KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

WEB ÜZERİNDEN PDF DOKÜMANLARININ YÖNETİMİ

YÜKSEK LİSANS TEZİ

Celal ATALAR

OCAK 2010
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

WEB ÜZERİNDEN PDF DOKÜMANLARININ YÖNETİMİ

Bilgisayar Mühendisi Celal ATALAR

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"Bilgisayar Yüksek Mühendisi"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 31.12.2009
Tezin Savunma Tarihi : 22.01.2010

Tez Danışmanı : Yrd. Doç. Dr. Hüseyin PEHLİVAN
Jüri Üyesi : Yrd. Doç. Dr. Tuğrul ÇAVDAR
Jüri Üyesi : Doç. Dr. Ali GANGAL

Enstitü Müdürü : Prof. Dr. Salih TERZİOĞLU

Trabzon 2010

ÖNSÖZ

Günümüzde birçok bilgi bilgisayar teknolojisi ile çok hızlı bir şekilde ilgili kişi, kurum ve kuruluşlara ulaştırılmaktadır. Söz konusu kitapların veya ücretli yayınların dijital olarak dağıtılması olduğunda, bu kısımda telif haklarının korunması gerekmektedir.

Geliştirilecek elektronik kütüphane uygulaması sayesinde kullanıcıların ulaşmak istedikleri bilgileri içeren kitap veya yayınların belirli bir kısmının gösterilmesi amaçlanmıştır.

Bu çalışmanın gelişmiş bir elektronik kütüphane oluşturacak programcılara bir ışık tutmasını dilerim.

Bu çalışmada danışmanlığımı üstlenen değerli hocam Yrd. Doç. Dr. Hüseyin PEHLİVAN'a ilgi, alaka ve yardımlarından dolayı teşekkürü bir borç bilirim. Ayrıca her zaman yanımda olan aileme ve dostlarıma da destekleri için teşekkür ederim.

Celal ATALAR
Trabzon 2010

İÇİNDEKİLER

ÖZET

Birçok kurum, kişi ve kuruluş kullandığı dokümanlar üzerinde hızlı bir erişim, yönetim ve organizasyon sağlamayı istemektedir. Bu amaçla, kullandıkları dokümanları fiziksel dosya veya klasörlerde saklamak yerine bilgisayar destekli elektronik dosyalama sistemini kullanırlar. Taşınabilir doküman formatı (PDF), diğer elektronik dosyalama sistemlerine göre birtakım üstün özellikler içerir. Bu çalışmada; elektronik kitaplar, tezler ve teknik raporlar gibi çeşitli PDF dokümanları içerisinde veya dokümanların sayfa imleri üzerinde bir sözcük ya da sözcük öbeğinin arama işleminin yapılması ve sonuçların yeni bir PDF dokümanı olarak üretilmesi gösterilmiştir. Bu yapı; PDF formatında doküman oluşturma, dokümanın çıktı biriminin belirlenmesi, dokümanın açılması, dokümana içerik eklenmesi, doküman içeriği veya sayfa imleri üzerinde aranan bir metinin bulunması, veritabanı içindeki bir tablo üzerinde gerçekleştirilen bir sorgu sonucunun PDF dokümanı olarak üretilmesi ve PDF dokümanının belirli bir kısmının görüntülenmesi gibi parçalardan oluşur. Java programlama dilinde geliştirilen çatı, JSP sunucuları üzerinde çalıştırılacak web uygulamalarına kolaylıkla yerleştirilebilir.

Anahtar Kelimeler: Elektronik arşivleme, Elektronik kütüphane, PDF, iText, PDFBox

SUMMARY

Management of PDF Documents over Web

A lot of corporations, persons and establishments is wanted to provide controlling, management, organization of the files that they use. That's why, instead of the files that they use keeping them on physical storage devices, they use computer-aided electronic filing system. Portable Document Format (PDF) includes more superior features other than electronic filing systems. In this practise, it is shown that searching a word or a phrase and generating and showing the results as a new PDF document in such as electronic books, arguments, technical reports etc. PDF documents or over the bookmarks of documents. This structure is formed the parts such as generating PDF documents, determining document's output unit, opening document, adding content to document, document content or finding a phrase that it is searched over the bookmarks, generating as a PDF file that the result of a inquiry over a table in the database and shown a particular length of a PDF document. The structure that it is developed in Java Programming Language is placed easily in web applications that it is running on JSP servers.

Key Words: Electronical archiving, Electronic library, PDF, iText, PDFBox

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

SEMBOLLER DİZİNİ

AA	Adobe Acrobat
AES	Gelişmiş Şifreleme Standardı (Advanced Encryption Standard)
AI	Adobe Illustrator
AL	Apache Lisansı (Apache License)
API	Uygulama Programlama Arayüzü (Application Programming Interface)
AR	Adobe Reader
ASF	Apache Yazılım Derneği (Apache Software Foundation)
CSS	Stil Şablonları (Cascading Style Sheets)
DRM	Dijital Hak Yönetimi (Digital Rights Management)
FDF	Formlar Veri Biçimi (Forms Data Format)
FTP	Dosya Transfer Protokolü (File Transfer Protocol)
HTML	Bağlantılı Metin İşaretleme Dili (HyperText Markup Language)
HTTP	Hiper Metin Transferi Protokolü (HyperText Transfer Protocol)
IE	Internet Explorer
IET	Mühendislik ve Teknoloji Enstitüsü (Institution of Engineering and Technology)
ISO	Uluslararası Standartlar Örgütü (International Organization Code for Standardization)
JSP	Java Sunucu Sayfaları (Java Server Pages)
LGPL	GNU Genel Kamu Lisansı (The GNU Lesser General Public License)
LZW	Lempel-Ziv-Welch
MO	Microsoft Office
MPL	Mozilla Kamu Lisansı (Mozilla Public License)
MS-DOS	Microsoft Disk İşletim Sistemi (Microsoft Disk Operating System)
NASA	Ulusal Havacılık ve Uzay İdaresi (National Aeronautics and Space Administration)
OC4J	Oracle Container for J2EE
ORC	Optik Karakter Tanıma (Optical Character Recognition)
PDF	Taşınabilir Doküman Formatı (Portable Document Format)
PS	PostScript

RTF	Zengin Metin Belgesi (Rich Text Format)
SMTP	Elektronik Posta Gönderme Protokolü (Simple Mail Transfer Protocol)
URL	Tekdüzen Kaynak Bulucu (Uniform Resource Locator)
XDP	XML Veri Paketi (XML Data Package)
XFA	XML Form Mimarisi (XML Forms Architecture)
XML	Genişletilebilir İşaretleme Dili (eXtensible Markup Language)

1. GENEL BİLGİLER

Arşivleme çok eski yıllardan beri insanlık tarihi açısından çok önemli bir yer tutmaktadır. Gerek gelecek nesillere ışık tutmak gerekse sonradan tekrar kullanılmak üzere oluşturulan veya üretilen bilgilerin arşivlenmesi gerekmektedir. Arşivleme işlemleri çok eski tarihlerden beri kağıt üzerinde yapılmaktadır. Arşivlemeye en güzel örnek olarak büyük bir kütüphane düşünülürse kitap veya belgelerin çok fazla alan kaplayacağı hemen fark edilebilir. Bunun yanında aranılan herhangi bir bilgiye ulaşmak da çok kolay olmamaktadır.

Bilgisayar teknolojilerinin hayatımızın her alanına dahil olmasıyla birlikte diğer birçok alanda olduğu gibi arşivleme alanında da çok büyük gelişmeler kaydedilmiştir. Bilgilerin elektronik ortam saklanması fikri ortaya atılmış ve çok hızlı bir gelişim göstermiştir.

1.1. Elektronik Arşivleme

Bilgisayar teknolojisinin kullanılmaya başlanması birlikte birçok kurum, kişi ve kuruluş dokümanlarını daha organize bir yapı olan, elektronik dosyalama şeklinde oluşturmayı ve arşivlemeyi uygun görmüşlerdir. Elektronik dosya yapısının avantajları şu şekilde sıralanabilir:

- Doküman ve dokümanlar içinde arama yapıldığında hızlı ve etkili sonuçlar geri döndürülmesi sağlanır. Dokümanların kendilerine ait bir indeksleme yapıları bulunmasa bile, bu özelliği sağlayacak birçok program bulunmaktadır.
- Dokümanların arşivlenmesi konusunda büyük kolaylıklar sağlar.
- Dokümanların birçok kişi tarafından kolaylıkla kullanılmasını sağlanır. Elektronik dosyalar bir web sitesine yerleştirilerek veya e-mail yoluyla birçok kişiye ulaştırılabilir.

Elektronik dosya arşivleme yapısı beraberinde bazı problemleri de getirmiştir. Elektronik ortamda dokümanların çok kolay bir şekilde alış verişi telif hakkı ve yasadışı kopyaların oluşturulması gibi problemleri ortaya çıkarmıştır. Elektronik dosya yapısı büyük kolaylıklar sağlasa da belgenin aslı elektronik olanına göre doğruluk ve güvenilirlik

açısından daha önemlidir. Diğer önemli faktör ise bir belgeyi kağıttan okumak, bilgisayar ekranında okumaya göre oldukça kolaydır.

Geliştirilen yeni teknolojiler sayesinde büro, ofis vb. yerlerde kağıt kullanılmadan verilerin elektronik ortamda saklanması amaçlanmaktadır. Bu teknolojiler elektronik dosyalara dijital imza eklenmesi gibi şirketler ve hükümetler tarafından giderek kabul gören bir yapıyı da beraberinde getirmiştir ve bu sayede telif haklarının korunması sağlanmıştır. Böylece elektronik dokümanlar daha saygın ve güvenilir hale gelmiştir. Bu sürecin öncülerinden bir tanesi Adobe Systems Incorporated firmasıdır.

1.1.1. PDF Formatının Oluşturulması ve Adobe Dokümanları

Adobe firması 1982 yılında John Warnock ve Chuck Geschke tarafından kurulmuş ve ilk ürünleri dijital fontlar olmuştur [1]. Günümüzde Adobe Creative Suite (Photoshop ve Illustrator) ve Acrobat şirketin en önemli ürünleridir.

Elektronik dosya oluşturma yeteneğine sahip PostScript (PS) programlama dili Adobe firması tarafından 1985 yılında üretilmiştir. Bu dilin amacı; elektronik dosyaya metin, şekiller ve basit resimler yerleştirmektir. Ayrıca, belgelerin yazdırılabilmesi için bir yazıcı kontrol yapısı içermektedir [2].

Aynı yıl içinde Macintosh bilgisayarlar için Adobe Illustrator (AI) isimli kendi dosya formatına sahip vektör tabanlı çizim yapabilen uygulamayı geliştirmiştir. AI, PS üzerinden üretilmiştir. 1989 yılında AI, Windows işletim sisteminde çalışabilir hale getirilmiş ve grafik endüstrisinde önemli bir yere gelmiştir.

Yüksek kaliteli görsel materyaller üretmek, uzun süre boyunca uzmanların öncelikleri arasında yer almıştır. Fakat PS ve Illustrator uygulamalarının geliştirilmesiyle birlikte bilgisayarı olan herhangi birisi üst seviyede doküman oluşturabilir hale gelmiştir. Bu iki teknoloji ile birlikte Adobe firması masaüstü bilgisayarlar için basın yayın devrimini gerçekleştirmiştir. Buna rağmen Adobe kurucuları bir şeylerin eksik olduğunu düşünüyorlardı. 1991 yılında John Warnock tarafından kaleme alınan "Camelot Paper" isimli kitapta şu sözlere yer verilmiştir: "Birçok program için genel sorun her türlü yazıcı için baskı işlemini sağlamaktır fakat bu dokümanların elektronik ortamda iletilmesini ve görüntülenmesini sağlayacak evrensel bir platform bulunmamaktadır. Oysaki her endüstri çok geniş bir yelpazedeki ağ teknolojileri, işletim sistemleri, bilgisayar konfigürasyonları üzerinden kötü de olsa dokümanların iletilmesini sağlayacak evrensel bir yapıya ihtiyaç

duymaktadır [3].” Bunun üzerine Adobe bünyesindeki mühendisler PS ve AI teknolojilerini kullanarak yeni bir doküman formatı ve her platformda görüntülenmesini sağlayacak bir tümleşik yazılım paketi geliştirdiler.

Bu yeni doküman formatı orijinal olarak “Değiştirilmiş PS” olarak adlandırılır ve taşınabilir doküman formatı (PDF) olarak bilinir. PS ve PDF birbirleri ile bağlantılı olmalarına rağmen farklı formatlara sahiptirler. PDF, PS gibi bir programlama dili değildir. PDF, PS dilinin karışık metinleri, grafikleri yorumlaması, dokümanın ekrana ve yazıcıya iletilmesi gibi özelliklerini kullanır [4].

Bir PDF dokümanı bir dizi sıralı sayfadan oluşur ve her sayfa metin, yazı tipi tanımlaması, sayfa kenar boşlukları, sayfa düzeni, grafiksel elemanlar, yazı rengi ve arka plan içerir. PS dilinin aksine, PDF linkler ve diğer bağlantı bilgileri gibi birçok doküman yapısı içerir ve sayfa bağımsız bir yapıya sahiptir. Çünkü PS bir programlama dilidir ve bir sayfa üzerinde yapılan değişiklik bütün sayfaları etkileyecektir ve bütün sayfalar aynı yapıda görüntülenecektir. PDF ise her sayfayı birbirinden bağımsız şekilde çizdirebilir.

PDF dosya formatı herhangi bir platformda (Unix, Macintosh, Windows, Linux veya Palm işletim sistemlerinde) görüntülenebilir ve yazdırılabilir. Bir PDF dokümanı bu platformların hepsini benzer olarak görür.

Camelot , Adobe firmasının Acrobat isimli ürününün orijinal ismidir. Acrobat ile PDF yapısını birbirine karıştırmamak gerekmektedir. Acrobat zaman içerisindeki PDF tanımlamaları ile geliştirilmiştir [5].

1.1.2. Acrobat Ailesi

Adobe firması Acrobat ailesini dokümanlarınızı oluşturup değiştirebileceğiniz, yorumları, düşünceleri ve eleştirileri toplayıp karşılaştırabileceğiniz ve bir dosyayı güvenli bir şekilde paylaşabileceğiniz bir uygulama paketi olarak tanımlar.

Kullanıcılar tarafından genellikle Adobe firmasına ait Adobe Reader (AR) isimli ürünü kullanılır. Ücretsiz olan bu program sayesinde PDF dokümanlarının görüntülenmesini sağlanır ve birçok işletim sistemi üzerinde çalışabilir. Bu program tek başına kullanabileceği gibi tarayıcıya uyumlu ek bir program olarak kullanılıp PDF dokümanlarının tarayıcı içinde görüntülenmesi sağlanır. AR ile PDF dokümanlarını görüntüleyebilir, yazdırabilir ve dokümanlar üzerinde arama yapılabilir fakat doküman oluşturamaz ve herhangi bir doküman üzerinde değişiklik yapamazsınız. AR ile Acrobat

genellikle birbirine karıştırılır. Örnek olarak ücretsiz bir okuyucunun herhangi bir PDF formuna verileri kaydetme özelliğine sahip olduğu düşünülmektedir.

AR dışında okuyuculara Preview, Ghostview ve Foxit örnek olarak verilebilir. Fakat bu okuyucular AR kadar yetenekli değildirler.

Bir dokümanı tasarlamak genellikle geliştiricinin işi değildir. Bu tipik olarak bir grafik tasarımcısının aşağıdaki Acrobat ürünlerini kullanarak gerçekleştireceği bir işlevdir. Bu ürünler aşağıdaki şekilde belirtilmiştir:

- Adobe Acrobat (AA) Elements ile PDF dokümanları görüntülenebilir, yazdırılabilir, üzerinde arama yapılabilir ayrıca herhangi bir uygulama yazdırılan dosyalardan PDF dokümanları oluşturulabilir. Microsoft Office (MO) uygulamasına ait özelleştirilmiş içeriği yönetebilir, şifreler ve kişi bazında özel izinler ile dokümana erişim hakları ayarlanabilir.
- Adobe Acrobat Standart, Adobe Elements ile aynı işlevselliğe sahiptir. Ayrıca çoklu kullanıcıdan gelen yorumları sıralama ve filtreleme araçları ile organize edebilir, uygulama dosyalarını tek bir PDF dokümanı içinde birleştirebilir, dokümanları dijital olarak imzalayabilir ve onaylayabilir, çeşitli Microsoft ürünlerine ait özelleştirilmiş içeriği düzenleyebilir.
- Adobe Acrobat Professional, Adobe Standart ürününe ek olarak vurgulayıcı, yapışkan not, kalem ve diğer yorumlama araçları kullanımına ve Adobe LiveCycle Designer ile akıllı formların üretilmesine izin verir.
- Adobe LiveCycle Designer AutoCAD, Microsoft Visio ve Microsoft Project gibi teknik çizim uygulamalarına ait nesne verilerini ve tabakaları yönetebilir.
- Adobe Distiller, PS yapısındaki bir dokümanı PDF formatına dönüştürmeyi sağlar.
- Adobe Capture güçlü bir optik karakter tanıma (ORC) aracıdır. Bu sayede çok fazla sayfaya sahip dokümanlar tarayıcı yardımıyla aranabilir PDF dosyaları haline getirilebilir.

Bu ürünlerin hepsi ticari ürünlerdir ve kullanabilmeniz için belirli bir lisans ücreti ödemeniz gerekmektedir.

1.1.3. PDF Tanımlamasının Akıllı Özelliği

Adobe firması PDF tanımlaması için telif haklarını elinde bulundurur. Fakat PDF kullanımını teşvik etmek için çeşitli ürünler ve uygulamalar arasındaki bilgi alış verişinde herhangi bir kişiye aşağıdaki telif hakları izinlerini verir [6]:

- İçeriği PDF formatına uygun olan dosyalar hazırlamak
- PDF biçiminde girdi alan, bunları görüntüleyen, yazdıran ve içeriğini yorumlayan yazılımları yazmak
- Üstte açıklanan amaçlar için Adobe firmasının telif haklarını elinde bulundurduğu veri yapıları ve operatörlerini kopyalamak (Örnek kodlar ve PS dili fonksiyon tanımlamaları da buna dahildir).

