayınlamak için web tabanlı coğrafi bilgi uygulaması geliştirmektir. Web uygulaması iki bağımsız bölümden oluşmaktadır, ilki internet harita uygulaması, ikincisi ise ortak veri tabanı uygulamasıdır. GeoServer üzerinde harita uygulaması geliştirildi. İkinci kısım Python'da geliştirilmiştir, bu uygulama veritabanı olarak PostGRE SQL Server'ı kullanır. Harita uygulaması, nokta veya alanın coğrafi özelliklerini saklar ve veritabanı uygulaması, bir nokta veya arazi özelliği hakkındaki metin verilerini saklar. Bu iki veri türü ile mülkün tüm özellikleri saklanabilir ve kullanıcılara sunulabilir. Javascript ile temel hesaplamalar veya mesafe ve alan ölçümü gibi analizler yapılabilir. Projenin aşamaları, uygulama tasarımı, kodlama ve geliştirme şeklinde yazılabilir.

4.2. GeoServer

GeoServer, Java ile yazılmış, kullanıcıların coğrafi verileri paylaşmasına ve düzenlemesine izin veren açık kaynaklı bir yazılım sunucusudur. Birlikte çalışabilirlik için tasarlanmış olup, açık standartları kullanarak herhangi bir büyük uzamsal veri kaynağından veri yayınlar. Topluluk odaklı bir proje olan GeoServer, dünyanın dört bir yanından çeşitli bireyler ve kuruluşlar tarafından geliştirilmiş, test edilmiş ve desteklenmiştir. GeoServer, Open Geospatial Consortium (OGC) Web Özellik Hizmeti (WFS) ve Web Kapsama Hizmeti (WCS) standartlarının yanı sıra yüksek performans sertifikalı uyumlu Web Harita Hizmeti'nin (WMS) referans uygulamasıdır. GeoServer, Geospatial Web'in temel bir bileşenini oluşturur.

GeoServer, 2001 yılında New York merkezli kar amacı gütmeyen bir teknoloji inkübatörü olan The Open Planning Project (TOPP) tarafından başlatıldı. TOPP, açık demokrasiyi mümkün kılmak ve hükümeti daha şeffaf hale getirmeye yardımcı olmak için bir dizi araç oluşturuyordu. Bunlardan ilki, vatandaşların hükümete ve şehir planlamasına katılımını sağlayan bir araç takımının, mekansal verileri paylaşma yeteneği ile büyük ölçüde geliştirileceğinin kabul edilmesiyle ortaya çıkan GeoServer'dı.

GeoServer kurucuları, World Wide Web'e benzer bir Geospatial Web tasarladılar. World Wide Web ile metin arayabilir ve indirebilirsiniz. Geospatial Web ile uzamsal veriler aranabilir ve indirilebilir. Veri sağlayıcılar, verilerini doğrudan bu web'de yayınlayabilecek ve kullanıcılar, günümüzde var olan dolaylı ve hantal veri paylaşım yöntemlerinin aksine, verilere doğrudan erişebilecektir.

GeoServer ile ilgili olanlar , açık kaynaklı bir GIS Java araç takımı olan GeoTools projesini kurdular. GeoTools aracılığıyla şekil dosyaları, Oracle veritabanları ve çok daha fazlası için destek eklendi. GeoServer'ın kurulduğu sıralarda, OpenGIS Konsorsiyumu (şimdi Open Geospatial Consortium) Web Özellik Hizmeti standardı üzerinde çalışıyordu. Birlikte çalışabilir bir veri formatı

olan GML (Coğrafi İşaretleme Dili) kullanarak uzamsal verileri doğrudan web üzerinde kullanılabilir hale getirmek için bir protokol belirtir. Mekansal verilerden oluşturulan harita görüntülerini oluşturmak ve görüntülemek için bir protokol olan bir Web Harita Hizmeti de oluşturuldu.

Diğer projeler birbiriyle bağlantılı hale geldi. Refrations Research , GeoServer'ın ücretsiz bir veritabanına bağlanmasını sağlayan ücretsiz ve açık bir uzaysal veritabanı olan PostGIS'i yarattı. Ayrıca, MetaCarta başlangıçta açık kaynaklı tarayıcı tabanlı bir harita görüntüleme aracı olan OpenLayers'ı yarattı. Birlikte, bu araçların tümü GeoServer'ın işlevselliğini geliştirdi. GeoServer artık bir düzineden fazla uzamsal veri kaynağından veri okuyabilir ve birçok farklı formatta çıktı alabilir. Şimdi ikinci on yılında, GeoServer uzamsal verileri herkes için daha erişilebilir hale getirme misyonunu sürdürüyor.

4.3. PostGRE SQL Server

PostgreSQL, Berkeley Bilgisayar Bilimleri Departmanında California Üniversitesi'nde geliştirilen, POSTGRES, Sürüm 4.2'ye dayalı bir nesne-ilişkisel veritabanı yönetim sistemidir (ORDBMS). POSTGRES, yalnızca bazı ticari veritabanı sistemlerinde çok daha sonra kullanılabilir hale gelen birçok konseptte öncülük etti. PostgreSQL , bu orijinal Berkeley kodunun açık kaynaklı bir soyundan gelmektedir. SQL standardının büyük bir bölümünü destekler ve birçok modern özellik sunar:

- Karmaşık sorgular,
- Yabancı anahtarlar
- Tetikleyiciler
- Güncellenebilir görünüm
- İşlem bütünlüğü
- Çok sürümlü eşzamanlılık denetimi

Ayrıca PostgreSQL , kullanıcı tarafından, örneğin yeni eklemeler gibi birçok yolla genişletilebilir.

- Veri tipleri
- Fonksiyonlar
- Operatörler
- Toplu fonksiyonlar
- İndeks yöntemleri
- Prosedürel diller

Ve liberal lisans nedeniyle, PostgreSQL herkes tarafından özel, ticari veya akademik olsun herhangi bir amaç için ücretsiz olarak kullanılabilir, değiştirilebilir ve dağıtılabilir.

Artık PostgreSQL olarak bilinen nesne-ilişkisel veritabanı yönetim sistemi , Berkeley'deki California Üniversitesi'nde yazılan POSTGRES paketinden türetilmiştir . Arkasında yirmi yılı aşkın bir süredir geliştirme ile PostgreSQL , artık her yerde mevcut olan en gelişmiş açık kaynaklı veritabanıdır.

Profesör Michael Stonebraker liderliğindeki POSTGRES projesi, Savunma İleri Araştırma Projeleri Ajansı (DARPA), Ordu Araştırma Ofisi (ARO), Ulusal Bilim Vakfı (NSF) ve ESL, Inc. tarafından desteklendi . 1986. Sistem için ilk kavramlar [ston86] 'da sunuldu ve ilk veri modelinin tanımı [rowe87] 'de görüldü . O zamanki kural sisteminin tasarımı [ston87a] 'da açıklanmıştır . Depolama yöneticisinin mantığı ve mimarisi [ston87b] 'de detaylandırılmıştır .

