T.C.
HALİÇ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

HASTALIK TEŞHİSİ İÇİN KURUMSAL UYGULAMA ARAYÜZLERİ VE PERFORMANS ANALİZLERİ

YÜKSEK LİSANS TEZİ

Hazırlayan
Doruk Tolga Atasoy

Danışmanı
Prof. Dr. Halûk GÜMÜŞKAYA

İstanbul – Haziran 2011

İÇİNDEKİLER

Sayfa No

KISALTMALAR LİSTESİ

ANN: Artificial Neural Networks
API: Application Programming Interface
B2C: Business to Customer
CDDL : Common Development and Distribution License
DAO: Data Access Objects
DBMS: Database Management System
EIS: Executive Information System
EJB: Enterprise Java Beans
HTML: Hyper Text Markup Language
HTTP: Hypertext Transfer Protocol
J2EE: Java 2 Enterprise Edition
JDBC: Java database connectivity
JDK: Java Development Kit
JEE: Java Enterprise Edition
JSP: Java Server Pages
PDA: Personal Digital Assistant
RACF: Resource Access Control Facility
RMI: Remote Method Invocation
SQL: Structured Query Language
TAIFM: Technical Information Architecture For Information Management
WAP: Wireless Application Protocol
WML: Wireless Markup Language
XML: Extensible Markup Language

SEKİLLER LİSTESİ

Sayfa No.

TABLOLAR LİSTESİ

Sayfa No

GENEL BİLGİLER

Adı ve Soyadı : Doruk Tolga Atasoy
Anabilim Dalı : Mühendislik
Programı : Bilgisayar Mühendisliği
Tez Danışmanı : Prof. Dr. Halûk Gümüşkaya
Tez Türü ve Tarihi : Yüksek Lisans – Mayıs 2011

HASTALIK TEŞHİSİ İÇİN KURUMSAL UYGULAMA ARAYÜZLERİ VE PERFORMANS ANALİZLERİ

ÖZET

Bu tezde temel hedef bazı kurumlarda var olan eski yazılımları dış dünyaya yeni web ve mobil arayüzlerle açmak için yeni mimari yaklaşımların geliştirilmesidir. Üzerinde çalışılan "Kurumsal Uygulama Arayüzleri" geliştirme alanı olarak bir hastanede kullanılabilecek hastalık teşhisine yönelik yazılımlar seçilmiştir.

Hastaların hastaneden aldığı rapor verileri, tezde "eğitim verisi" olarak kullanılmaktadır. Bu veriler tezde, Yapay Sinir Ağları (YSA) algoritmaları ile çalışan daha önceden ve bu tezde yeni geliştirilmiş programların eğitilmesinde kullanılmaktadır. Bu programlar hasta verileriyle eğitilerek çeşitli hastalıkların teşhisini yapabilmektedir. Eski yazılımların bir web sunucuda çalışan yeni web programlarıyla iletişimleri sağlanarak, Internet üzerinden bu eski yazılımların kullanımı sağlanmıştır.

YSA algoritmalarıyla çalışan programlar, küçük mobil cihazlar üzerinde çalıştırılarak, bir hastanın elinde bir cep bilgisayarıyla sunucudan bağımsız hastalık teşhisi yapabilmesi sağlanmıştır. Geliştirilen mobil ve web uygulamalarının performans analizleri tezde yapılmıştır.

Anahtar Kelimeler: Kurumsal Uygulama Arayüzleri, Bilgisayar Öğrenme Algoritmaları, Yapay Sinir Ağları, Performans Analizleri, Mobil teknolojiler, Yapay Sinir Ağları İle Hastalık Teşhisi

GENERAL INFORMATION

Name and Surname : Doruk Tolga Atasoy
Field : Engineering
Program : Computer Engineering
Supervisor : Prof. Dr. Halûk Gümüşkaya
Degree and Date : Master – May 2011

ENTERPRISE APPLICATION INTERFACES FOR DIAGNOSING ILLNESSES AND PERFORMANCE ANALYSES

ABSTRACT

The main motivation of this thesis is to develop new architectural approaches to open the legacy software which exists in organizations to the external world using new web and mobile interfaces. The applications used in a typical hospital for diagnosing illnesses were chosen as an area to be studied to develop “Enterprise Application Interfaces”.

The reports taken by patients from hospitals are used as “training data” in this thesis. This data in this thesis is used to train the applications based on artificial neural networks (ANN) which have been developed before and in this thesis. These programs can diagnose several illnesses after they have been trained with patients’ data. The use of legacy software over the Internet was established by providing the communication between old legacy applications and new web applications running on the web server.