Bu telif hakları izinlerine uygulanan koşullar aşağıdaki gibidir:

- PDF dokümanını girdi olarak alan yazılımların yazarları, oluşturdukları yazılımın erişim izinlerine uygunluğunu sağlamak amacıyla makul girişimlerde bulunmalıdır. Bu erişim izinleri doküman yazarının sunduğu haklardır. Yazarın niyetine uygunluğu kontrol etmek PDF dokümanını girdi olarak kullanan yazılımın sorumluluğu altındadır.
- Telif haklarına sahip olan veri yapıları ve operatörler kümesini kullanan herhangi birisi, uygun telif hakları bildirimini belirtmek zorundadır.

Üstte anlatılan genel fikir: Herhangi bir uygulama geliştirici PDF dosyalarını görüntülemek, oluşturmak ve değiştirmek amacıyla araçlar geliştirmekte serbesttir.

1.2. PDF Türleri

PDF; grafik sanatları endüstrisi, basın-yayın firmaları ve hükümetlerin içinde bulunduğu birçok sektörde fiili standart haline gelmiştir. Her bir sektörün farklı ihtiyaçları ve beklentileri bulunduğundan orijinal PDF tanımlamasından farklı PDF tipleri türetilmiştir. PDF tanımlamasının bazı alt parçaları uluslararası standartlar örgütüne (ISO) göre modellenmiştir.

Burada belirtilecekler dışında olan PDF türleri o kadar yenidir ki neredeyse hiçbir araç tarafından desteklenmemektedir.

1.2.1. Geleneksel PDF

Geleneksel PDF, değişmeyen içeriği ve yazdırılmaya hazır düzeniyle bitmiş bir ürün olarak düşünülebilir. Zengin metin belgesi (RTF) ve bağlantılı metin işaretleme Dili (HTML) biçimlerinin aksine ekranda nasıl gözüküyorsa kağıda basıldığında da aynı görünüme sahiptir. RTF ve HTML biçimlerinin çıktıları, onları görüntüleyen uygulamaya göre değişiklik gösterebilir.

Geleneksel PDF, her tipteki çoklu ortam, bağlantı, yer imi vb. içeren salt okunur ve sayfalandırılmış doküman biçimidir. Geleneksel PDF, içerisindeki yazının yapısı hakkında bilgi içermez. Örneğin geleneksel PDF tablo kavramını bilmez. PDF dosyası içinde tablolar gösterilebilir fakat tablo içindeki çizelgesel bilgi başka bir uygulamada kullanılmak amacıyla elde edilemez. Orijinal içeriği elde etmek için özelleştirilmiş OCR yazılımlarına ihtiyaç vardır. Kısaca geleneksel PDF oluşturmak tek yönlü bir işlemdir.

1.2.2. Etiketlenmiş PDF

Bazı durumlarda geleneksel PDF ihtiyaçları karşılamaz. PDF dosyaları kullanılacakları aygıta uyum sağlayacak şekilde üretilmek istenebilir. Örneğin son kullanıcılar dokümanları ekranı küçük olan el bilgisayarlarında görüntülemek isteyebilir. Doküman, görme özürlüler tarafından erişilebilir olması için mantıksal okuma sırası içermelidir. Resimler alternatif tanımlamalarla verilmelidir. Ayrıca paragraf ve tablo gibi doküman yapıları ayırt edilmek istenirse etiketlenmiş PDF formatına ihtiyaç duyulur.

Etiketlenmiş PDF, farklı amaçlar için sayfa içeriğinin ayıklanması ve tekrar kullanılmasına izin veren standart veri tipleri ve özellikleri kümesi tanımlayan biçimlendirilmiş bir PDF tipidir. Sayfa içeriği, karakterler, kelimeler ve yazının hatasız olarak belirlendiği bir biçimde temsil edilir. Etiketlenmiş PDF tipinde temel bir yerleşim modeli, standart yapı bileşenlerinin ve özelliklerinin bir kümesi bulunur.

1.2.3. Doğrusallaştırılmış PDF

Doğrusallaştırılmış PDF, etkin artımlı erişime izin verecek şekilde özel bir yolla organize edilmiştir. Böylelikle dokümanın görüntülenme performansı artar. Dokümanın ilk

sayfasını hızlı bir şekilde göstermek esas amaçtır. Eğer bir sayfa verisi yavaş bir kaynaktan geliyorsa sayfa içeriği veri geldikçe gösterilir.

1.2.4. Yerel Düzenleme Yeteneğini Koruyan PDF

AI uygulaması kullanılarak dokümanlar PDF olarak kaydedilebilir. Bu yolla kaydedilen PDF dosyaları AI ile açılarak düzenlenebilir. Fakat bu PDF dosyaları genel kullanım için uygun değildir. Geleneksel PDF ile karşılaştırıldığında daha fazla yer kaplarlar çünkü uygulamaya özgü birçok veri içerirler.

1.2.5. ISO Standardına Sahip PDF Türleri

Geçerli bir PDF dosyası oluşturmanın birçok yolu bulunmaktadır. Bu özgürlük bir avantaj olabileceği gibi bazen de dezavantaj olabilir. Bütün geçerli PDF dosyaları her içerik içinde kullanılabilir değildir. Bu problemi çözmek için farklı ISO standartları oluşturulmuştur.

1.2.5.1. PDF/X

Özellikle basın-yayın sektörü PDF dosyalarındaki bu çok çeşitliliğin kısıtlandırılması gerektiğini düşünmüştür. Bunun sonucunda toplanan konsorsiyum ISO (ISO 15930-1, -2, ve -3) standartlarını kullanan PDF/X' i kabul ettiklerini açıklamıştır. Bu yapının amacı; teknik detaylardan bağımsız bir şekilde PDF dokümanlarının basımı için bir servis sağlamaktır.

1.2.5.2. PDF/A

ISO (19005-1:2005) standardını içeren PDF/A, elektronik dosyaların uzun zamanlı kullanılmasını sağlamak diğer bir ifadeyle arşivleme amacıyla kullanılır. Örneğin; on yıllık bir Microsoft Word dosyasını Word' ün son sürümlerinden biriyle açmaya çalışırsanız, açılan doküman görüntüsü oluşturulduğu sürümdeki görüntüsüyle aynı olmayacaktır. Çoğu kelime işlemci programının aksine, PDF yalnızca verileri değil aynı zamanda dokümanın formunu da barındırır. Bu sayede, PDF dokümanları sürüm farklılığından bağımsız, geriye

dönüşümlü olarak görüntülenebilir. Örneğin; PDF 1.2 sürümünde yapılmış bir doküman PDF 1.6 sürümüne sahip bir uygulamada rahatlıkla okunabilir.

1.2.5.3. PDF/E

Mühendislik belgelerindeki akış diyagramlarının kurulabilmesi için PDF 1.6 sürümüne göre tanımlanmış bir standarttır.

1.2.5.4. PDF Formları, FDF ve XFDF

Bir PDF dokümanı etkileşimli formlar içerebilir, çoğu zaman bu formlar AcroForm olarak adlandırılırlar. AcroForm , kullanıcıdan etkileşimli olarak bilgi almayı sağlayacak bir yapıdır. Eğer tek forma sahip HTML sayfanız varsa, PDF dokümanınız bir web sitesi üzerinden etkileşimli olarak çalışabilir. Kullanıcı form içindeki onay düğmesine tıklarsa girilen veriler web sunucusuna farklı formatlarda (AcroForm içinde tanımlanmış çalışma şekline göre) gönderilir.

- HTML sorgu öbeği olarak
  -key1=value1&key2=value2&...
- Formlar Veri Biçimi (FDF) olarak
  Bir FDF dosyası form üzerindeki verileri tutar ve AcroForm içeren PDF dosyasına bir bağlantı içerir. FDF dosyası AR uygulaması ile açıldığında, orijinal PDF dosyasının üzerine FDF içinde bulunan veriler yerleştirilir.
- XFDF olarak
  FDF yapısına göre genişletilebilir işaretleme dili (XML) tabanlı alternatif bir yapıdır.
- PDF olarak
  Bu yöntemle tamamıyla doldurulmuş bir PDF dosyası web sunucusuna gönderilir. (Eğer sadece AR uygulamasına sahipseniz bunu yapmanız olanaksızdır)

1.2.5.5. XFA and XDP

Acrobat 7.0 ile üretilen formlar AcroForm yapısından farklıdırlar, XML form mimarisine (XFA) dayanırlar. XML veri paketi (XDP) , PDF içeriklerini XML olarak paketlemek için bir mekanizma sağlar. XFA kaynakları PDF dosyası içindeki XDP paketleri olarak tanımlıdırlar. Bu durumda; PDF dosyasına sahip XML formatında formlar tasarlanır.

1.3. PDF Sürümlerinin Gelişimi

İlk oluşturulan PDF sürümü belli başlı birkaç işlemi gerçekleyebilecek yeteneğe sahiptir. Zaman içerisinde gelişen teknolojiye adapte olmak ve farklı gereksinimleri karşılayabilmek amacı ile farklı sürümler oluşturulmuştur. Tablo 1.'de her bir PDF sürümüne eklenmiş yeni özelliklerin gösterildiği bir liste görülmektedir.

Tablo 1.'de halihazırdaki PDF sürüm özelliklerine yapılan eklemelerin olduğu görülmektedir. Ancak öyle bazı özellikler vardır ki bunlar farklı sürümlerin (etiketli PDF gibi) oluşturulmasına yol açmıştır.

PDF sürümlerinin farklılığına genel olarak kullanıcı birimini değiştirme, PDF içeriği ve sıkıştırma, şifreleme gibi üç başlık altında değinilebilir.

1.3.1. Kullanıcı Birimini Değiştirme

PDF dokümanının bir sayfasının genişlik ve yüksekliği kullanıcı birimi olarak değerlendirilir. Kullanıcı birimi basın-yayın ölçüsü olan birim cinsinden ifade edilir ve 72 birim bir inç'e karşılık düşer.

PDF 1.4 sürümünden itibaren sayfalar minimum 3x3 birim, maksimum 14400x14400 birim olabilmektedir. Bu değerler yaklaşık olarak minimum sayfa boyutu 0.04x0.04 inç olmak üzere, maksimum sayfa boyutunu 200x200 inç'e karşılık düşürmektedir. Bu tanımlamalar PDF 1.4 ve 1.5 sürümleri için doğrudur. Ancak Tablo 1'de PDF 1.6 sürümünden itibaren kullanıcı biriminin değiştirebileceği görülmektedir. PDF 1.6 sürümünde kullanıcı birimi minimum 1 ve maksimum olarak da 75000 birim olabilmektedir. Bir birimin varsayılan değeri 1/72 inç olduğundan kullanıcı birimi 1042 inç'e kadar çıkabilmektedir [7].

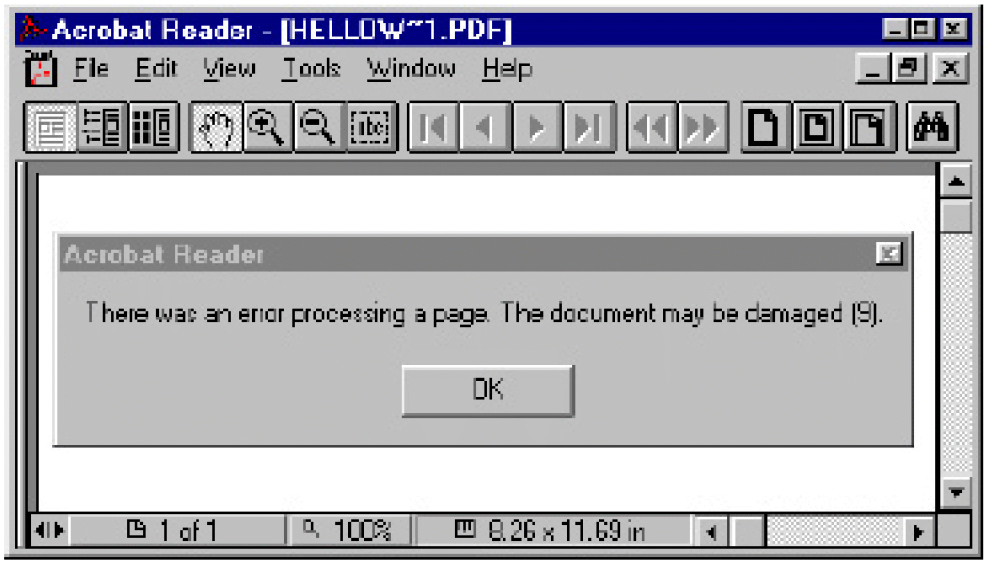
Tablo 1. PDF sürümlerinin oluşturulma yılı, Acrobat uygunluğu ve Özellikleri

PDF Sürümü	Yılı	Acrobat Sürümü	Özellikleri
PDF-1.0	1993	Acrobat 1	• Metin ve grafiklerden oluşan bir yapıyı ekrana yansıtma ve yazıcıdan çıktı alabilme
PDF-1.1	1994	Acrobat 2	• Parola korumalı PDF dosya oluşturabilme • Dış Bağlantı (Web Sayfası Adresi) ekleyebilme • Cihazlardan bağımsız renk ayarı yapabilme
PDF-1.2	1994	Acrobat 3	• (Zip/gzip) sıkıştırma desteği • Etkileşimli form desteği • Çince, Japonca ve Korece desteği
PDF-1.3	1999	Acrobat 4	• PDF dosyanıza dosya eklentisi yapabilme • Dosyalara dijital imza yerleştirebilme • Sayfa numaralandırma özelliği
PDF-1.4	2001	Acrobat 5	• 128 bit ile dosyanızı şifreleyebilme • Saydam sayfalar yapabilme • İşaretli PDF yapabilme
PDF-1.5	2003	Acrobat 6	• Şifreleme ve sıkıştırmaya özelliklerinin eklenmesi • İsteğe bağlı içerik grupları • Çoklu ortam dosyaları ekleyebilme desteği
PDF-1.6	2004	Acrobat 7	• Gelişmiş Şifreleme Standardı (AES) desteği • Yazdırma işlemi için sayfa ayarı yapabilme

Bu kullanıcı biriminde (75.000) oluşturulan PDF dokümanını Acrobat 7.0 veya daha sonraki sürümler ile düzgün bir biçimde görüntülenebilir. Acrobat 6, bu kullanıcı birimindeki sayfayı düzgün gösteremez çünkü 75.000 değerindeki kullanıcı biriminin ne anlama geldiğini bilmemektedir. Yüksek bir Acrobat sürümü ile oluşturulmuş PDF dokümanlarını düşük Acrobat sürümleriyle görüntülemeye çalışmak Şekil 1.1'deki uyarının ortaya çıkmasına sebep olacaktır. Bu uyarı son kullanıcının elindeki sürümün gösteremeyeceği yeni bir özellik içermese bile yine de belirecektir. Fakat bu uyarının belirmesi dokümanın görüntülenemeyeceği anlamına gelmemektedir. Oluşturulan doküman üst PDF sürümünden bir özellik içermiyorsa düzgün bir şekilde görüntülenebilir.

Şekil 1.1. Son kullanıcıdaki Acrobat sürümünden daha yüksek bir sürümle oluşturulmuş PDF dokümanını görüntülemeye çalışırken ortaya çıkan uyarı

Eğer üst PDF sürümüne ait özellik içeren bir dokümanını düşük bir Acrobat sürümü ile görüntülemeye çalıştığınızda, uyarı mesajı yerine Şekil 1.2'deki hata mesajını alırsınız.



Şekil 1.2. Yüksek bir PDF sürümü ile oluşturulmuş dokümanın Acrobat 2 ile görüntülenmeye çalışılması sonucu meydana gelen hata

Üretilen hata mesajı bize dosyanın hasarlı olduğunu belirmektedir. Eğer dokümanın daha yüksek Acrobat sürümleri ile başarılı bir şekilde görüntüleyebiliyorsak, mesajda gösterilenin aksine dosyanın hasarlı olmadığını bilinmektedir. Bu mesaj penceresinden sonra Acrobat kullanıcıya: "bu dosya görüntüleyicinin anlayamadığı bilgiler içermektedir" şeklinde bir mesaj penceresi daha gösterecektir.

1.3.2. PDF İçeriği ve Sıkıştırma

Tablo 1.1'e bakıldığında flate/ZIP sıkıştırmanın PDF 1.2'den itibaren var olduğunu, ancak tam sıkıştırma özelliğinin PDF 1.5 sürümüne kadar eklenmediğini görebiliriz.

PDF 1.2'den itibaren flate/deflate sıkıştırma yöntemi Acrobat tarafından varsayılan sıkıştırma olarak kullanılmaktadır. Bu, Huffman kodlama ve LZ77 sıkıştırma tabanı baz alınarak yapılan bir algoritmadır. LZ77 sıkıştırma, Lempel-Ziv-Welch (LZW)'nin ilk versiyonlarından biridir.

Dokümanın sıkıştırılması işlemleri PDF 1.5 sürümüne kadar sayfa akışı ile sınırlıdır. PDF 1.5 sürümü ile sadece sayfa akışı değil bunun yanında diğer öğeler de (çapraz-başvuru tabloları gibi) sıkıştırılabilir. Bunların haricinde sıkıştırılmamış bir şekilde oluşturulan PDF dokümanlarını sıkıştırılabilir. Ayrıca sıkıştırılmış dokümanlar da açılabilir.

1.3.3. Şifreleme

PDF'in standart güvenlik merkezi, erişim izinlerine ve bir belge için atanmış olan iki şifreye kadar izin verir. Bir kullanıcı şifresi (dokümanını açmak için) ve bir sahip şifresi (dokümanın yönetimi için) içerir. Şifreleme PDF dokümanı içinde bulunan bütün karakter katarı ve veri bloklarına uygulanır. Ancak diğer tip nesnelere uygulanmaz. Örneğin; belgenin içeriğinden çok yapısını tanımlamak için kullanılan mantıksal değerlere ve tamsayılara uygulanmaz.