POSTGRES , o zamandan beri birkaç büyük sürümden geçti. İlk “ demoware ” sistemi 1987'de faaliyete geçti ve 1988 ACM-SIGMOD Konferansı'nda gösterildi. [ston90a] 'da açıklanan Sürüm 1, Haziran 1989'da birkaç harici kullanıcıya sunuldu. İlk kural sisteminin ([ston89]) eleştirisine yanıt olarak, kural sistemi yeniden tasarlandı ([ston90b]) ve Sürüm 2 Haziran 1990'da yeni kural sistemiyle piyasaya sürüldü. Sürüm 3 1991'de ortaya çıktı ve birden çok depolama yöneticisi, geliştirilmiş bir sorgu yürütücüsü ve yeniden yazılmış bir kural sistemi için destek ekledi. Çoğunlukla, Postgres95'e kadar sonraki sürümler(aşağıya bakınız) taşınabilirlik ve güvenilirliğe odaklanmıştır.

POSTGRES , birçok farklı araştırma ve üretim uygulamasını gerçekleştirmek için kullanılmıştır. Bunlar şunları içerir: bir finansal veri analiz sistemi, bir jet motoru performans izleme paketi, bir asteroit izleme veritabanı, bir tıbbi bilgi veritabanı ve çeşitli coğrafi bilgi sistemleri. POSTGRES ayrıca çeşitli üniversitelerde bir eğitim aracı olarak kullanılmıştır. Son olarak, Illustra Information Technologies (daha sonra şimdi IBM'e ait olan Informix ile birleştirildi) kodu aldı ve ticarileştirdi. 1992'nin sonlarında, POSTGRES , Sequoia 2000 bilimsel hesaplama projesi için birincil veri yöneticisi oldu .

Dış kullanıcı topluluğunun boyutu 1993'te neredeyse iki katına çıktı. Prototip kodun ve desteğin bakımının veri tabanı araştırmasına ayrılması gereken çok fazla zaman aldığı giderek daha açık hale geldi. Bu destek yükünü azaltmak amacıyla Berkeley POSTGRES projesi resmen Sürüm 4.2 ile sona erdi.

1994 yılında Andrew Yu ve Jolly Chen, POSTGRES'e bir SQL dili yorumlayıcısı eklediler . Yeni bir ad altında, Postgres95 daha sonra orijinal POSTGRES Berkeley kodunun açık kaynaklı bir soyundan gelen dünyada kendi yolunu bulmak için web'e yayınlandı. Postgres95 kodu tamamen ANSI C idi ve boyut olarak %25 oranında kısaltıldı. Birçok dahili değişiklik, performansı ve sürdürülebilirliği iyileştirdi. Postgres95 sürüm 1.0.x, POSTGRES Sürüm 4.2'ye kıyasla Wisconsin Benchmark'ta yaklaşık %30-50 daha hızlı çalıştı . Hata düzeltmelerinin yanı sıra, başlıca geliştirmeler şunlardı:

- PostQUEL sorgu dili SQL ile değiştirildi (sunucuda uygulandı). (Arayüz kitaplığı libpq , PostQUEL'den sonra adlandırılmıştır.) Alt sorgular PostgreSQL'e kadar desteklenmezdi (aşağıya bakın), ancak Postgres95'te kullanıcı tanımlı SQL işlevleriyle taklit edilebilirler . Agregasyonları yeniden uygulandı. Sorgu maddesi desteği GROUP BY'de eklendi.
- Etkileşimli SQL sorguları için GNU Readline kullanan yeni bir program (psql) sağlandı . Bu, eski monitör programının yerini büyük ölçüde aldı.
- Yeni bir ön uç kitaplığı libpgtcl, desteklenen Tcl tabanlı istemciler. Örnek bir kabuk, Tcl programlarını Postgres95 sunucusuyla pgsqlsharayüzlemek için yeni Tcl komutları sağladı .

- Büyük nesne arayüzü elden geçirildi. Büyük nesnelerin ters çevrilmesi, büyük nesneleri depolamak için tek mekanizmaydı. (Tersine çevirme dosya sistemi kaldırıldı.)
- Örnek düzeyinde kural sistemi kaldırıldı. Kurallar, yeniden yazma kuralları olarak hâlâ mevcuttu.
- Kaynak koduyla birlikte normal SQL özelliklerini ve Postgres95'in özelliklerini tanıtan kısa bir eğitim verildi.
- Yapı için GNU make (BSD make yerine) kullanıldı. Ayrıca, Postgres95 yama uygulanmamış bir GCC ile derlenebilir (çiftlerin veri hizalaması düzeltildi).

1996 yılına gelindiğinde, “ Postgres95 ” adının zamana direnemeyeceği anlaşıldı . Orijinal POSTGRES ile SQL özelliğine sahip daha yeni sürümler arasındaki ilişkiyi yansıtmak için yeni bir ad olan PostgreSQL seçtik . Aynı zamanda, sayıları Berkeley POSTGRES projesi tarafından orijinal olarak başlatılan sıraya geri koyarak, sürüm numaralandırmasını 6.0'da başlayacak şekilde ayarladık. Birçok kişi , gelenek nedeniyle veya telaffuzu daha kolay olduğu için PostgreSQL'e “ Postgres ” (şimdi nadiren tüm büyük harflerle) olarak atıfta bulunmaya devam ediyor . Bu kullanım, bir takma ad veya takma ad olarak yaygın olarak kabul edilir. Postgres95'in geliştirilmesi sırasında vurgu , sunucu kodundaki mevcut sorunları belirlemek ve anlamaktı. PostgreSQL ile tüm alanlarda çalışmalar devam etse de vurgu, özelliklerin ve yeteneklerin artırılmasına kaymıştır.