The applications based on ANN algorithms were also run on small mobile devices, and they are provided to diagnose illnesses without a server on a packet PC carried by a patient. The performance analyses of mobile and web applications developed in this thesis were also conducted.

Key Words: Enterprise, Application Interfaces, Computer Learning Algorithms, Artificial Neural Networks, Performance Analysis, Mobile Technologies, Illness Diagnosis by using Artificial Neural Networks

1. GİRİŞ

Bu tez çalışmasında önce günümüz kurumsal uygulamalara yönelik yazılım arayüzleri geliştirme teknikleri araştırılmıştır. Bir kurum tarafından belirli bir kullanıcı grubuna sunulacak yazılım geliştirme yöntemleri içerisinde birçok teknoloji arasında en uygun olanına karar verilerek, yine geliştirilmesi düşünülen bu uygulamanın bilgisayar, web ve mobil ortamlardaki performanslarının analiz edilmesi için araştırmalar yapılmıştır.

Tezde üzerinde çalışılan ana problem alanı olarak, bir hastanenin hastaları için sunmuş olduğu hastalık teşhisine yönelik önceden geliştirilmiş eski programların dış dünyaya açılması hedeflenmiştir. Tezde bu temel amaçla geliştirilen yazılımlar, web arayüzü, farklı sunucu yapıları ve ufak taşınabilir el cihazları için hastalık teşhisine yönelik değişik mobil arayüzler olarak 3 temel parçadan meydana gelmektedir.

Hastalık teşhisi için birçok program ve uygulama modellenmiş ve hayata geçirilmiştir. Tasarlanan bu programların çoğu geliştirilirken, bu konuyla ilgili önceki çalışmalardan da faydalanılmış ve geliştirme aşamasında mümkün olduğunca en güncel teknolojiler kullanılmaya çalışılmıştır. Hastalık teşhisi yapan programlarda, Java, C# ve Delphi bilgisayar programlama dilleri kullanılmış ve nesne yönelimli programlama prensiplerine uyulmuştur. Geliştirilen bu yazılımlar mobil cihazlar ve web sunucusu gibi birbirinden farklı platformlarda çalıştırılabilecek düzeyde olup sonraki güncellemeler ve değişimler için kodlar mümkün olduğunca kolay anlaşılabilir seviyede yazılmıştır. Geliştirilen bu programların hastalık teşhisi yapabilmeleri için Yapay Sinir Ağı (YSA) algoritmalarından yararlanılmıştır. YSA insan beyninin matematiksel bir modelidir. Bu matematiksel model değişik programlama yöntemleriyle uygulamaların temelini oluşturmuştur.

Tüm bu programlar ve uygulamalar tasarlanırken, bu programların ve uygulamaların herhangi bir kurumda kullanılabilecek düzeyde olmalarına ve kurumsal uygulama arayüzlerinin ilgi alanından çıkmamalarına dikkat edilmiştir. Bu uygulamaların örnek kurum olarak alınan herhangi bir hastanenin teşhis işlemlerini gerçekleştirebilecek nitelikte olması ve herkesin rahatlıkla anlayabileceği nitelikte, yani kullanıcı dostu arayüzlere sahip olması düşünülmüştür.

Geliştirilen uygulamaların performans analizinde birçok kriter dikkate alınarak, günümüzde ileri derecede gelişmiş olan mobil teknolojilerinin kullandığı işletim sistemleri ile kişisel bilgisayarların kullandığı işletim sistemlerindeki çalışma anlarında gösterdikleri performans farkları incelenmiş ve yoğun kapasitede işlem yapan uygulamaların

günümüzde mobil cihazlarda kullanıcıların isteklerine ne kadar karşılık verebilecekleri hesaplanmış ve incelenmiştir.

1.1. Tezin Amacı ve Kullanılan Yöntemler

Bu tezin amacı bir hastanenin hastalarına ev ve işyeri gibi hastane dışından hastalık teşhisi yapabilme imkanını, kurumsal uygulama arayüzleri kullanan prototip uygulamalar ve yazılımlar aracılığıyla sağlamaktır. Ayrıca yine bu prototip yazılımlar gerçekleştirildikten sonra, aktif çalışma anlarında mobil uygulamanın, sunucu tarafındaki uygulamanın ve yine web üzerinden sunulan arayüz uygulamasının performanslarının analiz edilerek geliştirilen sistemin kullanılabilirliğinin gösterilmesidir. Tüm bu çalışmalar yapılırken günümüz modern Kurumsal Uygulama Arayüzleri ve Kurumsal Arayüz Geliştirme yöntemlerinin incelenmesi de amaçlanmıştır.