Bir PDF dokümanı, şifreyi bilmeyen herkese karşı kilitli durumdadır. İzinleri değiştirmek muhtemelen çözmek için bir PDF belgesini okumak istiyorsanız sahip şifresine ihtiyacınız vardır. Sahip şifresi aynı zamanda PDF belgesini açmanıza da olanak tanımaktadır. Maksimum şifre uzunluğu 32 karakterdir. Daha uzun bir şifre de girebilirsiniz ancak sadece ilk 32 karakteri alınacaktır. Bir yada her iki şifre de boşluk değeri olabilir. Bir kullanıcı şifresi belirlemezseniz, bütün kullanıcılar herhangi bir şifre

sorulmaksızın belgeyi açabileceklerdir. Ancak doküman üzerinde izinler ve kısıtlamalar varsa bu özellikler varolmaya devam edecektir. Bu koruma sadece psikolojiktir. Şifreleme anahtarı kullanıcı şifresinden elde edilir, bu yüzden bu şifre gerçek bir güvenlik sağlamaz. İçerik, PDF yönergesinde tanımlandığı üzere şifrelenir. Böyle bir dosyayı çözmek için bir program yazılabilir fakat bu yasadışı olacaktır [8].

Dosya izinlerinin nasıl değiştirildiğini öğrenmek için PDF yönergesi incelenirse, sahip şifresi atanmamış bir dosyayı çözmenin daha kolay olduğu görülebilir. Dokümanların güvenli bir şekilde koruma altına alınması için 128 bit anahtarlama kullanılmalıdır. Ayrıca her iki şifre kullanılmalı ve ikisi için de farklı diziliş sırası kullanılarak 32 karakterin tamamı doldurulmalıdır. Eğer 32 karakterden daha az sayıda bir şifre girilirse geri kalan kısımlar PDF yönergesinde belirtildiği üzere varsayılan doldurma yöntemiyle doldurulacaktır.

Tek kelimelik sadece harf veya rakamdan oluşan şifreler kullanım kolaylığı açısından elverişlidir. Ayrıca herhangi bir klavye sürçmesine karşın dosyayı açamama ihtimalini azaltır. Ancak daha iyi bir çevresel koruma için daha komplike bir şifre seçilmelidir. Eğer sahip şifresi biliniyorsa içeriği çözme işlemi yasal olarak yapılabilir. Eğer kullanıcı şifresi biliniyorsa içeriği çözmek için yasadışı programlar kullanarak korumasız bir kopya çıkarılabilir.

Şifreleme, sıklıkla kısıtlamaları güçlendirmek için kullanılır. Şifrelemenin gücüne bağlı olarak izinler kabul edilebilir yada kısıtlanabilir. 40 bit ve 128 bit olmak üzere iki tür şifreleme vardır. Tablo 1.1'e baktığınızda 128 bit şifrelemenin ancak PDF 1.4 sürümünden itibaren mümkün olduğu görülmektedir. 128 bit şifreleme daha ince düzeyde izin ayarları yapılmasına olanak sağlar.

Bir PDF belgesini kopyalamayı yada kaydetmeyi kullanıcıya yasaklamak mümkün değildir. AR'de bulunan kaydet veya farklı kaydet düğmeleri geçersiz kılınamaz. Bu düğmelerin geçersiz kılındığı varsayılsa bile insanlar başka bir uygulama ile dosyanın kopyasını alabilirler. Eğer gerçekten bu şekilde bir korumaya ihtiyaç duyulursa Dijital Hak Yönetimi (DRM) çözümüne başvurulmalıdır. DRM programları size belgeniz üzerinde çok daha detaylı kontrol olanakları sunar. Farklı DRM yazılım sağlayıcıları vardır ancak bu programlar biraz pahalıdır.

1.4. PDFBox Kütüphanesi

PDFBox, Apache Yazılım Derneği (ASF) firması tarafından PDF dokümanlar ile ilgili işlemler yapabilmek amacıyla geliştirilmiş açık kaynak kodlu bir Java kütüphanesidir. Bu kütüphane; PDF dokümanı oluşturma, varolan PDF dokümanları üzerinde çeşitli değişiklikler yapabilme yeteneğine sahiptir. Ayrıca ayrı PDF dokümanlarını birleştirebilme, PDF dokümanlarını şifreleme ve deşifreleme, metinden PDF dokümanı oluşturabilme, PDF dokümanını yazdırabilme ve herhangi bir PDF dokümanının içeriğini elde edebilme gibi özelliklere sahiptir.

PDFBox, Apache License v2.0. koşulları altında yayınlanmıştır [9]. ASF, Ocak 2004'te Apache Lisansı (AL) 2.0 için bir güncelleme yayımlamıştır. AL (2.0 sürümünden önce Apache Yazılım Lisansı olarak adlandırılmaktaydı), ASF tarafından yayımlanan bir özgür yazılım lisansıdır. AL (1.0, 1.1 ve 2.0 sürümleri) telif hakkı koruma ve feragat uyarısı gerektirmektedir. Lisans, özgür ve açık kaynak kodlu yazılımın geliştirilmesi için kaynak kodlarının kullanımına izin vermektedir [10]. ASF tarafından üretilen yada onun herhangi bir tasarısının ürünü olan tüm yazılımlar, AL koşullarına göre lisanslanmaktadır. ASF tarafından üretilmeyen yazılımlar da zaman zaman bu lisansı kullanabilmektedir.

Diğer özgür yazılım lisanslarında olduğu gibi AL, yazılımın kullanıcılarına yazılımı herhangi bir amaç için özgürce kullanma, dağıtma, düzeltme ve yazılımın değiştirilmiş sürümlerini dağıtma hakkını vermektedir. AL, yazılımların değiştirilmiş sürümlerinin aynı lisansı kullanarak yada özgür-açık kaynak kodlu yazılım olacak şekilde dağıtılması koşulunu barındırmamaktadır. AL yalnızca alıcıyı bilgilendirme amaçlı olarak Apache lisans kodlarının kullanıldığına dair bir uyarının bulundurulmasını zorunlu tutmaktadır. Böylece, Apache lisans kodlarının değiştirilmiş sürümlerinin alıcılarının, üst sürümü almalarına gerek kalmamaktadır. Yeniden dağıtılan yazılım paketlerinde bulunması gereken iki dosya:

- Lisans Belgesi (Kendi lisansının bir kopyası)
- Uyarı (Geliştiricileriyle birlikte lisanslanmış kütüphanelerin adlarını barındıran bir "uyarı" belgesi)

Lisanslanmış her belgede, yeniden dağıtım kodlarının barındırdığı tüm özgün telif hakkı ve patent uyarıları korunmalı ve her lisanslı dosyada, bildiri değişiklikleri, değişiklik yapıldığına dair uyarı ile dosyaya eklenmelidir.

1.5. iText Kütüphanesi

Geliştirilen uygulamalarda dinamik PDF dokümanı üretmek veya çalıştırmak amacıyla kullanılan bir kütüphanedir. Bu kütüphanenin uygulamalarda kullanılacak özellikleri şu şekilde sıralanabilir:

- Bir web tarayıcısına hizmet edecek dinamik değişkenli PDF oluşturulabilir.
- Bir XML dosyasını yada veritabanını referans alacak belgeler üretilebilir.
- PDF dokümanlarda çok çeşitli özelliklere sahip haritalar ve elektronik kitaplar üretilebilir.
- Halihazırda olan PDF dokümanlara yer imleri, sayfa numaraları, filigranlar ve diğer benzer özellikler eklenilebilir.
- Halihazırda PDF dokümanlardaki sayfaları birbirinden ayırtabilir yada birleştirebilir.
- PDF dokümanlara formlar ve dijital imzalar eklenebilir.

Bu kütüphane ile bir doküman oluştururken AA gibi bir program kullanılmasına gerek yoktur. Bunun yerine çeşitli gereksinimlere ihtiyaç duyan bir PDF dokümanı oluşturmak için doğrudan uygulamadan bir API kullanılır. Bir PDF dokümanı oluşturulurken bazı sebeplerden ihtiyaç duyulacak gereksinimler şu şekilde sıralanabilir:

- Oluşturulan içerik web ortamında yayınlanacaksa, PDF dokümanının daha iyi baskı kalitesi için HTML formatında olması gerekiyorsa bunun haricinde güvenlik sebebiyle veya dosya boyutunun küçük tutulması için.
- PDF belge çok fazla sayfa sayısından dolayı herhangi bir program yardımıyla el yordamıyla üretilemiyorsa yada içerik gelişmeye kapalı ise (kullanıcı girişi referans alınarak hesaplanma durumunda).
- Belgeler gözetimsiz bir şekilde oluşturulmaya ihtiyaç duyuyorsa (toplu bir şekilde).
- İçeriğin kişiselleştirilmesi ve/veya özelleştirilmesi gerekiyorsa.

iText, Mozilla Kamu Lisansı (MPL) altında yayınlanmış bir Java kütüphanesidir. MPL, açık kaynak koduna dayanan bir özgür yazılım lisans modelidir.

1998 yılının yazında Ghent Üniversitesi, öğrenciler tarafından kullanılan bir grup bağımsız programın yeniden tasarlanarak değiştirilmesi programını başlatmıştır. O zamana kadar, öğrencilerin dereceleri ve akademik takvim sonundaki final sonuçları sadece Microsoft Disk İşletim Sistemi (MS-DOS) üzerinde çalışan bir programla giriliyordu. Bu program tarafından üretilen belgeler sadece bir yazıcı türü ile bastırılabiliyordu. Bu ideal bir çalışma sistemi değildi. Akademik personel ve onların idari personeli Windows, Linux,

Mac OS, Solaris vb. gibi bütün işletim sistemlerini kullanıyordu. Yine de onların en hassas görevlerinden biri olan öğrenci derecelendirmesi için mecburen eski basit MS-DOS'u kullanmak zorundaydılar. Üniversite, bu durumun düzeltilmesi için en uygun zaman olduğuna karar verdi ve iki yazılımcıyı tamamen bir web tabanlı çözüm geliştirmesi için işe aldı. Bu iki yazılımcı Mario Maccarini ve Bruno Lowagie'dir.

Bu iki yazılımcı Apache JServ kullanarak birkaç Java Servlet yazıp 1998 sonbaharında ilk çevrimiçi öğrenci listesini, derslerini ve notlarını gösteren bir uygulama geliştirdiler. Bu bir tarayıcıda birkaç basit, sıradan HTML kodu idi ama aynı zamanda büyük bir atılımdı. Fakat belge basma sorunu hakkında henüz bir şey yapılmamıştı.

Her tarayıcı, HTML belgeyi kendine has yöntemlerle algılar. Firefox tarafından betimlenen bir tablonun görünümü, Internet Explorer (IE) tarafından betimlenen aynı tablonun görünümü ile tamamen birbirine benzemez. Sitil Şablonları (CSS) kullanmak iyi ayarlanmış bir sonuç için yardımcı olabilir ancak burada başka bir problem vardır. Son kullanıcı CSS'i iptal etmiş, kenar boşluklarını değiştirmiş, sayfa numarası eklemiş vb. olabilir. Üstelik, son kullanıcı genellikle HTML belgenin içeriğini belgeyi tablolayan uygulamayı (Microsoft Word gibi) kullanarak el yordamıyla değiştirebilir. Bu problemi ortadan kaldırmak için HTML-Word çözüm olarak düşünülmemiştir. Statik güvenilir bir mizanpaj için değiştirilemez sonuçlar sunan bir teknolojiye ihtiyaç duyulmuştur. Bu amaçla taşınabilir, güvenli, iyi çıktı sonuçları alınabilecek bir yapı olan PDF formatına karar verildi. Bunun sonucunda gerekli işlemleri yapabilecek iText kütüphanesi yazıldı.

iText bugünlerde birçok çevrimiçi ve çevrimdışı uygulamalarda doğrudan yada dolaylı olarak kullanılmaktadır. Farkında bile olmaksızın iText'i defalarca kullanmış olabilirsiniz. iText kullanan birçok yazılım, iText'in yayılmasına yardımcı olmaktadır. Macromedia ColdFusion ile bir PDF oluşturmuşsanız, bu dosya muhtemelen iText tarafından oluşturulmuştur. Günümüzün en önemli raporlama yazılımlarından biri olan BIRT ile raporlar oluşturuyorsunuz, programın varsayılan PDF motorunun iText olduğunu göreceksinizdir. Bunun haricinde iText'i kişisel uygulamalarınızda veya ticari programlarda da kullanabilirsiniz. E-ticaret uygulamalarında öğrencileri müşterilerle, kursları ürünlerle ve dereceleri de fiyatlarla değiştirebilirisiniz. Enerji şirketleri, müşterilerinin ne kadar gaz, elektrik, su vb. tükettiğini gösteren tablolar içeren faturalar yazmak için iText kullanmaktadır. iText kütüphanesi e-devlet uygulamalarında da popülerdir. Çünkü iText, elektronik imza kullanılarak bir PDF belgeye dijital imza atmaya olanak sağlamaktadır. Finans sektörü, müşterilerine yatırımları hakkında raporlar sunmakta

veya kiralama başvurularında kullanılmak üzere formlar üretmede iText'i kullanmaktadır. Üreticiler iText'i, parça grupları ve ürün elde etmekte kullandıkları hammaddeleri tablolamak üzere kullanabilmektedir. Ayrıca otomasyona dayalı üretim sistemlerinde üretim aşamasını tamamlamak üzere kullanılan barkod sistemlerinde de iText'i kullanılmaktadır. iText ile devre şemaları ve şehir haritaları da yapılmaktadır. Ulusal Havacılık ve Uzay İdaresi (NASA), dünyanın enlem-boylam görüntülerini ve kutuptan kutuba enlemsel görüntüleri gösteren PDF üreten bir yazılımda iText'i kullanmaktadır. Ayrıca Google takvimi, takvim sayfalarını üretmek için iText'i kullanır.

iText'in ilk versiyonları GNU Genel Kamu Lisansı (LGPL) lisansı altında yayınlanmıştır [11]. Ancak iText, bilgi teknolojileri işindeki büyük firmaların ilgisini çekmeye başladığı zaman, lisansı değiştirmek zorunlu hale gelmiştir [12].

Birçok şirket avukatı, LGPL'deki kurnazca detayları kullanmaya başladığı zaman LGPL yerine MPL lisansı geri uyum için alternatif olarak seçilmiştir [13]. Temel olarak MPL lisansı iText için, uygulamalarınızda iText kütüphanesini kullandığınızı müşterilerinize bildirmek zorunda olduğunuzu ve ayrıca kütüphanenin kaynak kodlarını da nerede bulabileceklerini söylemekle yükümlü olduğunuzu söylemektedir [14]. Buna ek olarak, eğer kütüphaneyi değiştirmişseniz, yaptığınız geliştirmeleri ve hata düzeltmelerini iText topluluğu için ulaşılabilir kılmanız anlamına gelmektedir. Bu durumda bütün topluluk kazanacaktır. Çünkü sürüm yükseltilmesi ile ilgili problemler de azalmış olacaktır.

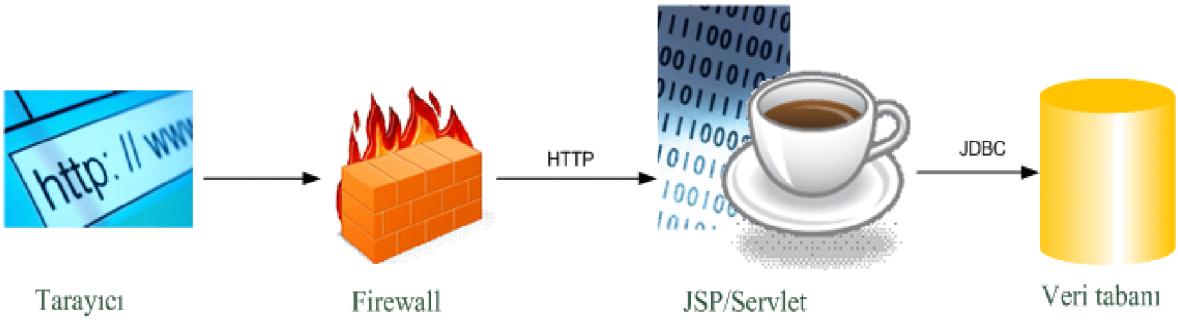
1.6. Java Sunucu Sayfaları (JSP) ve Servlet

Internet kullanımının artması ve web tarayıcılarının popülerleşmesi ile web tarayıcılar üzerinde çalışabilen uygulamaların sayısında çok fazla artış olmuştur. Önceleri web üzerinde durağan HTML kullanarak bilgi aktarımı gerçekleştirilmiştir. Teknoloji ve gereksinimlerin artması nedeniyle bu durum yetersiz olmaya başlamış ve dinamik web tabanlı uygulamaların varlığına ihtiyaç duyulmuştur [15].

Java'da web tabanlı dinamik uygulamalar geliştirmek için iki teknoloji vardır. Bu iki teknoloji Servlet ve JSP'dir. İkisi de Java teknolojisinin bütün avantajlarını kullanır. Her çağırıldıklarında tekrar tekrar yorumlanmazlar. Üzerilerinde herhangi kod değişikliği yapılmamışsa önceden hazırlanan yapı üzerinden çalıştırılırlar. Ayrıca her bir istek için

ayrı işlemler oluşturulmaz. İşlemlerin en optimum seviyede ortak olarak kullanılması sağlanır.

Servlet ve JSP'nin çalışabilmesi için ihtiyacımız olan Servlet ve JSP kodlarını çalıştıran bir web sunucusudur. Çalışması aşağıdaki gibi olmaktadır.