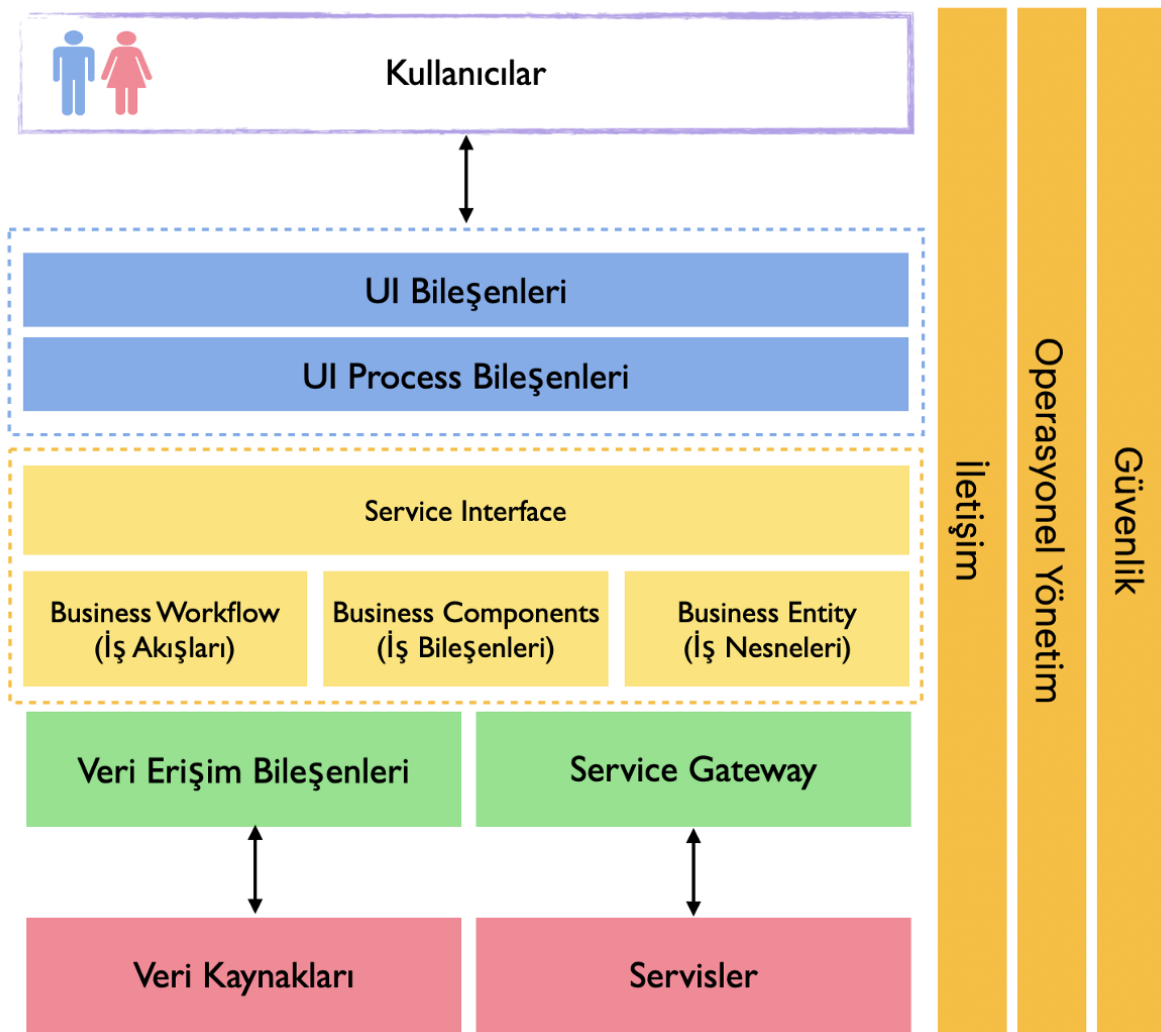
4.3.1. PostGIS

PostGIS, bir coğrafi veri tabanı teknolojisi araştırma projesi olarak Refrations Research Inc tarafından oluşturulan PostgreSQL ilişkisel veri tabanının uzamsal bir uzantısıdır. Refrations, Victoria, British Columbia, Kanada'da bulunan ve veri entegrasyonu ve özel yazılım geliştirme konusunda uzmanlaşmış bir CBS ve veritabanı danışmanlık şirkettir. PostGIS artık OSGeo Vakfı'nın bir projesidir ve dünyanın her yerinde, işlevselliğinden ve çok yönlülüğünden büyük fayda sağlayan birçok FOSS4G geliştiricisi ve kuruluşu tarafından geliştirilmekte ve finanse edilmektedir. PostGIS proje geliştirme grubu, OGC ve SQL/MM uzamsal standartlar, gelişmiş topolojik yapılar (kapsamlar, yüzeyler, ağlar), masaüstü kullanıcı arabirimi araçları için veri kaynağı alanlarında bir dizi önemli GIS işlevselliğini daha iyi desteklemek için PostGIS'i desteklemeyi ve geliştirmeyi planlıyor.

4.4. Uygulama Tasarımı

Kurumsal uygulamalar çok büyük miktarda veriyle ilgilenir. Bu tür ve miktarda veri ile performans ilk akla gelen problemdir. Performansın dikkate alınması gerektiğinde, uygulamanın mimarisi daha önemli hale geliyor. Veritabanının tasarımı ve uygulamanın tasarımı birbiriyle uyumlu olmalıdır. Uygulamanın her iki ana kısmı da diğeri düşünülerek tasarlanmalıdır. Böyle bir uygulama

geliştirilirken uygulamanın daha küçük parçalara bölünmesi gerekir, bu kısımlara katman adı verilir. Kurumsal uygulamalar çoğunlukla üç katmandan oluşur; veritabanı işlemlerinin gerçekleştirildiği veritabanı katmanı, hesaplamaların ve veri dönüşümlerinin yapıldığı iş katmanı, kullanıcının sistemle etkileşime girdiği arayüz katmanı. Şekil 4.1'de görüldüğü gibi katmanlar sırayla yerleştirilir ve veri işlemleri bu sırayla gerçekleşmelidir. İlk kullanıcı web sitesinden, üst katman olan bir komut yerleştirir. Bussines katmanı, girdileri işler ve girdiyi gerekli nesne tipine dönüştürür ve veri katmanındaki işlevi çağırır ve girdileri veri katmanı işlevine iletir. Veri katmanı işlevi, veri tabanını bağlar ve veri tabanı motoruna sorgu cümleleri gönderir ve veri tabanı motoru, veri tabanı üzerinde ve eğer iş katmanına iletilen dönüş değerleri varsa cümleleri çalıştırır.



Şekil 4.1. Uygulamanın ana katmanları.

4.4.1. Veri Erişim Katmanı

Proje veritabanı ilişkisel olmalı, tüm nesne türleri veritabanı tablolarında olmalı ve hepsi uygulama ile ilgili olması gerekenlerle ilgili olmalıdır. Performans, hata azaltıcı etkiler ve güvenlik nedeniyle veritabanı çalışmaları işlev ve prosedürlerle çağrılmalıdır. Veri erişim katmanı Python dili ile oluşturulmuştur. Bu katman tamamen nesne yönelimli olarak oluşturulmuştur. SQL bağlantıları ve SQL komutları temel sınıfta oluşturulur ve atılır. Bu katmandaki diğer sınıflar, temel sınıftan miras alınır ve kendi özelliklerine ve işlevlerine sahiptir. Veri erişim katmanının temel görevi, veritabanına bağlanarak kullanıcıdan gelen komutu yürütüp yürütmenin sonucunu döndürmektir.