Tezin geliştirilme esnasında daha önceden gerçekleştirilmiş ve uygulanmış projelerden yararlanılmıştır. Bu projelerden en önemlisi Borland Delphi teknolojileriyle YSA teknikleri ile geliştirilmiş olan “Teşhis” uygulamasıdır [1]. Tezimizde güvenilir YSA motoru olarak kullanılan bu eski uygulamanın, bir sunucuda çalışan bir web yazılımı ile iletişimi sağlanıp Internet ortamından Delphi ile çalışan bu yazılıma erişim sağlanmıştır. Ayrıca yine .Net 2008 ortamında ve C# diliyle geliştirilen bir YSA modelleme uygulaması temeli [2] üzerine mobil teşhis uygulamaları da geliştirilmiştir.

Hastalık teşhis uygulamaları geçmişte birçok farklı dalın uzmanı tarafından uzun zamanlar alan yoğun çalışmalar sonucu geliştirilmiştir. Dolayısıyla eskinin mirası olan benzer uygulamaları yeni teknolojilerle yeniden geliştirmeden önce, bu uygulamalar için sadece yeni web ve mobil arayüzler geliştirerek yaygın olarak hizmete sunmak çok önemlidir. Bu sayede bazı hastalıklar için, bir hastanın doktorla yüzyüze görüşmeden de hastalığını teşhis edebilmesi imkanı daha yaygınlaşacaktır.

Tezde yararlanılan eski hastalık teşhis programları hizmete sunulurken en uygun ve etkin teknolojiler tercih edilmiştir. Hızla gelişmekte olan ve güncellenen Java teknolojileri bu tez için uygun platformlardan birisi olarak düşünülmüştür. Ayrıca yine mobil cihazlar için .NET ortamında “smart phone programming” teknolojisiyle geliştirilmiş yazılım kullanıcı dostu hale getirerek mobil kullanıma da sunmak bu tezde önemli hedeflerden biri olmuştur.

Tezde yeni web ve mobil arayüzleri geliştirilen eski hastalık teşhis programlarının ana yapısında, yukarıda bahsedildiği gibi, YSA algoritmaları kullanılmaktadır. Hastalıklarla

ilgili verilerin bu yapay sinir ağları ile işlenmesi ve tezde geliştirilen programlar aracılığıyla hastaların anlayabilecekleri şekilde teşhis sonuçlarının hastalara sunulması hedeflenmiştir. Hastalıkların teşhisi için gerekli olan veriler hastanelerin veritabanlarından toplanmış olup uygulamalarda gerçek hasta verileri kullanılmıştır.

1.2. Tezin Yapısı

Bu tez 8 bölümden oluşmaktadır. İlk bölümde tezin konusuna bir giriş yapıldıktan sonra ikinci bölümde bu tezle ilgili akademik çalışmalar incelenmiştir.

Üçüncü bölümde, öncelikle bu tezin ana konularından birisi olan Kurumsal Uygulamalar, performans analizleri, yapay sinir ağları ve mobil programlama konusundaki güncel yaklaşımlar kısaca sunulmaktadır. Tezdeki yazılımlar geliştirilmeden önce, bu konularda araştırma, teknoloji öğrenme ve ön yazılım geliştirme çalışmaları yapılmış ve bu araştırmaların sonucunda teknoloji seçimindeki kararlara varılmıştır. Bu bölümde, yapay sinir ağları modelleri üstüne kurulmuş hastalık teşhis uygulamaları incelenmiştir. Ayrıca düşünülen uygulamaları gerçekleştirmek için gerekli olan yazılım geliştirme platformları ve teknolojilerine atıflarda bulunularak açıklamalar ve incelemeler yapılmıştır. Yazılım geliştirme işlemi tamamlandıktan sonra, karşılaşılan performans analiz gereksinimleri hakkında araştırmalar yapılarak bu performans analizlerini gerçekleştirebilecek ortamlara örnekler sunulmuştur.

Tezin dördüncü bölümünde ise geliştirilen web arayüzü detaylarıyla anlatılmış ve çalışma anlarındaki kullanılan önemli arayüzler sunulmuştur. Bu arayüzlerin hangi teknolojiler kullanılarak tasarlandığı ve çalışma prensipleri açıklanmıştır.

Tezin beşinci bölümünde ise aynı şekilde arayüz konusuna farklı bir açıdan devam edilmiştir. Buradaki fark bu arayüzün mobil cihazlar için olmasıdır. Mobil cihazlar için düşünülen çalışma prensipleri ve temel görünümler bu bölümde açıklanmıştır.