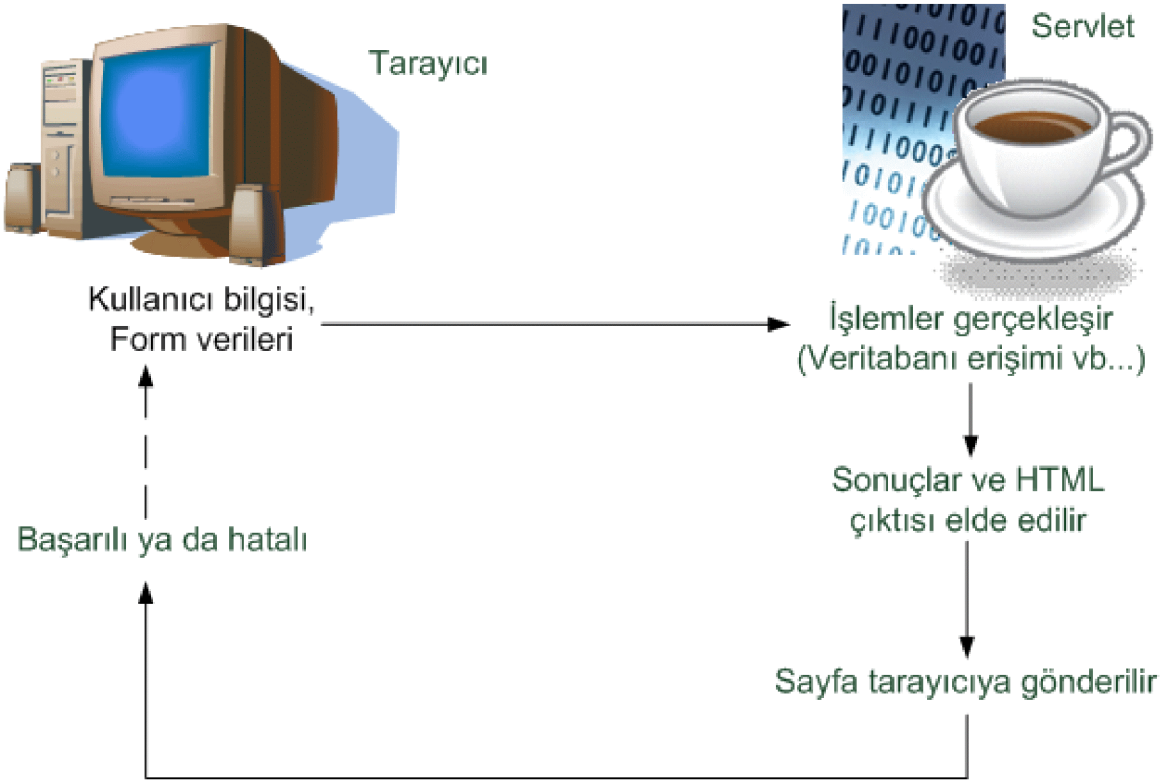
Şekil 1.3. Servlet ve JSP kodlarının çalışması için gerekli olan web sunucusunun çalışma şeması

Öncelikle tarayıcı aracılığı ile web uygulamasının adresi yazılır. Daha sonra adrese göre web uygulamasının çalıştığı web sunucusu bulunur. Web sunucusu karşılık gelen Servlet yada JSP'yi çalıştırır. JSP ve Servlet'in çalışması sonucunda elde edilen içerik kullanıcıya HTML olarak aktarılır. Kullanıcı yapılan işlemlerin hiç birinden haberdar değildir. İşlemler tamamıyla sunucu üzerinde gerçekleşir.

1.6.1. Servlet

Servlet, web sunucuları üzerinde çalışan Java sınıfıdır. Web sunucusu olarak Oracle firmasına ait Oracle Container for J2EE (OC4J) yada Apache Tomcat gibi ürünleri kullanabilirsiniz. İkisi de ücretsiz ürünlerdir. Ancak OC4J sadece web sunucusu değildir. Aynı zamanda uygulama sunucudur. Web sunucuları aracılığı ile HTML, Servlet ve JSP içeriği çalıştırılır [16].

Web sunucusu aracılığı ile kullanıcı isteğine karşılık dinamik sayfalar üretilir ve bu sayfalar kullanıcının web tarayıcısına gönderilir. Yani HTML sayfaları dönmeden önce Servlet'te yapılması gereken işlemler yapılır. İşlem sonucu elde edilen bilgi web sunucusu aracılığı ile gönderilir.



Şekil 1.4. Servlet çalışma şeması

Servlet'in temel çalışma mantığı basittir. Web tarayıcısı Tekdüzen Kaynak Bulucu (URL) adresi ile ilgili Servlet'i tarafından tarayıcıya gönderilir.

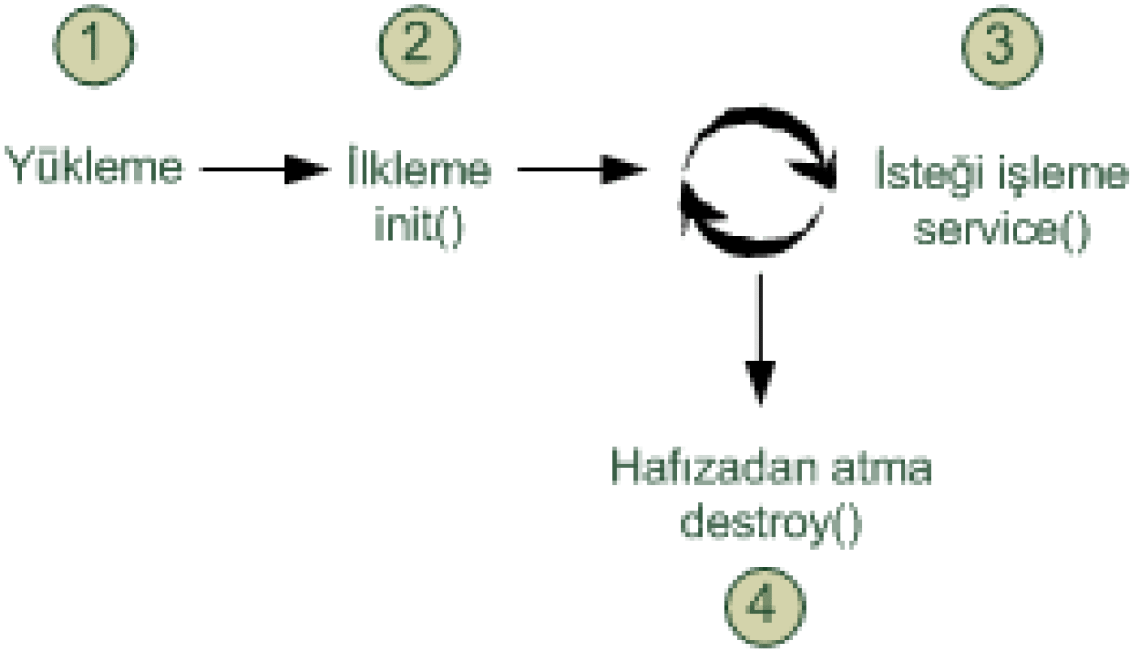
Servlet, önceden dediğimiz gibi Servlet arayüzünü gerçekleştiren bir sınıftır. Servlet'lerin çalıştığı uygulamaya Servlet makinesi (Servlet engine) denir. Servlet'lerin çalışabilmesi için Servlet makinesi bulunan bir web sunucusuna (OC4J, Apache Tomcat, Resin, JRun gibi…) ihtiyaç vardır.

Birden fazla kullanıcı aynı anda Servlet'leri çağırabilir. Böylece kullanıcı isteklerine daha hızlı cevap verilmiş olur. Servlet metotları çağrılırken çoklu iş parçacıkları kullanılır. Çok fazla istek olduğu durumda Servlet'in kaç tane nesnesinin kullanılacağına ve iyileştirilmesine web sunucusu karar verir.

1.6.1.1. Servlet Hayat Döngüsü

Servlet'in çalışabilmesi için ilk olarak Servlet sınıfı hafızaya yüklenir. İlkleme işlemlerini gerçekleştirebilmek için her bir Servlet nesnesi için bir kere olmak üzere init( )

metodu çağrılır. Tarayıcılardan istek geldiğinde ise çalışacak olan service( ) metodudur. Servlet'in nesnesi hafızadan atılmadan önce destroy( ) metodu çağrılır.

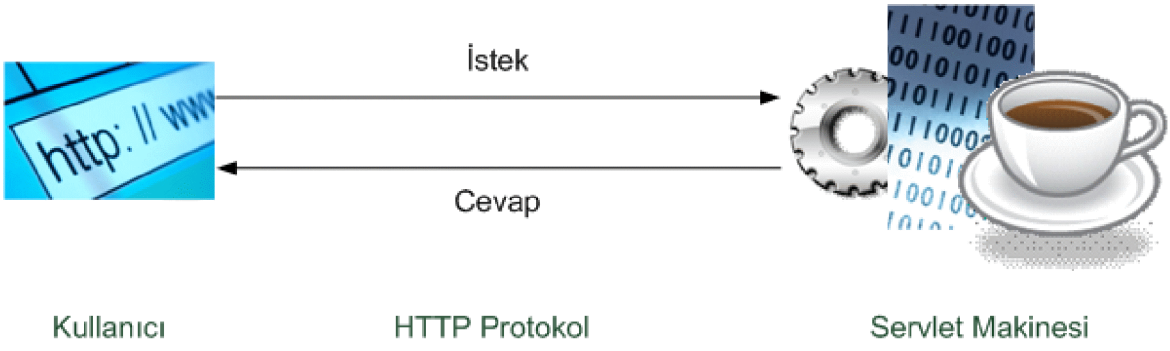


Şekil 1.5. Servlet hayat döngüsü

1.6.1.2. Hiper Metin Transferi Protokolü (HTTP) Servlet

Sun Microsystems firmasına ait Servlet Uygulama Programlama Arayüzünde (API) sadece HTTP protokolü için Servlet bulunur. Fakat Dosya Transfer Protokolü (FTP), Elektronik Posta Gönderme Protokolü (SMTP) gibi başka protokoller üzerinde çalışan Servlet'ler de oluşturulabilir.

HTTP Servlet'ler Servlet arayüzünü gerçekleştiren HttpServlet sınıfından türerler. Kullanıcı, Web tarayıcısı aracılığı ile istekte bulunur. Servlet'te bu isteğe uygun metodu çalıştırarak durum kodunu ve sonucu tarayıcıya geri gönderir.



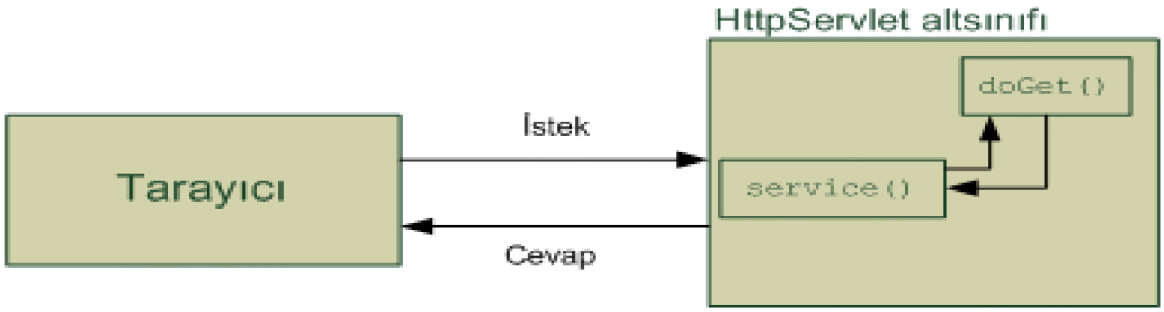
Şekil 1.6. HTTP Servlet çalışma şeması

HTTP protokolünde istekler çeşitli metotlar aracılığı ile gerçekleştirilir. HTTP 1.0 istek metotları GET, HEAD, POST olmak üzere üç tanedir. HTTP 1.1 ile PUT, OPTIONS, DELETE, TRACE ve CONNECT olmak üzere beş adet yeni metot daha eklenmiştir.

Yaygın olarak kullanılanları GET ve POST metotlarıdır. HTTP metotlarının Servlet'teki karşılıkları doXXX( ) şeklindedir. XXX yerine HTTP metodunun adı gelir. Genel yazımı da ilk harf büyük diğer harfleri küçüktür. Yani, HTTP GET isteğine karşılık verecek ise doGet( ), POST isteğine karşılık verecek isek doPost( ) metodunun üzerine yazmamız gerekir.

HTTP GET, ile sunucudan istekte bulunulur. Tarayıcıdan adres girilip istek iletildiğinde yada HTML etiketini kullanarak belirttiğimiz bağlantıda GET metodu ile istek gerçekleşir. GET metodunda kullanıcı bilgileri URL üzerinden gönderilir. Bu nedenle şifre gibi kritik bilgiler alınması gerektiğinde çok kullanışlı değildir. Aynı zamanda bilgiler URL üzerinden gönderildiği için bilginin boyutunda da bir kısıt vardır. Aşağıdaki URL ifadesinde soru işaretinden sonraki kısım sorgu ifadesi olarak adlandırılır. Her bir parametre & karakteri ile ayrılır.

/servlet/IlkServlet?ad=Celal&soyad=Atalar



Şekil 1.7. GET metodu kullanılarak yapılan isteğin yanıtlanma şeması

1.6.2. JSP

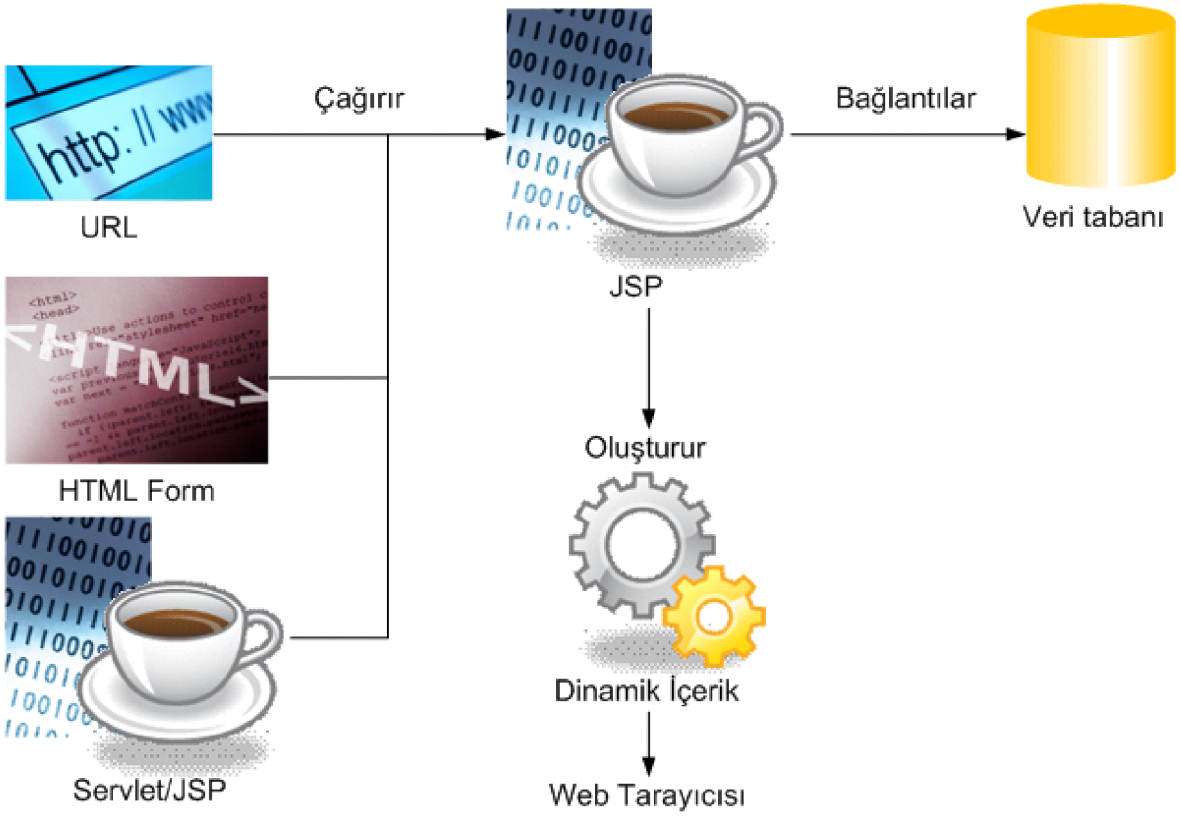
JSP dosyalara farklı bir biçimde yazılmış Servlet'ler diyebiliriz. JSP dosyalarda HTML içerikle dinamik içerik ayrı ayrı yazılır. HTML içeriği, dinamik içeriği değiştirmeden çok kolay bir şekilde değiştirebilirsiniz. Çok fazla programlama bilgisi gerektirmez ve daha çok web tasarımcıları tarafından kullanılırlar.

JSP, Servlet'lerin bir üstünde olan yapıdır ve her bir JSP aynı zamanda bir Servlet'tir. Servlet gibi tamamıyla Java'da yazılmaz. HTML gibi durağan içerikleri de dinamik koddan ayrı olarak içerir. Bu açıdan yazımı Servlet'lere göre daha kolaydır. Genel olarak bir kullanıcı arayüzünü geliştirmek için kullanılır. Web tasarımcılarından deneyimli programcılara kadar herkes tarafından rahatlıkla kullanılır.

JSP sayfaları URL, HTML form ve diğer Servlet/JSP sayfalardan çağrılır. JSP' e bağlantı sonucunda dinamik içerik oluşturulur ve web tarayıcısına aktarılır. JSP sayfaları, alt tarafta Servlet'te olduğu gibi bütün Java teknolojisinden yararlanır.

Bir JSP sayfasını doğrudan çalıştırabilmemiz için yazmamız gereken URL adresinin genel ifadesi aşağıdaki gibi olacaktır.