4.4.2. İş Katmanı

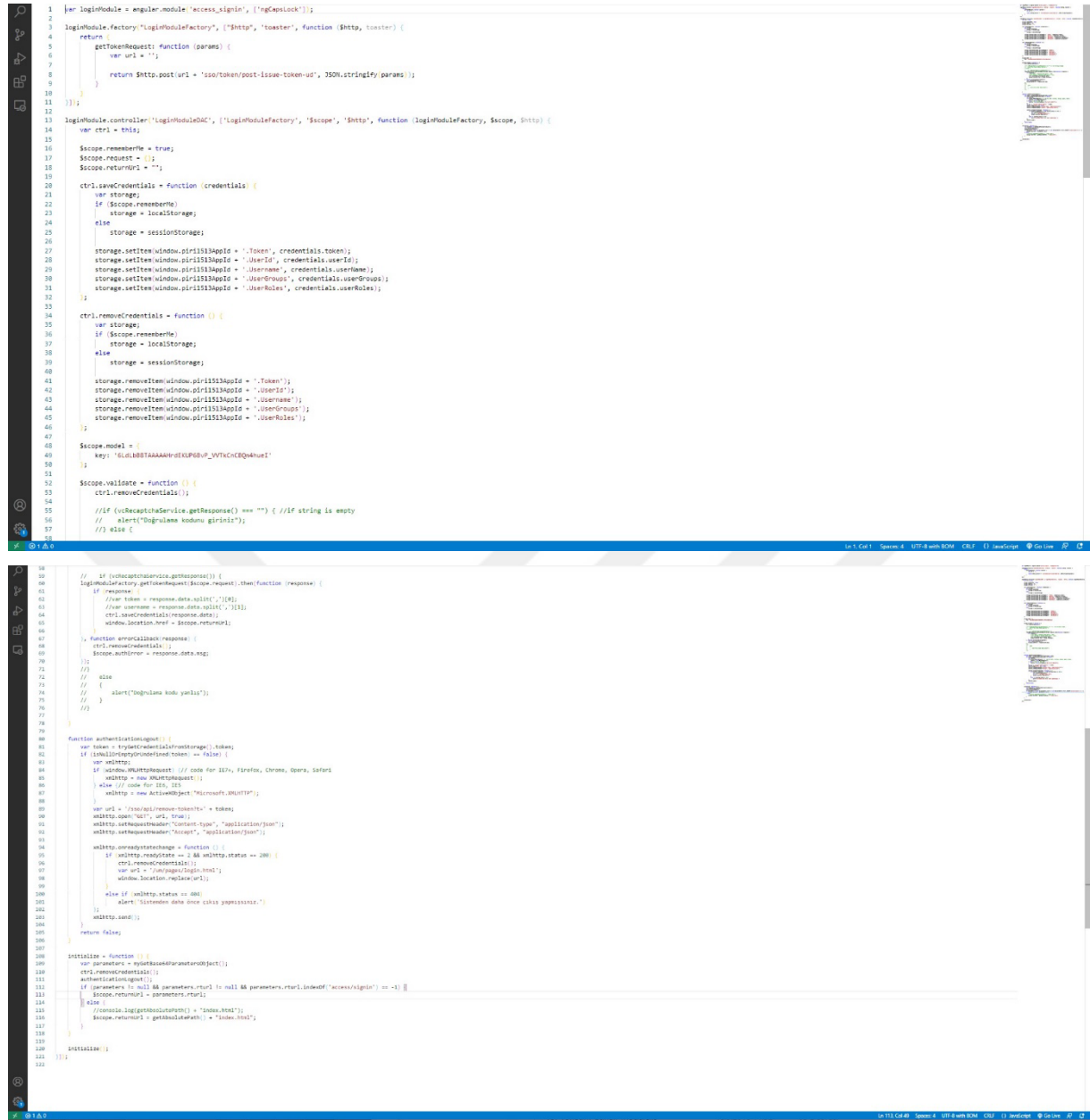
İş katmanı, veri erişim katmanı ile kullanıcı arayüzü katmanı arasında yer alır ve bu iki katmanı basitçe birbirine bağlar. İş katmanı, kullanıcı tarafından gönderilen verilerin işlendiği ve veritabanı sorguları için hazırlandığı ve veritabanından gelen verilerin işlendiği ve daha kullanıcı dostu bir şekle dönüştürüldüğü yerdir. İş katmanı da Python dili ile oluşturulmuştur.

4.4.3. Kullanıcı Arayüzü Katmanı

Arayüz katmanı, kullanıcının gördüğü ve etkileşimde bulunduğu sunum katmanıdır. Bu katman HTML, Javascript, CSS ve bazen XML, XSLT ve Active X'ten oluşur. Bu kısımda veriler kullanıcıya HTML, Javascript ve CSS aracılığı ile sunulmaktadır. Bu katman Python, Javascript, HTML 5.0 ve CSS ile geliştirilmiştir.

4.5. Proje Kodları ve Arayüzleri

4.5.1. Kullanıcı Giriş Ekranı Backend ve Frontend



Şekil 4.4. Kullanıcı Giriş Ekranı Backend.

4.5.2. Uygulama Anasayfası

Uygulamanın anasayfa görseli Şekil 4.5' te gösterilmiştir.