Tezin altıncı bölümü ise, sistem gerçekleştirildikten ve uygulamaya sokulduktan sonraki aşama olarak geçer. Bu bölümde gerçekleştirilmiş olan yazılımların tüm açılardan performansı analiz edilmektedir. Farklı ortamlarda performanslar karşılaştırılarak, yapay sinir ağlarının çalışma ortamlarının evrenselliği sorgulanmıştır. Yani her tür cihazda yapay sinir ağları modelleri üzerine kurulu hastalık teşhis programlarının istenilen verimlilikte çalışıp çalışamayacağı araştırılmıştır.

Yedinci bölüm, tez çalışmalarının bir sonuca bağlandığı bölümdür. Bu tezle nasıl bir gelişmeye fayda gösterilebileceği ve hangi akademik konularda aydınlatıcı bir kaynak

olabileceği sunulmaktadır. Tüm yapılan çalışmaların temel çıkış noktası olan, bir Kurumsal Yazılım Mimarisi ile bir hastane hastalarının rahatlıkla hastane dışından hastane verilerine ve programlarına erişimini ve bulundukları yerden kendi kendilerine hastalıklarını teşhis edebilmelerini gerçekleştiren bir projenin, bir kuruma ve kuruma bağlı kişilere nasıl faydalar ve kolaylıklar sağlayacağı sunulmaktadır.

Sekizinci son bölümde bu tezde yararlanılan kaynakların bir kısmı yer almaktadır.

2. İLGİLİ ÇALIŞMALAR

2.1. Giriş

Bu bölümde tezde çalışılan konulara yakınlık gösteren uygulamalar, yaklaşımlar, araştırmalar incelenmekte ve yine bunlara bağlı olarak mevcut teknolojiler, istemci-sunucu tabanlı sistemler, yazılım teknikleri ve yapay sinir ağları ile hastalık teşhisi gerçekleştiren sistemler sunulmaktadır.

Yapay sinir ağları farklı hastalıkların teşhisi için geniş bir alanda kullanılan matematiksel bir yaklaşımdır [3]. Bundan dolayı hastalıkların teşhisi için geliştirilmesi düşünülen yazılımların yapay sinir ağları tekniklerini ve yöntemlerini kullanmasına karar verilmiştir.

2.2. Akademik Çalışmalar

Kurumsal uygulama arayüzleri (Enterprise Application Interfaces) ve kurumlarda kullanılabilecek nitelikteki uygulamaları geliştirme alanında özellikle günümüzde sayısız çalışma mevcuttur. Bu bölümde, bu tezin genel konusuna en yakın görünen örneklerden bazıları incelenmiştir.

İlk olarak bu tezin ilgilendiği ana başlıklardan birisi olan hastalık teşhisi ve hasta takibi gibi konulara paralellik gösteren bir çalışma, “Yaygın Sağlık Koruma” sistemi Georgia Devlet Üniversitesinden Upkar Varshney tarafından 2007 senesinde öne sürülmüştür [4]. Amerika’daki sağlık sektöründe, hızla yükselen maliyetler, medikal sorunlar, kısıtlı finansal imkanlara rağmen sağlık çalışanlarının hastalara daha iyi bir hizmet sunmalarının gerekliliği gibi problemler bulunmaktadır. Kablosuz ağların gelişimi ve yaygın olarak kullanılması, hastalar, terapistler, doktorlar ve diğer sağlık koruma sektöründe çalışanlar arasındaki irtibatı güçlendirmekte ve kolaylaştırmaktadır. Kablosuz teknolojiler, ambulans çalışanlarına, hastane personeline gerçek zamanlı bir şekilde ihtiyaç duyulacak ilaçları bildirme imkânını sunmaktadır. Bu da hastane çalışanın durumu değerlendirmesine ve hastanın tedavisi için gerekli olan ortamı sağlamalarına kısa bir sürede imkân vermektedir. Kablosuz ağlar hastaları devamlı olarak her yerden takip edebilme imkanı sağlamaktadır. Hastalık veya ciddi bir problem olma durumunda doktora hızlı bir şekilde bildirimde bulunma imkanı sağlamaktadır ki, bu da ani kriz durumlarında hastanın kurtulma şansını yükseltmektedir. Vücuda bağlanan ara birimler aracılığıyla elde edilen veriler hızlı bir şekilde kablosuz ağlardan aktarılabilir ve doktor bu verileri anlık

olarak değerlendirebilir. Tüm bu gelişmeler tedavi ve teşhis hızında büyük artış imkanı sağlayabilecektir. Acil durumlara zamanında müdahale imkanı verecektir. Bu çalışma bilgisayar ve kablosuz teknolojilerin sağlık sektörüne nasıl uyarlanabileceği hakkında yeni fikirler sunmaktadır. Bu yeniliklere örnek olarak, bazı yazılımlar aracılığıyla sağlık personelinin hastalık teşhisi yapabilmesine yardımcı olabilmesi verilebilir.