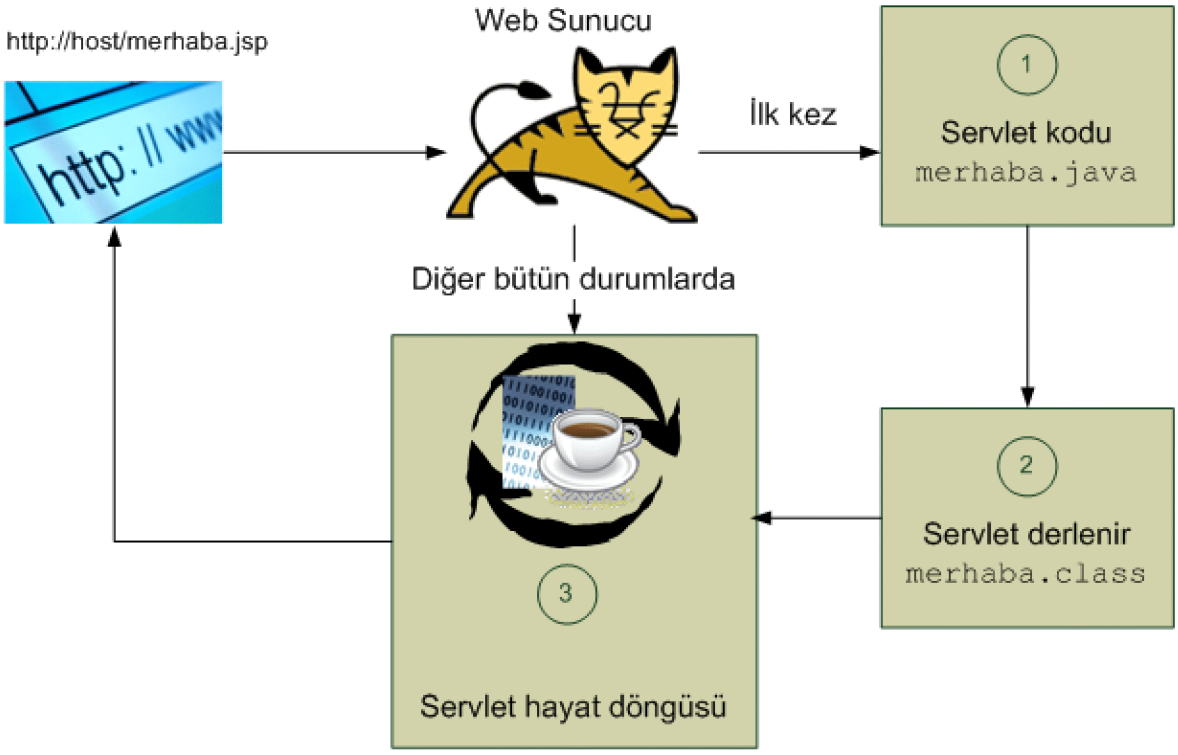
http://host:port/<context-root>/JSPSayfasının_Adı.jsp



Şekil 1.8. JSP sayfaların yapısı ve çalışması

1.6.2.1. JSP Hayat Döngüsü

Bir JSP sayfası ilk çağrılıyor ise önce Servlet'e dönüştürülür. Bunun için ilk olarak Servlet için kaynak kodu oluşturulur. Kaynak kodu tam zamanlı hata ayıklayıcı derleyiciler kullanılarak derlenir ve .class uzantılı bayt kodu oluşturulur. Artık JSP dosyamız web sunucusu için Servlet'tir ve Servlet gibi çalışır. Servlet'in çalışması sonucu içerik web sunucusu tarafından web tarayıcısına gönderilir.



Şekil 1.9. JSP hayat döngüsü

1.6.2.2. Temel JSP Elemanları

Her teknolojide olduğu gibi JSP teknolojisinde de kullanılan temel elemanlar vardır. Bu elemanlardan tanımlama elemanı dışındaki diğer elemanlar için oluşturulacak kod _jspService adlı metot içerisine konulur. _jspService, Servlet'teki servis metoduna karşılık gelir. _jspService metodu, kullanıcıdan gelen GET ve POST isteklerine cevap vermeyi sağlar.

1.6.2.2.1 İfade Elemanı

Bu eleman ile içinde yazılan nesnenin web tarayıcısında görüntülenmesini sağlar. Bu elemanın özellikleri şu şekilde sıralanabilir:

- <%= ifadesi ile başlar. % ve = bitişik olmalıdır aksi halde çalışmaz.
- Servlet dönüştürüldüğünde ifadeler için oluşturulacak kod _jspService metodu içine yazılır
- %> ifadesi ile sonlanır.

- Eşittir ifadesinden sonra bir nesne yada bir nesne döndüren metot gelmemelidir.
- Nesne ve metot yazımından sonra noktalı virgül kullanılmaz.

1.6.2.2.2 Dil Elemanı

JSP sayfası içinde Java kodlarımızı çalıştırmak için kullandığımız elemandır ve özellikleri şu şekilde sıralanabilir:

- <% ifadesi ile başlar.
- İçine konulan Java kodu _jspService metoduna eklendiği için her istek olduğunda çalıştırılır.
- %> ifadesi ile sonlanır.

1.6.2.2.3 Tanımlama Elemanı

Sınıf seviyesinde yeni bir nesne yada metot tanımlamak için kullanılır. Özellikleri aşağıdaki şekilde sıralanabilir:

- <%! İfadesi ile başlar.
- Java nesne ve metot tanımlamalarını içerir.
- %> ifadesi ile biter.

1.6.2.2.4 Direktif Elemanı

JSP sayfasının özelliklerini belirlemek, özel etiketler eklemek için kullanılır ve özellikleri şu şekilde sıralanabilir:

- <%@ ifadesi ile başlar.
- %> ifadesi ile sonlanır.
- Çalışmakta olduğunuz JSP sayfasına başka JSP sayfaları eklemenizi ve sayfanın özelliklerini değiştirmeyi sağlar.

1.7. Elektronik Kütüphane Uygulamaları

Bilgisayar sistemlerinin ve teknolojinin gelişmesiyle birlikte elektronik ortamda bulunan dokümanlar üzerinden araştırma yapabilme ve istenilen gerekli bilgilere ulaşılması

son derece kolaylaştırılmıştır. Bu elektronik kütüphane uygulamalarına aşağıdaki birkaç örnek verilebilir.

1.7.1. IEEE

IEEE Xplore, elektrik mühendisliği, bilgisayar bilimleri ve elektronik alanlarında yazılmış, dünyanın en kaliteli teknik dokümanlarına tam erişim sağlayan bir dijital kütüphanedir. IEEE Xplore; IEEE bültenleri, işlem hareketleri, dergiler, mektuplar, konferans kayıtları ve standartlar hakkında yazılı dokümanlar içerir. Ayrıca Mühendislik ve Teknoloji Enstitüsü (IET) yayınlarını da içerir [17].

1.7.2. Ebrary

On yıldan uzun bir süredir Ebrary; kütüphaneler, şirketler, hükümetler, yayıncılar, sivil toplum kuruluşları ve birlikler, toplayıcılar ve dağıtımcılar, bireyler ve diğer benzer kurumların dijital içerik ihtiyaçlarının adreslenmesi konusunda onlarla birlikte çalışmaktadır [18]. Misyonu, son kullanıcılara en güçlü araçlarla bilgi sağlama, keşfetme ve yönetme olanağı sağlarken, müşterilerimizin önemli bilgileri elde etmesinin ve dağıtmasının mümkün olan en düşük maliyette ve en yüksek etkinlikte sağlanmasıdır.

İçeriğinde bulunan kılavuzların, e-kitapların, kitapçıkların, raporların ve diğer telif haklarına sahip dokümanların esnek üyelik ve satın alma koşullarına ek olarak, organizasyonların ihtiyaçları olan dijital materyalleri dağıtması ve kendi ihtiyaçlarına göre entegre etmesi konusunda organizasyonlara kendi teknolojisini kullanmaları konusunda yetki de vermektedir.

Ebrary, halihazırda 2700 müşteriye ve dünya çapında 17 milyon son kullanıcıya hizmet vermektedir. 350'den fazla dünya lideri yayıncıyla birlikte ortaklıkta bulunan Ebrary, büyüyen uluslararası ağında aracılar, yayınevleri ve diğer ortaklarıyla birlikte ürün ve servislerini dünya çapında dağıtmaktadır.

1.7.3. ScienceDirect

ScienceDirect, 2500'den fazla bilirkişinin incelediği dergilerden ve 11000'den fazla kitaptan makaleler ve bölümler içeren tamamıyla yazılı dokümanlardan oluşan sektör

lideri bir veritabanıdır [19]. Halihazırda 9,5 milyon başlık ve makale bulunmaktadır ve her yıl bu sayıya yarım milyon daha eklenmektedir.

Platform, kullanıcıların bilgiyi elde etme süreci verimliliğini çoğaltan, bilgi edimini işlevselleştiren sofistike arama olanağı sunar. Yeni araçlar, araştırma iş akışını kolaylaştırır ve erken dönemlerde yayınlanmış olan içeriğe çabuk erişimi sağlar; depolanabilir, basılabilir ve dağıtılabilir içeriğin elverişli bir biçimde çoklu indirilmesini sağlar.

Web arayüzü, bilginin anlambilimsel teknoloji kullanılarak (örneğin NextBio) diğer içeriklerle zenginleştirilip sunulması konusunda yeni yöntemler sunar. Ayrıca 2003'ten bu yana birçok yazar, sisteme ekstra katma değer yaratan ses ve görüntü dosyaları, içeriği destekleyici diğer materyaller yüklemektedir ve bu şekilde araştırma sonuçları salt baskı formatının ötesine geçmektedir.

1.7.4. Elsevier

ScienceDirect, Elsevier'in bir parçasıdır. Elsevier, dünyanın en büyük bilimsel, teknik ve medikal bilgi sağlayıcısıdır, 2000'den fazla kitap, bülten ve ikincil veritabanı yayınlamaktadır.

Elsevier, Reed Elsevier Plc Group şirketinin bir üyesidir. Grup, bilgi sağlayıcı ve yayınlayıcılar sektöründe dünya lideridir. Elektronik ortamda bilimsel bilgi alış-verişi, alım-satım sektöründe faaliyet göstermektedir. Reed Elsevier, profesyonel kullanıcılara internet yoluyla yüksek kaliteli ve esnek bilişim çözümleri sunmaktadır.

2. YAPILAN ÇALIŞMALAR, BULGULAR VE TARTIŞMA

2.1. Giriş

Çalışmada kullanıcıların, bir sunucu üzerinde bulunan elektronik kitaplar içerisinde arama yapmasını ve yapılan aramanın sonucunda bulunan kayıtların kullanıcılara bir web tarayıcı üzerinde gösterilmesini sağlamak amacıyla bir elektronik kütüphane sistemi geliştirilmiştir. Bu işlevleri sağlaması amacıyla çeşitli kütüphaneler kullanılarak PDF içerisinde arama yapabilen, PDF oluşturabilen, PDF içeriği değiştirebilen ve yapılan bir veritabanı sorgusunu PDF olarak temsil edebilen ve kaydedebilen bir çatı geliştirilmiştir. Çatı, belirli işlevleri yerine getirmek amacıyla bünyesinde yapılar ve fonksiyonlar barındıran kod topluluğudur. Geliştirilen çatı ile sistemdeki üç ana işlevi yerine getirmek mümkündür:

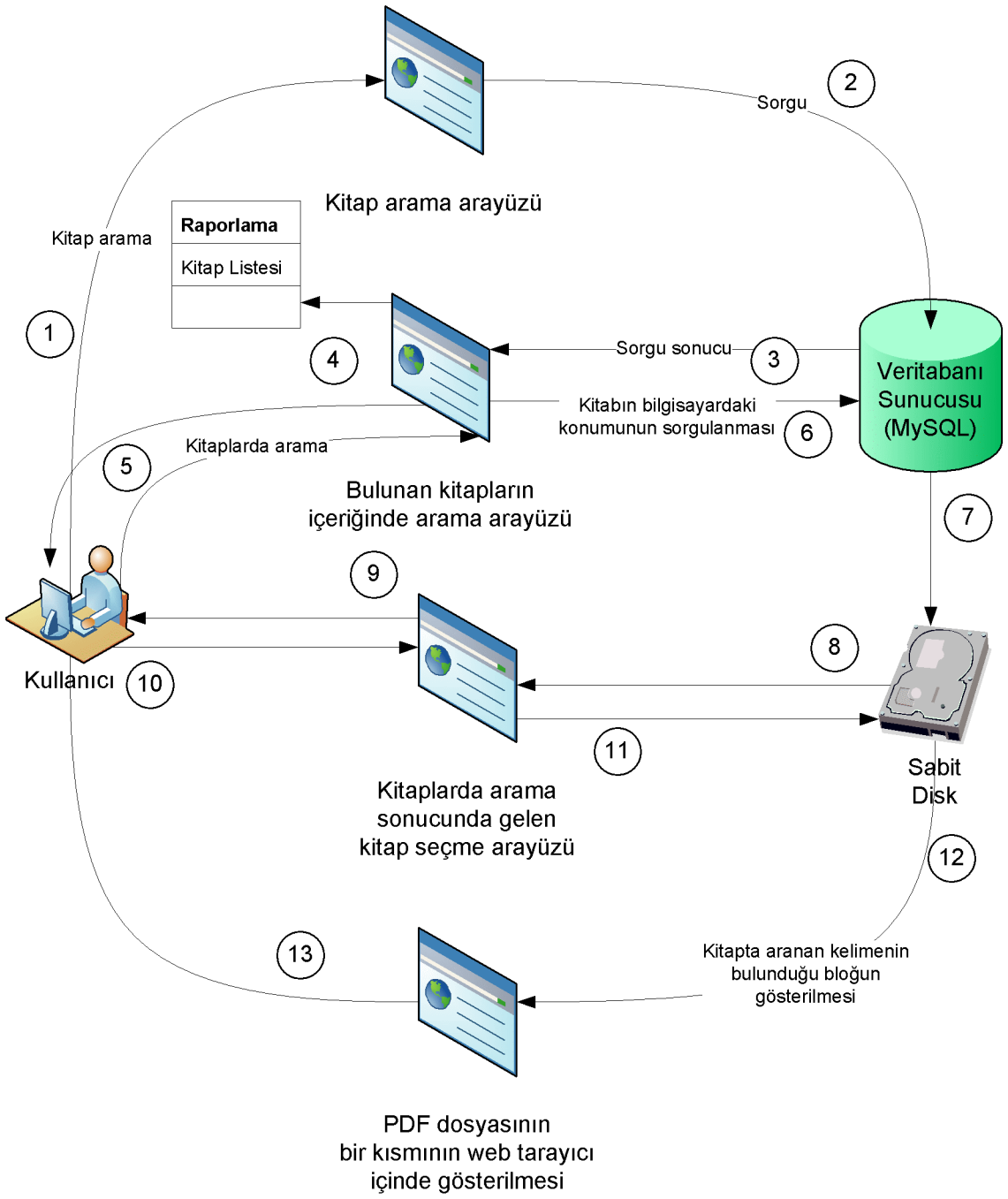
- PDF içerisinde arama yapmak
- Bir veritabanı sorgusunun sonucunu PDF olarak temsil etmek
- PDF dokümanının belirli bir kısmının gösterilmesi

PDF içerisinde iki farklı arama yapılabilmektedir. Bunlardan biri PDF içeriğinde arama yapar. Diğeri ise PDF yapısında bulunan yer imleri (bookmark) üzerinde arama yapar. Yer imlerinde arama yapmak, normal aramaya göre hızlıdır fakat yapılan arama daha kısıtlı bir sonuç kümesi verir.

Elektronik kütüphane içinde bulunan kitaplarda arama işlemini gerçekleştiren etkileşimli sistemin çalışmasının blok şeması Şekil 2.1.' de verilmiştir.

Şekil 2.1.'deki şemada kullanıcı ilk olarak kitap arama arayüzünü kullanarak istediği kategorideki kitapları arar. Kitap arama işlemi veritabanında kayıtlı olan kitap adlarına göredir. Arama sonucunda bulunan kitapların listesi, geliştirilen çatı içerisindeki raporlama fonksiyonu kullanılarak PDF'ye dönüştürülür. Aynı zamanda kullanıcının arama yapmak istediği kitapları seçmesi için kitap listesi web sayfasında da listelenir. Kullanıcı web sayfasında, içinde arama yapmak istediği kitapları seçer ve aramak istediği kelimeyi belirterek arama işlemini başlatır.

Elektronik kitaplar sistemin bulunduğu sunucuda, kullanıcının herhangi bir internet bağlantısı ile erişemeyeceği bir dizinde bulunur. Buradaki amaç kullanıcıların doğrudan internet bağlantısını kullanarak kitapların tümüne erişmesini engellemektir.



Şekil 2.1. PDF'de arama yapan sistemin blok şeması

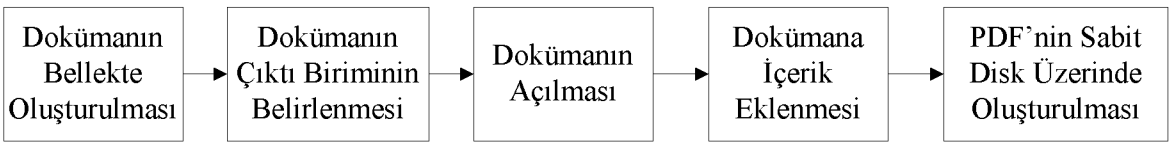
Kullanıcı arama işlemini başlattıktan sonra veritabanındaki kayıtlara göre kitapların sabit disk üzerindeki konumları bulunur ve kitaplar içinde arama işlemi gerçekleştirilir. Kullanıcının yaptığı arama sonucunda eğer aranan ifade bulunduysa, bulunan ifadeyi

içeren kitaplar web sayfasında listelenir. Kullanıcı listelenen kitaplar arasından istediklerini seçerek ifadenin kitap içinde bulunduğu bloğu görüntüleyebilir. Kitabın tamamını göstermek yerine sadece aranan ifadenin bulunduğu küçük bir kısmını (blok) görüntülemek, tüm kitabın izinsiz kopyalanmasını engellemek amacıyla yapılmıştır.

Çalışmada belirli işlevleri yerine getirmek amacıyla çeşitli kütüphanelerden yararlanılmıştır. Yapı ve fonksiyonların bulunduğu çatı Java dilinde oluşturulmuştur. Kullanıcı ile etkileşimi sağlayan web arayüzü JSP ile oluşturulmuştur. JSP kodları içerisinden çatı fonksiyonlarına çağrılar bulunmaktadır. Kitaplar ile ilgili verilerin saklanması için MySQL veritabanı yönetim sistemi kullanılmıştır. PDL yapısındaki PDF'lerden ham metni elde etmek amacıyla PDFBox kütüphanesi kullanılmıştır [20]. Doküman üretilmesi, raporlama, PDF'nin web tarayıcıda görüntülenmesi vb. işlevleri sağlamak amacıyla da iText kütüphanesi kullanılmıştır [21].

2.2. PDF Dokümanın Üretilmesi

PDF dokümanının üretilmesinde iText kütüphanesinden yararlanılmıştır. Dokümanın üretilmesi işlemi temel olarak beş kısımda incelenebilir.