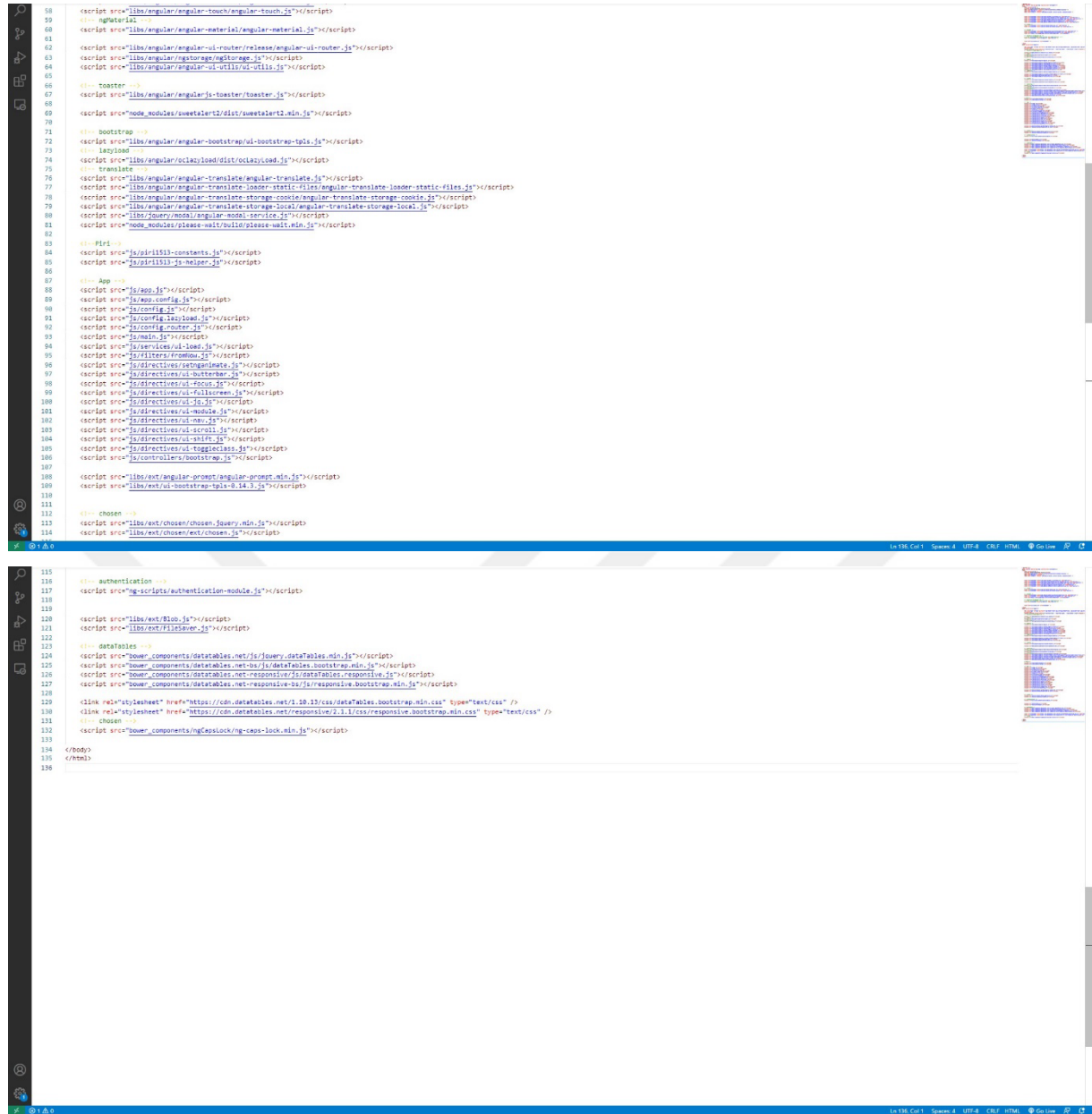
Uygulama anasayfasının ön ofis (Frontend) kodları Şekil 4.6 ve 4.7’ de gösterilmiştir.

```

1 <DOCTYPE html>
2 <html lang="en" data-ng-app="ng-controller='settingsCtrl'">
3 <head>
4   <meta charset="utf-8" />
5   <title>{{enterpriseConstants.title}} </title>
6   <meta name="description" content="{{enterpriseConstants.metaDescription}}"/>
7   <meta name="keywords" content="" />
8   <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1" />
9
10
11 <link rel="stylesheet" href="/libs/assets/animate.css/animate.css" type="text/css" />
12 <link rel="stylesheet" href="/libs/assets/font-awesome/css/font-awesome.min.css" type="text/css" />
13 <link rel="stylesheet" href="/libs/assets/linearicons/css/linearicons.min.css" type="text/css" />
14 <link rel="stylesheet" href="/libs/jquery.bootstrap/dist/css/bootstrap.css" type="text/css" />
15 <link rel="stylesheet" href="/libs/angular/angular.js/toaster/toaster.css" type="text/css" />
16 <link rel="stylesheet" href="/node_modules/sweetalert2/dist/sweetalert2.min.css" type="text/css" />
17
18
19 <!-- Chosen -->
20 <link rel="stylesheet" href="/libs/ext/chosen/chosen.min.css" type="text/css" />
21 <link rel="stylesheet" href="/libs/ext/chosen/ext/chosen_spinner.css" type="text/css" />
22
23
24 <!-- SpinKit -->
25 <link rel="stylesheet" href="/node_modules/please-wait/hold/please-wait.css" type="text/css" />
26 <link rel="stylesheet" href="/bower_components/spinkit/spinkit.css" type="text/css" />
27 <link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet">
28
29 <!-- BuildIcons css/app.min.css -->
30 <!-- <link rel="stylesheet" href="/css/font.css" type="text/css" /> -->
31 <link rel="stylesheet" href="/css/app.css" type="text/css" />
32
33
34 <link href="/css/custom.css" rel="stylesheet" />
35
36 </head>
37 <body ng-controller="AppCtrl">
38
39   <div class="page" is="app" ng-class="{ 'app-header-fixed':app.settings.headerFixed, 'app-aside-fixed':app.settings.asideFixed, 'app-aside-fisher':app.settings.asideFisher, 'app-aside-docc':app.settings.asideDock, 'container':app.settings.container}">
40     <toaster-container toaster-options="{ 'position-class': 'toast-top-right', 'close-button':true}"></toaster-container>
41
42     <!-- BuildJS JS/app-angular.js -->
43
44     <script src="/node_modules/js-cookie/src/js.cookie.js"></script>
45
46     <!-- jQuery -->
47     <script src="/libs/jquery/jquery/dist/jquery.js"></script>
48
49     <!-- Bootstrap -->
50     <script src="/libs/jquery/jquery/bootstrap/dist/js/bootstrap.js"></script>
51
52     <!-- Angular -->
53     <script src="/libs/angular/angular/angular.js"></script>
54
55     <script src="/libs/angular/angular-animate/angular-animate.js"></script>
56     <script src="/libs/angular/angular-aria/angular-aria.js"></script>
57     <script src="/libs/angular/angular-cookies/angular-cookies.js"></script>
58     <script src="/libs/angular/angular-messages/angular-messages.js"></script>
59     <script src="/libs/angular/angular-resource/angular-resource.js"></script>
60     <script src="/libs/angular/angular-route/angular-route.js"></script>
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

34

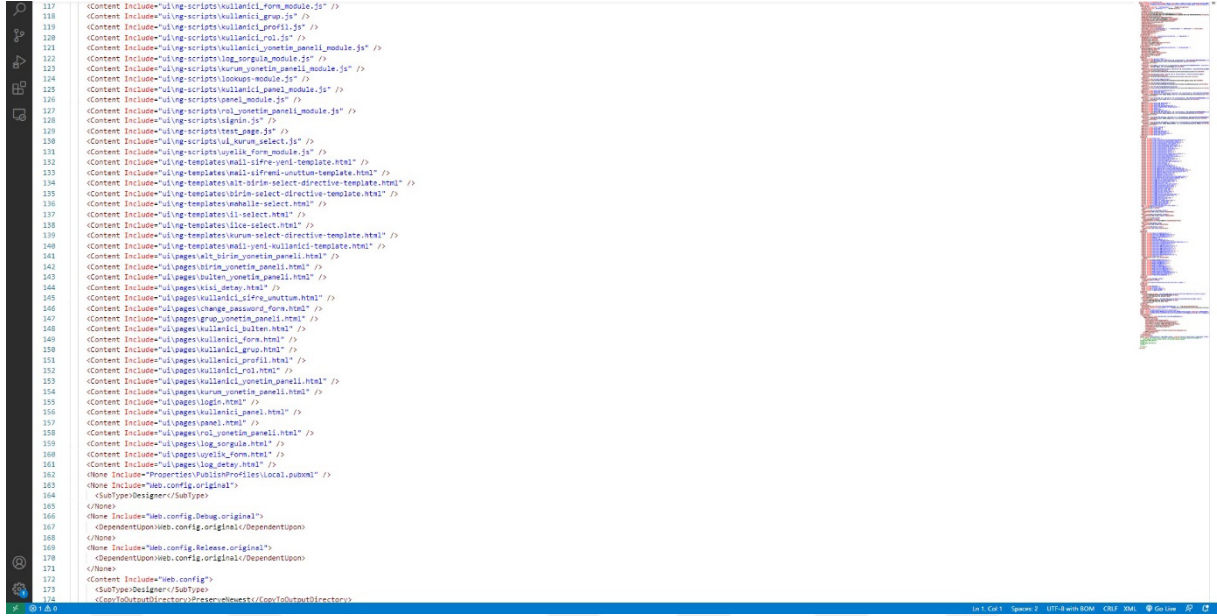


Şekil 4.7. Uygulama anasayfasının ön ofis (Frontend) kodları.

Uygulama anasayfasının arka ofis (Frontend) kodları Şekil 4.7’ de gösterilmiştir.

4.5.3. Uygulamanın Arka Ofis (Backend) Kodları

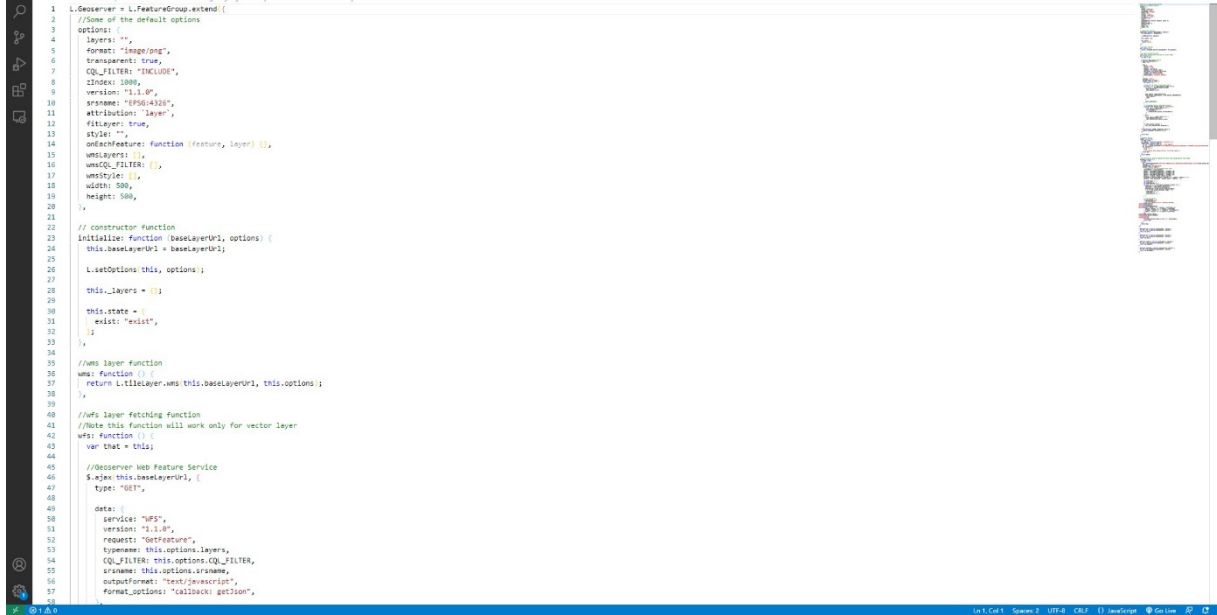
Uygulamanın arka ofis (Backend) kodları Şekil 4.8 ve 4.9’ da verilmiştir.

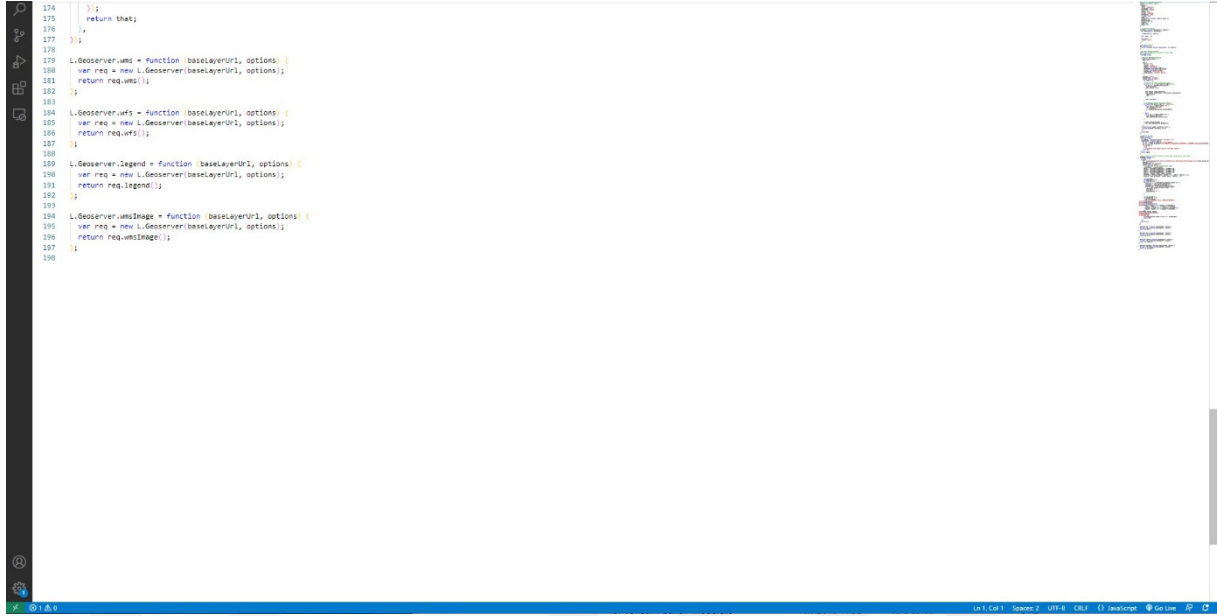


Şekil 4.9. Uygulamanın arka ofis (Backend) kodları.

4.5.4. Uygulamanın GeoServer Kodları

Uygulamanın GeoServer yani raster verilerimizin yüklendiği, cache edildiği ve tilelar oluşturulduğu sunucumuzun kodları Şekil 4.10 ve 4.11’ de verilmiştir.

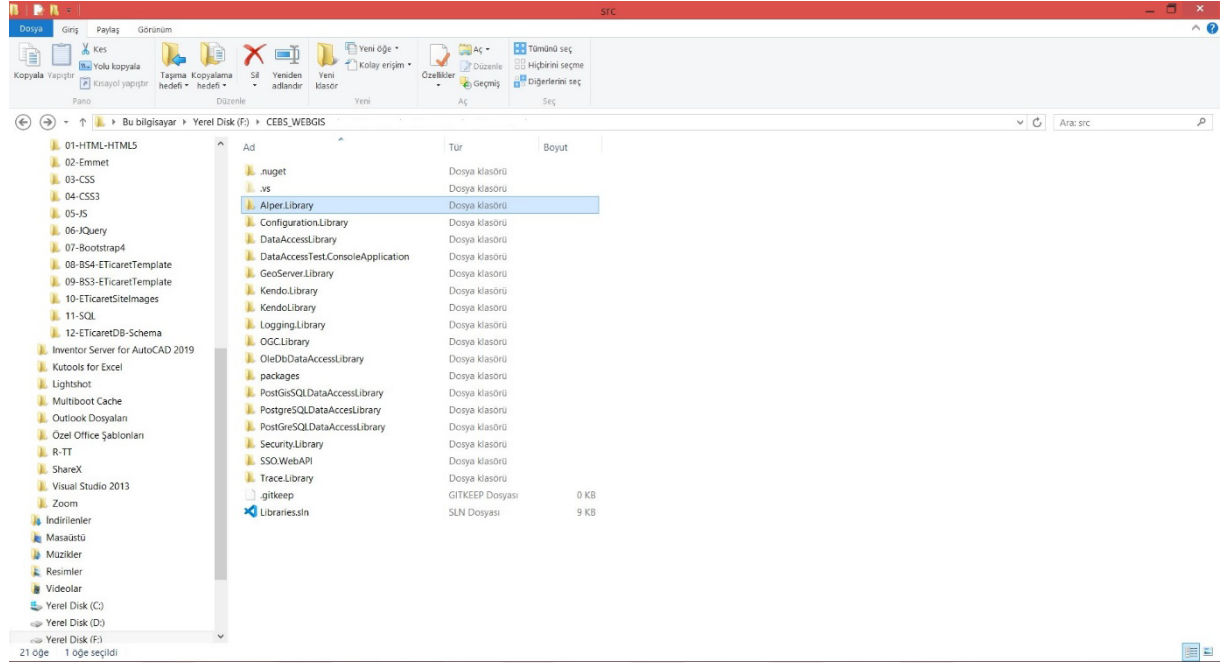




Şekil 4.11. Uygulamanın GeoServer yani raster verilerimizin yüklendiği, cache edildiği ve tileler oluşturulduğu sunucumuzun kodları.

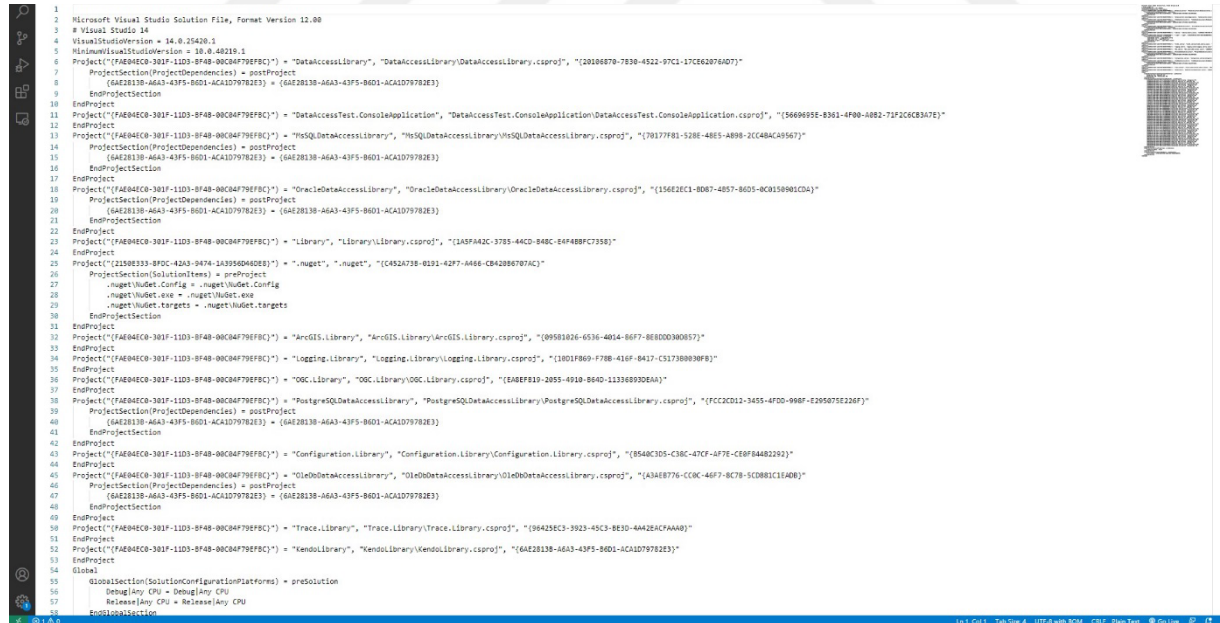
4.5.5. Uygulama İçin Yazılan DLL Listesi

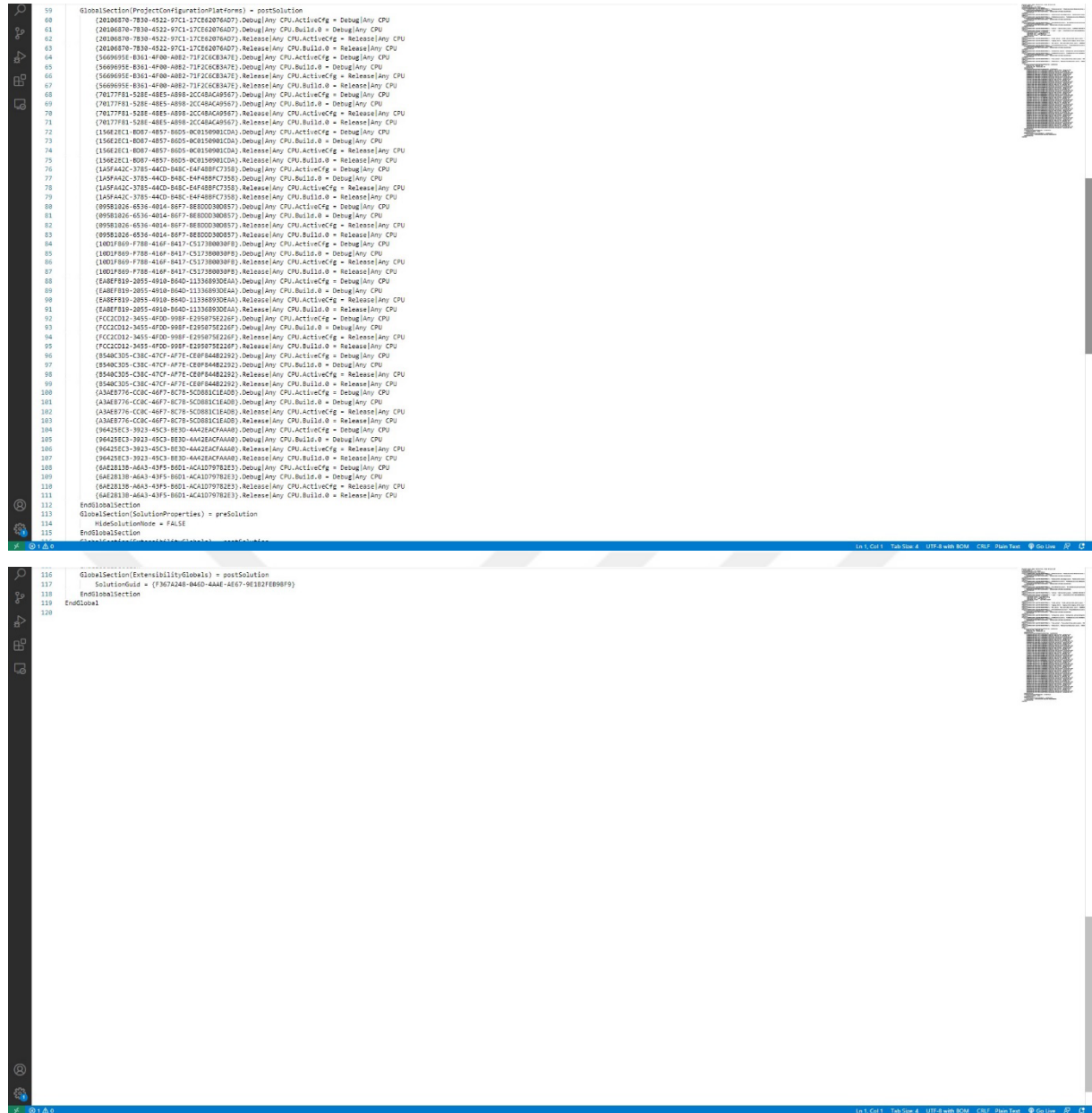
Uygulamanın sağlıklı ve hızlı şekilde çalışabilmesi için yazılmış olduğum kütüphaneler DLL olarak oluşturulmuştur. DLL (Dynamic Link Library) içinde fonksiyonlar barındıran ve eklendiği projede özel olarak yazılmış bu fonksiyonların çalıştırılabildiği kütüphanelerdir. Şekil 4.12’ de uygulama için özel olarak yazdığım DLL’ lerin listesi verilmiştir.



Şekil 4.12. DLL listesi.