Şekil 2.2. PDF dokümanın üretilmesi

Şekil 2.2'deki işlevi gerçekleştiren "Merhaba Dunya" içeriğine sahip PDF dosyasını oluşturan kod bloğu aşağıda verilmiştir.

```
Document dokuman=new Document();   //  Dokümanın Oluşturulması
PdfWriter yazici;
yazici= PdfWriter.getInstance (dokuman, new FileOuputStream("C:\\Selam.Pdf");
// Dokümanın Çıktı Birimini Belirlenmesi
dokuman.open();  //Dokümanın açılması
dokuman.add(new Paragraph("Merhaba Dunya"); // Dokümana İçerik Eklenmesi
dokuman.close();
```

PDF dokümanının üretilmesi sırasında kullanılan sınıflar ve bu sınıflara ait fonksiyonlar iText kütüphanesi içinde tanımlıdırlar.

Document (); içerik ve dokümana ait üst veri bilgilerini (metadata) içeren bir nesnedir. PdfWriter sınıfı, üretilecek dokümanın PDF formatında olduğunu gösterir ve PDF dokümana ait tüm içerik bilgilerini belirler. PDF haricindeki doküman formatlarını oluşturan yazıcı sınıfları kendilerine ait alt başlıkta açıklanacaktır. Document ve PdfWriter sınıfları iText kütüphanesi içinde tanımlıdır.

2.2.1. Dokümanın Oluşturulması

Document nesnesini oluştururken birinci sayfanın boyutlarını, arka plan rengini ve kenar boşluklarını belirleyebilirsiniz. Özel ölçülerle belirlenmiş sayfa boyutları Rectangle nesnesi yardımıyla tanımlanabilmektedir.

```
Rectangle sayfa_boyutlari = new Rectangle (216f, 720f);
Document dokuman = new Document( sayfa_boyutlari );
```

Rectangle nesnesi iText kütüphanesi içinde tanımlıdır. Bu nesnenin birinci parametresi sayfanın genişliğini , ikinci parametresi ise sayfanın yüksekliğini belirler. Kolaylık sağlaması açısından A0-A10, B0-B5 , LEGAL, LETTER, HALFLETTER, _11*17, LEDGER, NOTE, ARCH_A-ARCH_E, FLSA ve FLSE gibi standart sayfa boyutları tanımlanmıştır. Bu standartlar iText kütüphanesi içindeki PageSize sınıfı içinde tanımlıdırlar. Fonksiyonlar yardımıyla sayfa yerleşiminin nasıl olacağı (yatay veya düşey) belirlenebilmektedir. Sayfa boyutları belirtilmemiş doküman varsayılan olarak A4 kağıdı ebatlarında oluşturulur.

Doküman sayfalarının arka plan rengi java.awt.Color nesnesi içindeki Color fonksiyonu yardımıyla değiştirilebilir. Color fonksiyonu RGB renk uzayını kullanmaktadır, farklı renk uzayları için (CMYK gibi) iText kütüphanesi içindeki sınıflar kullanılabilir.

Dokümana içerik eklenmeden önce birinci sayfanın kenar boşluklarının belirlenmesi gerekmektedir. Aşağıdaki kod birinci sayfası A5 kağıdı boyutlarında 36 pt (1,27 cm) sol kenar , 72 pt (2,54 cm) sağ kenar, 108 pt (3.81 cm) üst kenar, 180 pt (6.35 cm) alt kenar boşluğuna sahip bir PDF dokümanı oluşturur.

Document document = new Document(PageSize.A5, 36, 72, 108, 180);

2.2.2. Dokümanın Çıktı Biriminin Belirlenmesi

Dokümanın çıktı birimi, disk üzerinde bir PDF dokümanı olabileceği gibi aynı içeriğe sahip bir zengin metin belgesi (.rtf) , Java Servlet çıktı akışı, HTML formatında bir web sayfası olabilir;

```
PdfWriter.getInstance(dokuman,
new FileOutputStream("Merhaba.pdf")); // PDF formatında doküman
RtfWriter2.getInstance(dokuman,
new FileOutputStream("Merhaba.rtf")); // zengin metin belgesi formatında doküman
HtmlWriter.getInstance(dokuman,
new FileOutputStream("Merhaba.htm")); // web sayfası formatında doküman
```

Web tabanlı uygulamalarda PDF dokümanı bir web tarayıcısına javax.servlet.ServletOutputStream sınıfı ile gönderilebilir.

RtfWriter2, HtmlWriter sınıfları iText kütüphanesi içinde tanımlıdırlar. PDF dokümanlarının üretilmesine benzer şekilde, RtfWriter2 sınıfı kullanılarak Microsoft Word dosyaları, HtmlWriter sınıfı ise html formatında web sayfaları üretilebilir.

2.2.3. Dokümanın Açılması

Parametresiz olarak oluşturulan dokümanlarda birinci sayfanın boyutları, arka plan rengi ve kenar boşlukları gibi özelliklerin doküman açılmadan önce belirlenmesi gerekir;

```
Document document = new Document(PageSize.A5, 36, 72, 108, 180);
PdfWriter.getInstance(dokuman, new
FileOutputStream("Merhaba.pdf"));
dokuman.open();
```

Özellikleri belirtilmemiş dokümanların ilk sayfaları varsayılan ayarlarda açılır. Bu şekildeki dokümanlarda, programcı ancak ikinci sayfadan itibaren sayfa özelliklerini belirleyebilir. Sayfa düzenlemelerinin bütün dokümana uygulanmamasının sebebi ise programcının her sayfayı farklı özelliklerle (sayfa boyutları, sayfa yerleşimi vs.) donatabilmesine izin vermektir.

Birçok doküman sürüm bilgilerini ve dokümana ait genel bilgileri dosya başlığında bulundurur. Bu nedenle doküman açılmadan önce sürüm bilgisinin ve üst bilgilerin belirtilmesi gerekir.

Doküman açıldığında OutputStream'e aşağıdaki kodlar yazılır.

```
%PDF-1.4
%âãÏÓ
```

İlk satırdan anlaşılacağı üzere oluşturulan PDF dokümanının sürümü belirtilmiştir. İkinci satır biraz daha farklı görünmektedir ve yüzde sembolü ile başlamaktadır. Bu bir PDF yorum satırıdır ve herhangi bir fonksiyon içermez. Bu satırın eklenmesi gerekli değildir, fakat dosya transferi sırasında dosya içeriğinin (metin yada ikili dosya) belirlenmesi açısından yazılması tavsiye edilir. PDF dokümanları ikili dosya yapısındadır. Sürümü belirlenmeden açılan bir PDF dokümanı varsayılan olarak 1.4 sürümünde oluşturulur. Doküman açılmadan önce PDF dokümanının sürümü belirlenmelidir;

```
Document dokuman = new Document();
PdfWriter writer = PdfWriter.getInstance(dokuman,
new FileOutputStream("Versiyon_1_6.pdf"));
writer.setPdfVersion(PdfWriter.VERSION_1_6);
// PDF-1.6 versiyonunda PDF doküman    iText tanımlı  fonk.
dokuman.open();
```

PDF doküman sürümleri gelişimlerine göre farklı özellikler içerirler; PDF-1.0 sürümü metin ve grafiklerden oluşan bir yapıyı ekrana yansıtma ve yazıcıdan çıktı alabilme yeteneğine sahiptir. PDF-1.1 sürümünde önceki sürüme ek olarak parola korumalı PDF dokümanı oluşturabilme, dış bağlantı (web sayfası adresi) ekleyebilme ve dış cihazlardan bağımsız renk ayarı yapabilme özellikleri eklenmiştir. PDF-1.2 (Zip/gzip) sıkıştırma

desteği, etkileşimli form desteği ve Çince, Japonca ve Korece dil desteğini sağlamıştır [22]. PDF-1.3 sürümü ile birlikte, PDF dokümana dosya eklentisi yapabilme, dokümanlara dijital imza yerleştirebilme ve sayfa numaralandırma özellikleri getirilmiştir. PDF-1.4 sürümünde 128 bit ile doküman şifreleyebilme, şeffaf sayfalar yapabilme ve işaretli PDF dokümanı yapılabilmektedir. PDF-1.5 sürümü şifreleme ve sıkıştırmaya ek yeni özellikler, isteğe bağlı içerik grupları oluşturulabilmesini ve çoklu ortam dosyaları ekleyebilme desteğini sağlamaktadır. PDF-1.6 sürümü, AES desteği ve yazdırma işlemi için sayfa ayarı yapabilme özelliklerini sunmaktadır.

PDF dokümanına ait üst veri bilgisi bilgi katalogunda tutulur. Üst veri bilgisi, PDF dokümanı açılmadan önce belirlenmelidir. PDF dokümana üst veri eklenmesi ;

```
dokuman.addTitle("Merhaba");     // Doküman Başlığının Belirlenmesi.
dokuman.addSubject("Metadata Eklenmesi Örneği"); //Konu Eklenmesi
dokuman.addKeywords("Metadata, Üst Veri");   //Anahtar sözcüklerin Eklenmesi
dokuman.addCreator("iText ile üretilmiştir."); //Dokümanın Üretildiği Ortamın
dokuman.addAuthor("Celal Atalar");     // Yazarın Eklenmesi
dokuman open();        // Dokümanın açılması
```

Bu kısımda belirtilmemesine rağmen PDF dokümanı üreten program veya kütüphanenin ismi bile değiştirilebilir. Fakat bu etik açıdan doğru değildir.

Böylelikle PDF dokümanın üst veri bilgisi tamamlanıp doküman açılmıştır. Şimdi dokümana içerik ekleyebiliriz.

2.2.4. Dokümana İçerik Eklenmesi

Bir PDF dokümanına içerik üç farklı yöntemle eklenebilir. En kolay içerik ekleme yöntemi, iText kütüphanesinin basit bloklarını kullanmaktır. Dokümana içerik eklenmesi için iText kütüphanesinde tanımlı Paragraph() fonksiyonunu kullanabilirsiniz. Bu fonksiyon sayesinde içeriğin dosya üzerinde hangi konumdan itibaren başlayacağı veya kenar boşlukları gibi parametrelerin belirtilmesine gerek duyulmaz. Ayrıca iText kütüphanesi içinde bulunan font , renk , sayfanın sınır genişlikleri , girintili yazma vs. gibi sınıflar sayesinde dokümana içerik eklenmesi diğer yöntemlere göre oldukça basittir [23].

Diğer bir yöntem, eğer bir PDF uzmanı iseniz, PDF operatörlerine karşılık gelen iText fonksiyonlarını kullanmaktır. Örnek olarak aşağıda "Merhaba Dunya" içeriğine sahip bir PDF dosyasının içerik akışı (content stream) gösterilmiştir.

```
<</Length 55>>stream
q
BT
36 806 Td
0 -18 Td
/F1 12 Tf
(Merhaba Dunya)Tj
ET
Q
Endstream
```

Bu içeriğe sahip PDF dokümanı aşağıdaki Java kodu ile oluşturulabilir;

```
PdfContentByte cb = writer.getDirectContent();
BaseFont bf = BaseFont.createFont(
BaseFont.HELVETICA, BaseFont.CP1252, BaseFont.NOT_EMBEDDED);
cb.saveState(); // q
cb.beginText(); // BT
cb.moveText(36, 806); // 36 806 Td
cb.moveText(0, -18); // 0 -18 Td
cb.setFontAndSize(bf, 12); // /F1 12 Tf
cb.showText("Merhaba Dunya"); // (Merhaba Dunya)Tj
cb.endText(); // ET
cb.restoreState(); // Q
```

moveText fonksiyonu ile imleç doküman içindeki uygun yere getirilir. Font ve özellikleri belirtildikten sonra dokümana "Merhaba Dunya" içeriği girilir.

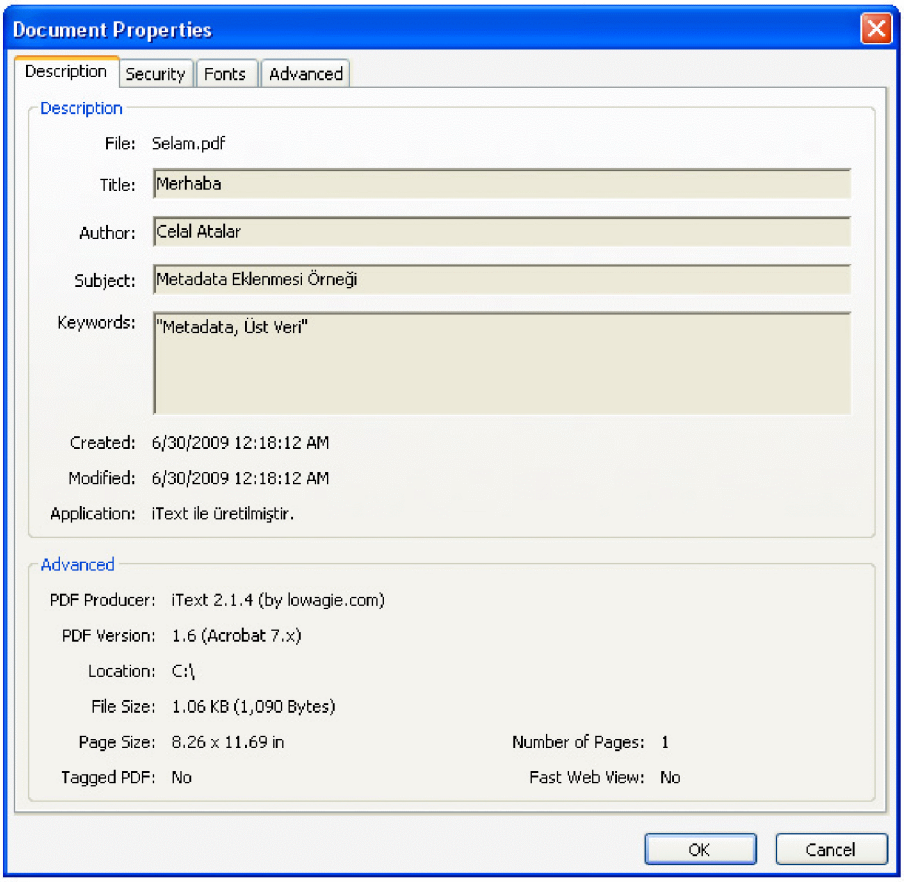
Sonuncu yöntem ise; eğer Java programlama dilinde uzman iseniz, iText sınıflarını ve java.awt sınıfı içindeki Graphics2D fonksiyonlarını kullanmaktır. Bu yöntemi kullanarak aşağıdaki kod ile "Merhaba Dunya" dokümanı üretilebilir.

```
PdfContentByte cb = writer.getDirectContent();
Graphics2D graphics2D =cb.createGraphics(PageSize.A4.width(),
PageSize.A4.height());
graphics2D.drawString("Merhaba Dunya", 36, 54);
graphics2D.dispose();
```

2.2.5. Dokümanın Kapatılması

Bu aşamaya kadar yapılan tüm işlemler bellekte saklanmaktadır. PDF dosyasının herhangi bir sabit disk üzerinde oluşturulabilmesi için dosyanın kapatılması zorunludur.

Merhaba Dunya



Şekil 2.3. Oluşturulan Selam.pdf dokümanına ait içerik ve özellikleri

2.3. PDF Doküman Üzerinde Bir Metinin Aranması

PDF dokümanlarda arama işlemi; dokümanın içeriği üzerinde yapılan arama işlemi ve dokümanının sayfa imleri üzerinde yapılan arama işlemi olmak üzere iki kısma ayrılabilir.

PDF dokümanın içeriği üzerinde yapılan arama işlemi (gelişmiş arama) aranılan bilgiye ulaşma konusunda sayfa imleri üzerinde yapılan arama işlemine göre daha isabetli olmaktadır. Fakat PDF dokümanın içeriğinin metine çevrilmesi işlemi çok fazla süre gerektirdiğinden bu arama işlemi hız bakımından bir dezavantaj içermektedir.

PDF dokümanın sayfa imleri üzerinde yapılan arama işleminde (hızlı arama) elde edilebilecek sonuçlar içerik üzerinde yapılan arama işleminde olduğu kadar isabetli değildir. Fakat bu arama işleminde metine çevrilecek olan sadece sayfa imleri olduğundan hız bakımından büyük bir avantaj sağlamaktadır.

PDF dokümanlarının ikili dosya oldukları daha önceki kısımlarda belirtilmiştir. Dolayısıyla doküman içeriğindeki veriler direk okunabilir değildirler. PDF dokümanından içeriğinin metine dönüştürülmesi işlemi PDFBox kütüphanesinin yardımıyla sağlanır [24].

Gelişmiş arama işlemi sırasında kullanılan PDDocument ve PDFTextStripper sınıfları PDFBox kütüphanesi içinde tanımlıdırlar. PDDocument nesnesi açılan PDF dokümanına bir işaretçi oluşturur, PDFTextStripper sınıfı ise PDF içeriğinin metine dönüştürülme işlemini sağlayan sınıftır. Gelişmiş aramada PDF dokümanın içeriği metine çevrildikten sonra arama işlemi onlu sayfa grupları üzerinde yapılmaktadır. Arama işlemi sırasında büyük-küçük harf kullanımı sebebiyle ortaya çıkabilecek sorunları önlemek amacıyla, düz metin ve aranan metin tamamen büyük harfe çevrildikten sonra arama işlemi yapılır. Arama işlemi sonucunda; eğer aranan metin bulunmuş ise, sonuç olarak metinin bulunduğu onlu bloğun ilk sayfa numarası geriye döndürülür.

Hızlı arama işleminde PDF dokümanına ait sayfa imlerinin elde edilmesi sırasında iText kütüphanesinden yararlanılır. Sayfa imleri bir ağaç yapısında olup, bütün düğümler açıldıktan sonra arama işlemi gerçekleştirilir. Bu arama işleminde de büyük-küçük harf kullanımından kaynaklanacak problemleri ortadan kaldırmak amacıyla aranan metin ve sayfa imleri tamamen büyük harfe çevrildikten sonra arama işlemine geçilir. Eğer aranan metin sayfa imleri içerisinde bulunmuşsa, sonuç olarak metinin bulunduğu sayfa iminin işaret ettiği sayfa numarası geriye döndürülür.

2.4. Raporlama

Raporlama; üretilen veya oluşturulan verilerin arşivlenmesi açısından oldukça önemlidir. Ayrıca raporlama ile üretilen veriler, çeşitli yöntemlerle kullanıcılarla paylaşılabilir.