Yazılan DLL’lerin uygulamaya eklenmesi (import process) için gerekli kodlar 4.14’te verilmiştir.





Şekil 4.13. DLL'lerin uygulamaya eklenmesi.

5. SONUÇLAR VE ÖNERİLER

Çalışmanın temel amacı, web tabanlı coğrafi bilgi sistemleri uygulaması oluşturmak ve bu amacı gerçekleştirerek Mersin il sınırı dahilindeki parselleri, planları, çeşitli haritaları ve yapıları yayınlayan bir web sitesi oluşturmaktır. Bu web sitesi, coğrafi bilgilerin yanı sıra metin bazlı bilgileri de içermektedir. Coğrafi bilgiler, PostGIS sunucusunda, yayınlanan haritalar GeoServer sunucusunda depolanır ve grafik olmayan bilgiler PostGRE SQL' de depolanır. Web sitesi Python, Javascript ve HTML 5.0 üzerine kurulmuştur. Python, hızlı uygulama geliştirme için en iyi çözümlerden biridir. Kullanıma hazır bileşenler, kolay hata ayıklama ve hata düzeltme, Python' ın üstün özellikleri olarak sayılabilir. Tüm bu çağdaş teknolojiler kullanılarak bir haritalama web sitesi oluşturulmuştur, site oldukça büyük ve kapsamlı bir uygulama olmakla birlikte, bilinen CBS web uygulamalarının çoğu yeteneklerine sahiptir. Ayrıca ölçeklenebilir (scalable) bir altyapıya sahiptir.

Her yazılım projesinin sorunları ve zorlukları vardır, bu yazılım geliştirmenin doğasıdır. Bir yazılım projesinin başarısı, yazılımın maliyetini ve kalitesini dengelemeye yatar. Projenin geliştirme aşamasında karşılaşılan çeşitli sorunlar vardı. GeoServer' ın konfigürasyonu yapılırken karşılaşılan ilk sorun GeoServer sunucusunun yapılandırılması zor ve bazı noktalar kolayca gözden kaçabilir olmasıdır. Bir diğer zorluk ise PostGIS sunucusunda barındırılan konumsal ve metin bazlı verilerin ilişkilendirilip sunumu yapılan harita üzerine işlenmesidir. Çünkü bu işlemlerin dökümantasyonu ne yazık ki oldukça yetersizdir. Bir diğer ciddi sorun ise performanstır. Ortalama bir internet kullanıcısının web sayfasını tarayıcıya yüklemek için sadece on altı saniye beklediği iyi bilindiğinden, performans ve hızlı yükleme web siteleri için çok önemli konulardır. Bir web sayfası anında yüklenmelidir ve doğru olmak zorundadır. Veritabanı ve ArcGIS sunucusu farklı bilgisayarlarda kurulmalı ve çalıştırılmalıdır. Performansla ilgili başka bir konu, istemci tarafı bileşenleri ve javascript ile ilgilidir. Görüntü verileriyle çalışırken tarayıcının büyük görüntü dosyaları, XML ve javascript ile ilgilenmesi gerekir. Bu büyük dosyalar tarayıcının yavaş çalışmasına ve bazen donmasına neden olur. Optimizasyon bu tür bir uygulama için çok önemlidir. Uygulama kodları amaca göre optimize edilmelidir. Hassasiyet de optimumda olmalıdır, daha yüksek hassasiyet ekstra veri yükü getirir.