Günümüzde rahatlık, kolaylık ve istenilen herhangi bir bilgiye hızlı bir şekilde erişim yöntemlerinden biri de bilgisayar teknolojisini kullanmaktır. Oluşturulan raporların web üzerinden kullanıcılara ulaştırılması bu yöntemlerin başında gelebilir. Fakat kullanıcıların çok çeşitli işletim sistemi ve platformlar kullanabilecekleri unutulmamalıdır. Bu sebeple, paylaşılacak belgenin her platformda görüntülenebilir olması gerekir. Her platformda görüntülenebilen dosya formatı PDF, raporlama için kullanılabilir.

Bilgisayar teknolojisinde raporlama kavramının en çok kullanıldığı alanlardan biri de veritabanı tabloları üzerinden sorgu sonucu elde edilen verilerdir. Herhangi bir tablo üzerinde yapılan sorgu sonucu elde edilen veriler PDF formatında üretilip kullanıcılara kolaylıkla dağıtılabilir.

Geliştirilen çatıda; bir tablo üzerinden elde edilen sorgu sonucunu (ResultSet) parametre olarak alıp verileri PDF formatında raporlayan bir fonksiyon oluşturulmuştur. Böylece çatıyı kullanacak programcının raporlama işlemleri için PDF formatının karmaşık yapısını bilmesine gerek yoktur. Sorgu sonucunun geliştirilen fonksiyona parametre olarak verilmesi yeterli olacaktır.

Geliştirilen fonksiyon öncelikli olarak sorgu sonucundan sütün isimlerini elde eder. Sütun isimleri yardımıyla sorgulanan tablo üzerinde birinci kayıttan başlanarak son kayıta kadar bütün veriler elde edilir. Örneğin; kullandığınız bilgisayar üzerinde MySQL veritabanı yönetim sistemi altındaki "Deneme" veritabanı içerisinde bulunan "Musteriler" isimli bir tabloya ait yapı ve içerisinde bulunan veriler Şekil 2.4.'deki gibi olsun.

tckimlikno	adi	soyadi	sabittelefonu	ceptelefonu
58567442078	CELAL	ATALAR	04623254927	05056372551
58567442079	NAZLI	SERT	04623254927	05056372636
58567442076	AYGÜL	SÖNMEZ	04623254927	05057896734
56587378131	JALE	YURTSEVEN	04628373711	05051313132
47498319238	ÇINAR	ÖZER	05041313123	04621331312
58567442071	GÖNÜL	ATALAR	05386748912	04623257893

Şekil 2.4. Musteriler tablosunun yapısı ve verileri

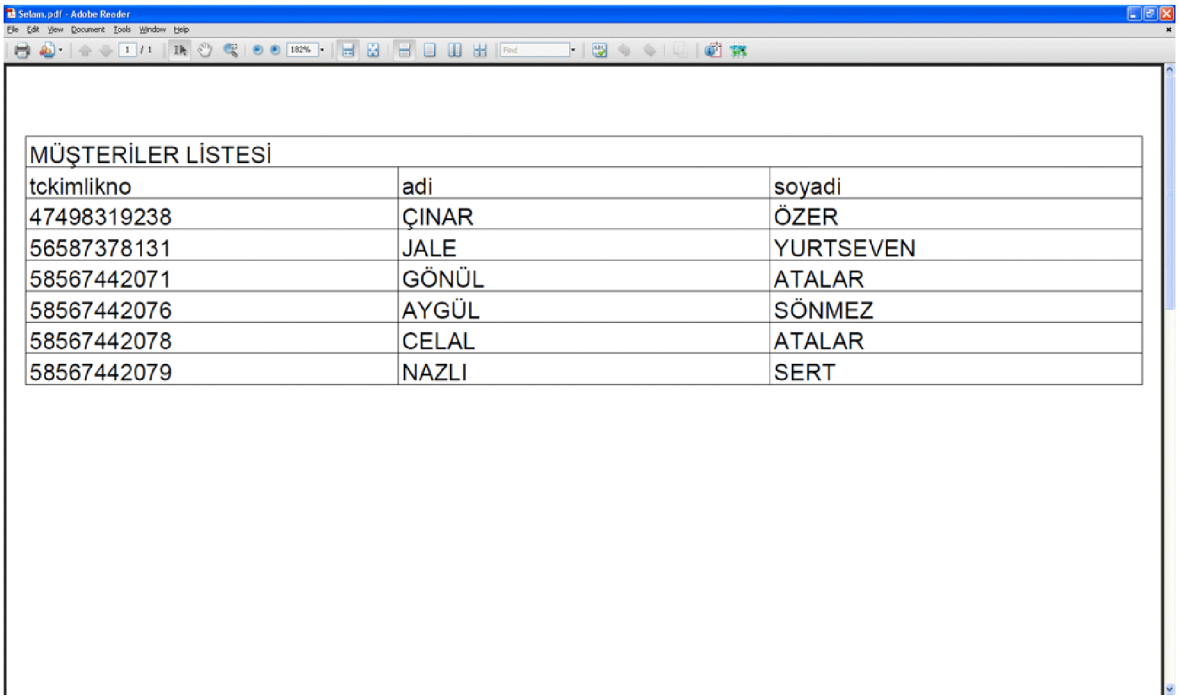
Musteriler tablosu üzerinde;

```
Connection conTest = DriverManager.getConnection("jdbc:mysql://localhost/Deneme");
Statement komut= conTest.createStatement();
ResultSet rs = komut.executeQuery("SELECT tckimlikno,adi,soyadi FROM Musteriler
Order By tckimlikno");
```

şeklinde bir sorgu cümlesi çalıştırıldığında, gösterilecek olan sütun sayısının üç adet olduğu açıktır. ResultSet sınıfı tipindeki rs değişkeni içerisinde üç sütunun ismi ve bu üç sütunun sorguya göre içereceği veriler bulunmaktadır.

Sorgu sonucunda oluşan kayıtlar üzerinde sırasıyla sütun isimlerine göre veriler elde edilerek iText kütüphanesi içinde tanımlı PdfPTable ve PdfPCell sınıfları yardımıyla PDF formatında raporlama üretilir. PdfPTable nesnesi bütün raporu içeren yapıya, PdfPCell sınıfı ise bir kayıt için herhangi bir sütun değerine işaret eder.

Ayrıca kullanıcıdan raporlama için bir rapor başlığı da parametre olarak alınır.



MÜŞTERİLER LİSTESİ		
tckimlikno	adi	soyadi
47498319238	ÇINAR	ÖZER
56587378131	JALE	YURTSEVEN
58567442071	GÖNÜL	ATALAR
58567442076	AYGÜL	SÖNMEZ
58567442078	CELAL	ATALAR
58567442079	NAZLI	SERT

Şekil 2.5. Musteriler tablosu üzerinde SQL sorgusu sonucu oluşturulan PDF raporu

Oluşturulan PDF raporunda sütunlara ayrılan alan mesafesinin aynı olduğuna dikkat edilmelidir. Bu sayede üretilecek raporda verilerin yerleşimi konusunda problem olmayacaktır. Sorgu içerisinde bulunan sütun sayısına uygun şekilde PDF dokümanının genişliği eşit parçalara bölünür. Ayrıca rapor işlemleri için PDF belgesinin genişliğinin boyutu ayarlanabilmektedir. Böylece çok fazla sütunun sıkıştırılmadan gösterilebilmesi sağlanır.

Raporlama kısmında problem oluşturabilecek noktalardan birisi de dil problemidir. Raporda Türkçe karakterlerin gösterilebilmesi için yazı tipi tanımlaması yapılması gerekmektedir [25]. Geliştirilen fonksiyonda;

```
BaseFont bf;
Font font;
bf = BaseFont.createFont("c:/windows/fonts/ARIAL.TTF", "ISO-8859-9" ,
BaseFont.EMBEDDED);
font = new Font(bf, 12);
```

şeklinde bir kod bloğu yardımıyla Türkçe karakterlerin PDF dokümanı içinde görüntülenebilmesi sağlanır.

2.5. Kitap Ekleme

Sisteme yeni kitapların eklenmesi için yapılmış arayüzdür. Bu kısımda sistem yöneticisi kitabın ismini, kategorisi ve bir gözat yapısıyla PDF dokümanını seçerek kitabı sisteme yükler. Eğer eklenecek kitabın kategorisi veritabanında bulunan kategorilerden biri değilse kategori kısmını kendi yazarak işlemi tamamlar. Bu sayede sisteme dışardan kitap eklenirken otomatik olarak farklı kategoriler de veritabanına dahil edilmiş olmaktadır.

2.6. Veritabanındaki Yapı

Veritabanı yönetim sistemi olarak MySQL kullanıldığını daha önce belirtmiştik. Sistem altında “pdf” isimli tek bir veritabanı bulunmaktadır. Bu veritabanı içinde de “pdf” isimli dört sütunu sahip tek bir tablo bulunmaktadır. Birinci sütun kitap numarası tutmaktadır. Bu sütun pozitif tamsayı, benzersiz ve otomatik artan özelliklerine sahiptir.

İkinci sütun PDF dokümanının ismini içerir ve düz metin yapısındadır. Üçünü sütun kitabın ait olduğu kategoriyi içerir ve düz metin yapısındadır. Dördüncü sütun ise kitapların sunucu üzerinde bulunduğu konumu içermektedir. Dosyalar sunucu üzerinde "D:\pdf" klasörü içerisinde bulunmaktadır.

Kategoriler için ayrı bir sütun veya ayrı bir tablo bulunmamaktadır. "pdf" tablosu üzerinde kategori sütunu üzerinde;

SELECT DISTINCT(kategori) FROM pdf Order By kategori;

sorgu cümlesi çalıştırıldığında tablo üzerinde bulunan kategori isimleri listelenecektir. DISTINCT özel kelimesi ile aynı kategori isimlerinin bir defadan fazla listelenmesi önlenmiştir.

2.7. PDF Dokümanının Belirli Bir Kısmının Gösterilmesi

Programın asıl amacının PDF dokümanının belirli bir kısmının gösterilmesi olduğunu daha önce belirtmiştik. Görüntüleme işlemi arama işlemi sonucunda geri dönen sayfa numarasından itibaren on sayfadan ibarettir. Görüntüleme işlemi sırasında iText kütüphanesinden yararlanılmıştır.

PDF belgesinin kullanıcının web tarayıcısına aktarılması işlemi direk bellek üzerinden gerçekleştirilir. Bu sayede arama işlemleri sonucu parça PDF dokümanlarının sabit disk üzerinde oluşturulmasına gerek yoktur. Bu yapı; kullanıcıların dokümanlar üzerinde yaptığı arama sonuçları oluşacak parça PDF belgelerinin sabit disk üzerinden silinmesi gibi ve bu silinme işleminin ne zaman yapılacağı gibi problemleri de ortadan kaldırmaktadır. Böylece programcı bu problemler ile uğraşmak zorunda da kalmayacaktır.

Görüntüleme işlemi JSP yerine Servlet yapısıyla yapılmaktadır. PDF dokümanının web tarayıcısı içinde görüntülenmesi işleminin JSP ile yapılması birçok soruna neden olabilir. Çünkü her web tarayıcısı (IE, Mozilla Firefox, Opera, Google Chrome vs.) HTML kodlarını kendisine göre yorumlar.

Bazı sunucular JSP çıkışını ikili olarak algılamaz. Bu nedenle içerikte soru işaretleri görürsünüz. PDF dosyaları, problemsiz bir biçimde sunucuların dosya sistemine yazılır ancak bu dosyalar bir bağlantıya servis edildiği zaman, kullanıcı PDF belgesini açınca sadece boş sayfa görür. JSP sayfaları Servlet'lere dahili olarak gömülür. HTML hizmeti vermek için bir JSP sayfası yazmak yada benzer bir teknoloji kullanarak kodlamak bir Servlet yazmaktan daha kolaydır. JSP ile PDF dokümanı görüntülemeye çalışırsanız,

muhtemelen girintiler, boşluklar ve benzeri şeyleri eklemek zorunda kalacaksınız. JSP yazmaya alışkın iseniz, bunun alışkanlık olacağını bilmelisiniz. Her ne kadar bu yaptığınız yazdığınız kodların çoğu için iyi bir şey olsa da eğer ikili bir içerik üretmek istiyorsanız bu kötü bir fikirdir. PDF dokümanlarını görüntülemek için en iyi yol Servlet yapısını kullanmaktır.

Kullanıcının PDF dokümanını görüntüleyebilmesi için bilgisayarı üzerinde bir PDF okuyucusunun yada web tarayıcısı uygun bir PDF okuyucu eklentisine sahip olması gerekir. Eğer kullanıcı bilgisayarına AR kurarsa, web tarayıcısına eklenti de kurulmuş olur. Bazı web tarayıcıları dokümanın içeriğinden çok dosya uzantısına bakar. IE uzantısı ".pdf" olan PDF dokümanlarını iyi şekilde görüntüler. Bir PDF dokümanını bir Servlet yapısında hizmete sunarsanız, IE kullanıcılarının şikayetleri ile karşılaşabilirsiniz. Bu problemi gidermek amacıyla IE'ye dosyanın PDF olduğunu anlatacak ;

http://server.com/servlet/Servlet?dummy=dummy.pdf

şeklinde bir yol izlenmelidir. Fakat bu çok da şık değildir.

2.8. PDF Çatısı

PDF dokümanları ile ilgili olarak arama, raporlama ve PDF dokümanının belli bir kısmının görüntülenme işlemleri için belirli fonksiyonların oluşturulduğu daha önce belirtmiştik.

Java dilinde geliştirilen bir çalışma ile bu fonksiyonlar "PdfMaster.jar" isimli bir kütüphane içinde bir çatıda toplanmışlardır. Çatı içerisindeki fonksiyonların prototipleri aşağıdaki şekilde belirtilmiştir.

```
public static int Arama(String PdfPath, String Aranacak)
public static int HizliArama(String PdfPath, String Aranacak)
public static int Raporlama (ResultSet rs, String RaporBasligi)
```

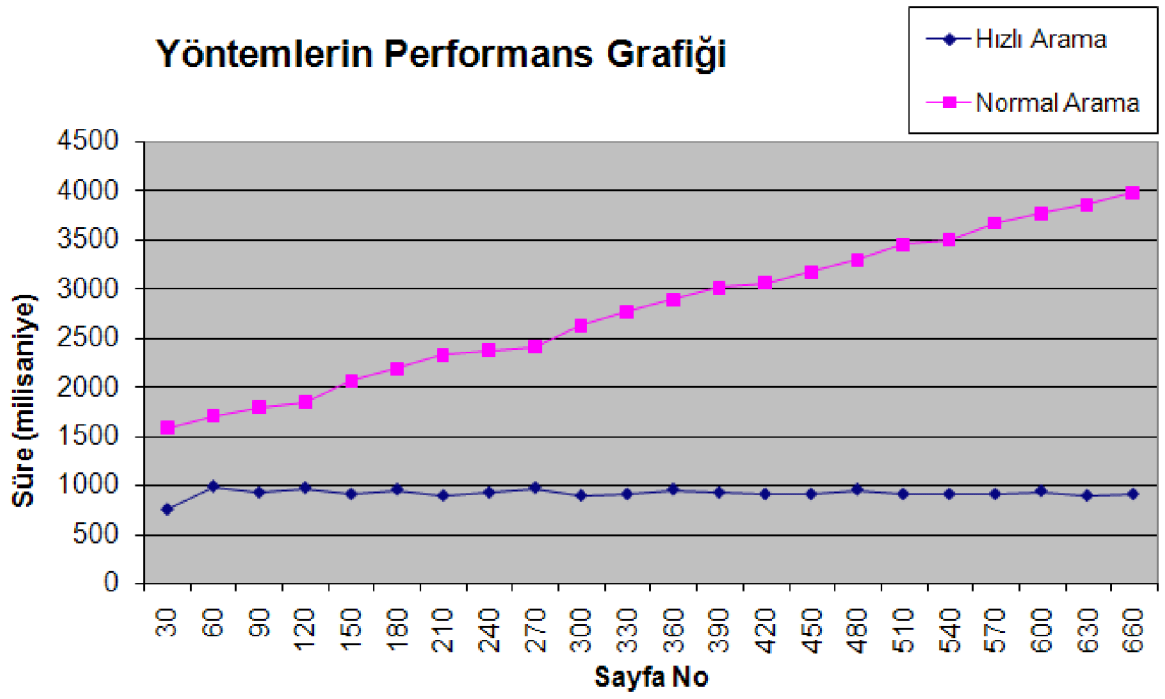
"Arama" fonksiyonu PDF dokümanı içerisinde bir metin ifadesinin aranması için kullanılmaktadır. Birinci parametre PDF dokümanının bilgisayar üzerinde nerede bulunduğuna, ikinci parametre ise doküman içerisinde aranacak metin ifadesini işaret etmektedir.

"HizliArama" fonksiyonu PDF dokümanının sayfa imleri üzerinde bir metin ifadesinin aranması işlemini sağlamaktadır. Birinci parametre PDF dokümanının bilgisayar üzerinde nerede bulunduğuna, ikinci parametre ise sayfa imleri üzerinde aranacak metin ifadesini işaret etmektedir.

Raporlama fonksiyonu herhangi bir tablo üzerinde yapılan sorgu sonucu oluşan sonuç kümesinin PDF formatında görüntülenmesini sağlar. Birinci parametre yapılan sorgu sonucu oluşan sonuç kümesini, ikinci parametre ise rapora verilecek başlığa işaret etmektedir.

Geliştirilen PDF çatısı sayesinde programcılar PDF ile ilgili bu işlemleri, PDF formatının karmaşık yapısını öğrenmeye gerek kalmadan kolaylıkla yapabilirler. "PdfMaster.jar" kütüphanesini uygulamalarına dahil etmeleri yeterli olacaktır.

Geliştirilen PDF çatısındaki arama fonksiyonları arasındaki hız performans açısından değerlendirilmesi Şekil 2.6.'de gösterilmiştir.



Şekil 2.6. Hızlı Arama ile Normal Arama arasındaki performans değerlendirmesi.