Tüm önerileri bir araya getirirsek, Aktif-Aktif-Pasif olmak üzere üç düğümlü (node) bir cluster sistem üzerine CBS uygulaması kurulmalı, ayrıca bu cluster sistemin yedekli internet ve enerji hatlarına sahip olması mutlak surette gereklidir. Tüm bu veriler birbirinden farklı lokasyonlarda barındırılan biri on-site bir diğeri de off-site olan yedekleme ünitelerinde anlık olarak incremental gün sonunda da full-backup alınarak yedeklenmelidir. CBS veri detayının seviyesi tam olarak belirlenmelidir. Ekstra yüksek detaylı veriler uygulamanın performansını düşürür ancak kişilerin ihtiyaçlarını karşılayamayan düşük detaylı veriler işe yaramaz.

KAYNAKLAR

Anderberg, A. (2002). "History of the Internet and Web." <http://www.anderbergfamily.net/ant/history>.

Antenucci, J. (1991). "Geographic Information Systems: A Guide to the Technology." Springer.

Bartelme, N. (1989). "GIS-Technologie." Springer, Berlin.

Bidgoli, H. (2003). "The Internet Encyclopedia." John Wiley & Sons.

Burrough, P. (1986). "Principles of Geographical Information Systems for Land Resources Assessment." Clarendon Press, Oxford.

Davis, B. (1996). "GIS: A Visual Approach." On Word Press, Santa Fe, USA.

Erden, T. (2001). "Geodetic, Photogrammetric And Satellite Technologies - Development And Integrated Application Symposium Notes." Sofia, Bulgaria, pp. 08-09.

Farrel, A. (2004). "The Internet and Its Protocols." Elsevier.

Michael, M. (2003). "HTML/XHTML (HyperText Markup Language/Extensible HyperText Markup Language)."

Taştan, H. (1998). "Coğrafi Veri Kalitesi" Harita Dergisi, Yayın no: 120.

Voisard, A., ve David, B. (2002). "A database perspective on geospatial data modeling." IEEE Transactions on Knowledge and Data and Engineering, Vol. 14, No. 2.

Wikipedia; "Online Encyclopedia." <http://wikipedia.org/>.

Yomralıoğlu, T. (2000). "Coğrafi Bilgi Sistemleri: Temel Kavramlar ve Uygulamalar." Karadeniz Teknik Üniversitesi, Trabzon, Türkiye.