3. SONUÇLAR

Günümüzde birçok verinin elektronik ortamda saklanması çok yaygın bir duruma gelmiştir. Bu çalışmada bir elektronik kütüphane sistemi geliştirilmiştir. Platform (Unix, Macintosh, Windows, Linux veya Palm işletim sistemlerinde) bağımsızlığını sağlamak amacıyla dokümanlar için PDF dosya formatı seçilmiştir. Ayrıca, bir PDF dokümanın üretilmesi, PDF dokümanların içeriği veya sayfa imleri üzerinde bir sözcük veya sözcük öbeğinin aranması, veritabanında bulunan bir tablo üzerinde gerçekleştirilen sorgu sonucu oluşan kayıtların PDF formatında gösterilmesi ve ücretli yayın ve kitapların telif haklarının korunması amacıyla arama işlemi yapılan PDF dokümanının son kullanıcılara tarayıcı içinde belirli bir kısmını (on sayfasını) gösterme işlemleri yapılmıştır.

Doküman üzerindeki sözcük veya sözcük öbeği arama işlemleri sırasında, PDF doküman içeriğinin hepsi bir anda düz metine çevrilmemiştir. İkili yapıdan düz metine çevirme işlemleri dokümanın onlu sayfa blokları üzerinden yapılmıştır. Yapılan çalışmalar zaman bakımından en iyi sonucun, onlu sayfalar şeklinde ikili yapıdan düz metine çevrilerek arama yapılması gerektiğini göstermiştir. Örneğin; kullanıcının herhangi bir PDF dokümanı üzerinde bir sözcük öbeğini aradığını varsayarsak, bu sözcüğün arama işlemine geçilmeden, PDF dokümanı üzerinde tam olarak nerede olduğunu bilemeyiz. Fakat sözcüğün doküman içeriğinin başlangıcında, ortasında olma olasılığı vardır ve bu olasılık bize göz ardı edilemez bir zaman kazandırabilir. Bu durumda sözcüğün dokümanın başlangıçında veya ortasında bulunması, geri kalan kısmı üzerinde arama yapma işlemini gereksiz kılmaktadır. Bu algoritma zaman açısından büyük bir avantaj sağlamaktadır. Çünkü aranılan sözcüğün doküman üzerinde bulunması, dokümanın sonuçlar kümesine dahil edilmesi için yeterli olacaktır. Arama işlemleri için alınabilecek en kötü zaman süresinin, aranılan sözcük öbeğinin dokümanın sonuna doğru bulunması olduğu açıktır.

Veritabanı içindeki bir tablo üzerinde yapılan sorgu sonucu oluşan verilerin PDF formatında gösterildiğini ve Türkçe karakter problemini gidermek amacıyla da bir yazı tipi tanımlanması gerektiğini daha önce belirmiştik. Burada dikkat edilmesi gereken önemli bir husus bulunmaktadır. Bu husus, veritabanının ve içerisindeki tabloların Türkçe karakter yapısına uygun olarak oluşturulması gerektiğidir. Aksi halde yazı tipi tanımlaması yapılsa dahi Türkçe karakterler normal bir şekilde görüntülenemezler.

PDF dokümanların sayfa imleri üzerinde yapılan arama işlemleri sırasında, her PDF dokümanının standart bir sayfa imi yapısına sahip olmadığı görülmüştür. Farklı PDF'ler üzerinde çalışılarak farklı sayfa imleri için farklı algoritmalar kullanılmıştır.

Geliştirilen uygulama farklı bilgisayar konfigürasyonları üzerinde çalıştırılmış ve bilgisayar işlemcisinin çekirdek sayısı arttıkça, PDF dokümanlar üzerindeki arama işlemi zamanının azaldığı görülmüştür. Çünkü uygulama, en küçük iş parçası (Thread) yapısına göre çalışmaktadır.

Çalışma sırasında PDF dokümanın üretilmesi ve PDF dokümanlar üzerinde metin aranması hakkında 1 adet bildiri 2. Mühendislik ve Teknoloji Sempozyumu' na sunulmuş ve bildiri kitapçığında basılmıştır.

4. ÖNERİLER

1. Bu çalışmadan yararlanılarak çok daha gelişmiş elektronik kütüphane uygulamaları oluşturulabilir.
2. Geliştirilebilecek elektronik kütüphane uygulamalarına üyelik sistemi eklenerek üyelik derecelerine göre kullanıcılara farklı ayrıcalıklar (gösterilebilecek sayfa sayısında değişiklik gibi) verilebilir.
3. Geliştirilebilecek elektronik kütüphanede bir kullanıcının gün içinde yapabileceği arama işlemleri sayısı sınırlandırılabilir. Bu sınırlandırma işlemi, Internet Protokol (IP) veya Ortam Erişim Yönetimi (MAC) adresleri sayesinde yapılabilir.
4. PDF doküman üretilmesi işleminden yararlanılarak, herhangi bir düz metindeki yazı PDF dokümanı şekline dönüştürülebilir.
5. PDF'in her platformda görüntülenebilir olması özelliği sayesinde, çok büyük veritabanları üzerinden yapılacak sorgu sonuçlarının kullanıcılara ulaştırılması için geliştirilen çatı içerisindeki raporlama fonksiyonu Java, JSP ve Servlet içeren büyük uygulamalar için ideal olabilir.
6. Geliştirilmiş PDF çatısı, PDF ile ilgili işlemler için kolaylıkla kullanılabilir.

5. KAYNAKLAR

1. Adobe Systems Incorporated, Type 1 Font Format Supplement, United States of America, 1994.

2. Adobe Systems Incorporated, PostScript Language Referance Third Edition, United States of America, 1999.

3. Warnock, J., The Camelot Paper, First Edition, PlanetPDF, 1991.

4. Geschke, C. ve Warnock, J., PDF Reference, Fifth Edition, Adobe Press, 2004.

5. Adobe Systems Incorporated, Acrobat 7.0 PDF Open Parameters, United States of America, 2005.

6. Adobe Systems Incorporated, XMP Adding Intelligence to Media, United States of America, 2005.

7. Adobe Systems Incorporated, Adobe Type 1 Font Format, United States of America, 1990.

8. Steward, S., PDF Hacks, First Edition, O'Reilly Media, United States of America, 2004.

9. http://tr.wikipedia.org/wiki/Apache_Lisansı, Apache Lisansı, 17 Kasım 2009.

10. Kanlıoğlu, C., Javaloji-I, Cilt 1, 1.Baskı, Bilginç Bilgisayar Eğitimi ve Danışmanlığı Tic. Ltd. Şti, 2003.

11. http://www.gnu.org/copyleft/lesser.html, GNU Genel Kamu Lisansı, 22 Ekim 2009.

12. http://itextpdf.com/, iText Web Sitesi, 12 Mart 2009.

13. http://www.mozilla.org/MPL/, Mozilla Kamu Lisansı, 15 Kasım 2009.

14. http://itextpdf.com/terms-of-use/index.php, iText Kullanım Koşulları, 08 Aralık 2009.

15. Kanlıoğlu, C., Javaloji-II, Cilt 2, 1.Baskı, Bilginç Bilgisayar Eğitimi ve Danışmanlığı Tic. Ltd. Şti, 2003.

16. http://ieeexplore.ieee.org/Xplore/guesthome.jsp, IEEE Ana Sayfası, 14 Aralık 2009.

17. http://www.ebrary.com/corp/, Ebrary Ana Sayfası, 16 Aralık 2009.

18. http://www.sciencedirect.com/, ScienceDirect Ana Sayfası, 17 Aralık 2009.

19. http://www.elsevier.com/, Elsevier Ana Sayfası, 18 Aralık 2009.

20. http://pdfbox.apache.org/, PdfBox Ana Sayfası, 10 Nisan 2009.

21. http://sourceforge.net/projects/itext/, iText PDF Kütüphanesi, 04 Şubat 2009.

22. Adobe Systems Incorporated, Adobe Font Metrics File Format Specification, United States of America, 1998.

23. http://1t3xt.info/tutorials/keywords/, iText Dokümanları, 20 Eylül 2009.

24. http://pdfbox.apache.org/javadoc/index.html, PdfBox Dokümanları, 25 Kasım 2009.

25. Adobe Systems Incorporated, The Compact Font Format Specification, United States of America, 2003.

6. EKLER

Netbeans platformunda geliştirilen elektronik kütüphanenin çalışma aşamaları kodları kendilerine uygun başlıklar altında sunulmuştur.

Ek 1- Programın İlk Web Sayfasının (index.jsp) Kod Bloğu

```
<%--
   Document   : index
   Created on : Nov 13, 2009, 12:41:23 AM
   Author     : NEO
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ page import="java.sql.*;" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
  <%
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        Connection conTest =
DriverManager.getConnection("jdbc:mysql://localhost/pdf","root","1");
  %>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Ana Sayfa</title>
   </head>
   <body>
   <form action="arama.jsp" method="POST">
      <table align="center">
      <tr>
         <td>
            Kategori :

         </td>

      <td align="center">

      <select name="kategori">

         <option>Tum Kategoriler</option>

      <%

         Statement komut= conTest.createStatement();

         ResultSet rs = komut.executeQuery("SELECT DISTINCT(kategori) FROM pdf
Order By kategori");
```

Ek-1'in Devamı

```
      while(rs.next())
          {
            %>
            <option>
            <%
            out.print(rs.getString("kategori"));
            %>
            </option>
            <%
          }
       %>
       </select>
      </td>
   </tr>

   <tr>
      <td>
         <input name="aranacak">
      </td>
      <td>
         <input type="submit" value="Ara!">
      </td>
   </tr>

   </table>
   </form>
   </body>
</html>
```

Ek 2- Programın İkinci Web Sayfasının (arama.jsp) Kod Bloğu

```
<%--
   Document   : arama
   Created on : Dec 17, 2009, 2:00:31 PM
   Author     : NEO
--%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ page import="java.sql.*" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">

<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Arama</title>
   </head>
   <body>
      <%
         Class.forName("com.mysql.jdbc.Driver").newInstance();

         Connection conTest =
DriverManager.getConnection("jdbc:mysql://localhost/pdf","root","1");

         int wheresayi=0;
         String kategori=request.getParameter("kategori");
         String aranacak=request.getParameter("aranacak");
         if(!aranacak.equals(""))
            wheresayi++;
         if(!kategori.equals("Tum Kategoriler"))
            wheresayi++;
         String sorgu="SELECT * FROM pdf ";
         if(wheresayi>0)
         {
            sorgu+="WHERE ";
            if(!aranacak.equals(""))
            {
               sorgu+="adi LIKE '%"+aranacak+"%' ";
               if(wheresayi>1)
                  sorgu+="AND kategori='"+kategori+"'";
            }
            else
            {
               sorgu+="kategori='"+kategori+"'";
               if(wheresayi>1)
                  sorgu+="AND adi LIKE '%"+aranacak+"%' ";
            }
         }
```

Ek-2'in Devamı

```
Statement komut= conTest.createStatement();
        ResultSet rs = komut.executeQuery(sorgu);
        %>
        <script language="javascript">
          function AramaTipiSetle(AramaTipi)
          {
            if(document.SecmeForm.aranacak.value=="")
              alert('Lütfen aranacak kelimeyi girin');
            else
            {
              document.SecmeForm.aramatipi.value=AramaTipi;
              document.SecmeForm.submit();
            }
          }
        </script>
        <form action="secme.jsp" method="POST" name="SecmeForm">
        <table>
        <tr><td colspan="2" align="center"><b>Bulunan PDFler</b></td></tr>

        <%
        while(rs.next())
        {
        %>
       <tr>
         <td>
           <input type="checkbox" name="pdfchck" value="
           <%
           out.print(rs.getString("id"));
           %>
           ">
         </td>
         <td>
           <%
           out.print(rs.getString("adi"));
           %>
         </td>
       </tr>
  <%
       }//while(rs.next())
  %>
  <tr><td colspan="2">
      <input name="aramatipi" value="" type="hidden">
      <input name="aranacak">
      <input name="Arama" value="Arama" type="button"
onclick="AramaTipiSetle('Normal');">
      <input name="hizliarama" value="Hızlı Arama" type="button"
onclick="AramaTipiSetle('Hizli');"> </td></tr></table></form></body></html>
```

Ek 3- Programın Üçüncü Web Sayfasının (secme.jsp) Kod Bloğu

```
<%--
   Document   : secme
   Created on : Dec 19, 2009, 12:50:20 AM
   Author     : NEO
--%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@ page import="java.sql.*,PdfMaster.PdfCore" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>Görüntülenecek Dosya Seçme</title>
   </head>
   <body>
      <form name="goruntuleme" action="goruntuleme" method="POST"
target="_blank">
         <input type="hidden" name="baslangic">
         <input type="hidden" name="bitis">
         <input type="hidden" name="pdfid">
      </form>
      <script language="javascript">
         function SecmeSetle(bas,son,id)
         {
            document.goruntuleme.baslangic.value=bas;
            document.goruntuleme.bitis.value=son;
            document.goruntuleme.pdfid.value=id;
            document.goruntuleme.submit();
         }
      </script>
      <% String[] pdfchck;
         Class.forName("com.mysql.jdbc.Driver").newInstance();
         Connection conTest =
DriverManager.getConnection("jdbc:mysql://localhost/pdf","root","1");
         if (request.getParameter("aranacak") != "") {
            out.println("Aranan Kelime : " + request.getParameter("aranacak") + "<br>");
         }
         pdfchck = request.getParameterValues("pdfchck");
         if (pdfchck != null) {
            %>
            <table border="1" bordercolor="darkblue" cellpadding="2"
cellspacing="0"><tr bgcolor="darkblue" style="color:white"><td>PDF
Adı</td><td>Sayfa No</td><td>Görüntüleme</td><td>Arama Zamanı</td></tr>
            <%
            for (int i = 0; i < pdfchck.length; i++) {
            Statement komut= conTest.createStatement();
            ResultSet rs = komut.executeQuery("SELECT id,path,adi FROM pdf WHERE
```

Ek-3'in Devamı

```
id="+pdfchck[i]);
            rs.next();
            int SayfaNo=-1;
            if(request.getParameter("aramatipi").equals("Normal"))
               SayfaNo=PdfCore.Arama(rs.getString("path"),
request.getParameter("aranacak"));
            else
               SayfaNo=PdfCore.HizliArama(rs.getString("path"),
request.getParameter("aranacak"));
            if(SayfaNo!=-1)
            {
               out.print("<tr><td>");
               out.print(rs.getString("adi"));
               out.print("</td><td>");
               out.print(SayfaNo);
               out.print("</td><td>");
               out.print("<a href='#'
onclick=\"SecmeSetle('"+String.valueOf(SayfaNo)+"','"+String.valueOf(SayfaNo+10)+"','
"
+rs.getString("id")+"');\">Görüntüle</a>");
               out.print("</td><td>");
               out.print(PdfCore.AramaZamani);
               out.print("</td></tr>");
            }
            }
            %>
            </table>
            <%
            } else {
            out.println("<b>none<b>");
            }

      %>
   </body>
</html>
```

Ek 4- Programın Dördüncü Web Sayfasının (goruntule) Servlet Kod Bloğu

```
package Kodlar;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.lowagie.text.Document;
import com.lowagie.text.pdf.PdfCopy;
import com.lowagie.text.pdf.PdfReader;
import com.lowagie.text.pdf.PdfWriter;
import java.io.ByteArrayOutputStream;
import javax.servlet.ServletOutputStream;

import java.sql.*;

/**
 *
 * @author NEO
 */
public class goruntuleme extends HttpServlet {

    /**
    * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
methods.
    * @param request servlet request
    * @param response servlet response
    */
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
} finally {
            out.close();
        }
    }
```

Ek-4'in Devamı

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {
      processRequest(request, response);
   }
protected void doPost(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {
      makePdf(request, response, "POST");
   }
public String getServletInfo() {
      return "PDF'i web tarayıcıda gösteren servlettir.";
   }
      public void makePdf(HttpServletRequest request,
                        HttpServletResponse response, String methodGetPost)
                        throws ServletException, IOException {
                  try {
                  String id = (String) request.getParameter("pdfid");
                   String bas = (String)request.getParameter("baslangic");
                   String son = (String)request.getParameter("bitis");
                   if(id==null || bas==null || son==null)
                      return;
  Class.forName("com.mysql.jdbc.Driver").newInstance();
                   Connection conTest =
DriverManager.getConnection("jdbc:mysql://localhost/pdf","root","1");
                   Statement komut= conTest.createStatement();

                   ResultSet rs = komut.executeQuery("SELECT path,adi FROM pdf
WHERE id="+String.valueOf(id));
                   if(!rs.next())
                      return;
  PdfReader reader = new PdfReader(rs.getString("path"));
                   Document document = new Document();
                           ByteArrayOutputStream baos = new ByteArrayOutputStream();
                           PdfCopy copy = new PdfCopy(document, baos);
                   int sonn=Integer.parseInt(son);
   if(reader.getNumberOfPages()+1<sonn)
                      sonn=reader.getNumberOfPages()+1;
document.open();
                   for (int i = Integer.parseInt(bas); i < sonn ; i++ ) {
                   copy.addPage(copy.getImportedPage(reader, i));
                   }
                           copy.setViewerPreferences(PdfWriter.PageModeUseThumbs);
```

Ek-4'in Devamı

```
                document.close();


                response.setHeader("Expires", "0");
                response.setHeader("Cache-Control",
                        "must-revalidate, post-check=0, pre-check=0");
                response.setHeader("Pragma", "public");

                response.setContentType("application/pdf");

                response.setContentLength(baos.size());

                ServletOutputStream out = response.getOutputStream();
                baos.writeTo(out);
                out.flush();

            } catch (Exception e2) {
                System.out.println("Error in " + getClass().getName() + "\n" + e2);
            }
        }
}
```

ÖZGEÇMİŞ

1983 yılında Trabzon'da doğdu. İlköğrenimini Tevfikbey İlkokulu ve Cumhuriyet İlköğretim Okulu'nda, orta öğrenimini Trabzon Lisesi'nde tamamladı. 2000 yılında Karadeniz Teknik Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü'nde lisans programına başladı ve 2005 yılında bu bölümden mezun oldu.

İki yıla yakın bir süre özel sektörde çalıştıktan sonra, 2007 yılında Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda yüksek lisans programına başladı. 2007 yılının Kasım ayında Ordu Üniversitesi Mesudiye Meslek Yüksekokuluna Öğretim Görevlisi olarak ataması yapılmıştır ve halen bu görevine devam etmektedir. Yabancı dil olarak iyi derecede İngilizce bilmektedir.