Bir problem bir yapay sinir ağı ile modellendikten sonra, eğitim ve test gibi işlemler sonucunda bazı kararlara varılır. Teşhisi yapılmak istenilen hastalıkların yapay sinir ağları ile modellenerek, bir bilgisayar programı aracılığıyla analiz edilmesi ile ilgili birçok çalışma gerçekleştirilmiş ve uygulanmıştır. Bu tezin örnek problem konularından birisi olan diyabet hastalığının teşhisinde de bu ağın eğitimi ve test edilmesi gerekmektedir. Almanya'da yapılan bir çalışmada [5] diyabetik bir kişinin kan şekeri metabolizması için tasarlanan farklı yapay sinir ağları modelleri kullanılarak alınan sonuçlar birbiriyle karşılaştırılarak deneysel sonuçlar ortaya çıkarılmaktadır. Bu çalışmada aynı zamanda diyabet hastalığının belirgin özelliklerine değinilmiştir. Yapay sinir ağları kullanarak diyabet hastalığının teşhisini sağlamak ve diyabet hastalığında rol oynayan doğal mekanizmaları yapay sinir ağları modellerine aktarmak hedeflenmektedir.

Bu alanda yine benzer bir çalışma M. Pradhan ve R. K. Sahu tarafından 2011 yılında Hindistan'ın Fakir Mohan Üniversitesinde gerçekleştirilmiştir [6]. Bu çalışma, öğrenme algoritmasıyla bir yapay sinir ağını test etmeyi hedeflemektedir. Çalışmada ileri beslemeli geri yayınım algoritması öğrenim tekniğiyle eğitilmektedir. Yapay sinir ağları ile Tip 1 (insülinden bağımsız) ve Tip 2 diyabetin teşhisinde önemli sonuçlar almıştır. Çalışmada hastalık bulgularının desenleri yapay sinir ağları aracılığıyla tanınmıştır. Çalışma aynı zamanda yapay sinir ağlarının hastalık teşhisindeki güvenilirliğini de göstermektedir.

Göğüs kanserinin yapay sinir ağları ile teşhis edilmesine yönelik, P. Venkatesan ve M. L. Suresh tarafından Hindistan'ın Sathyabama Üniversite'sinde 2009 yılında bie çalışma gerçekleştirilmiştir [7]. Çalışmada göğüs kanseriyle ilgili veriler Hindistan'da bulunan bir kanser enstitüsünde kayıtlı bulunan 2000 civarında hastadan toplanmıştır. Bu çalışmada çok katmanlı bir yapay sinir ağı modeli ve sigmoid fonksiyonu kullanılmaktadır ve ağ geri yayınım algoritması ile eğitilmektedir. Sistem modelinin oturtulmasında profesyonel bir yapay sinir ağı eğitim uygulaması olan NeuNet Pro kullanılmıştır. Elde edilen sonuçlar konusunda bir uzmanın değerlendirmesinden de yararlanılmıştır. Bu çalışma yapay sinir ağlarının yeni ve merak edilen alanlardaki yeteneklerini ve kapasitesini de gözlemlemiş ve YSA modelinin diğer modellere göre daha iyi ve uygun bir model olduğu kararına

varılmıştır. Çalışmaya göre yapay sinir ağları uygun bir şekilde veri girişleriyle doğru bir yolla eğitildiğinde doğru tahmin sonuçları vermektedir.

Bu tezde yapay sinir ağlarını taşınabilen küçük bir cihazda çalıştırabilmek ve kabul edilebilir sonuçlar alabilmek gibi bir problem üzerinde çalışılmıştır. Çünkü bu cihazlar genellikle güçlü mikroişlemcilere, bilgisayarlara kıyasla geniş hafızaya, yüksek hızlı giriş/çıkış ve ağ kapasitelerine sahip değildir. Birçok yapay sinir ağı algoritması bu elemanların güçlü ve yeterli olmasına ihtiyaç duyacağından bu algoritmaları mobil cihazlarda kullanmak bazen zor hatta imkansız olabilir. Son zamanlarda mobil cihazların donanım kapasitelerindeki gelişmelerin ve eklenen yeni özelliklerin sonucu, eskiden mümkün olmayan uygulamaların bu cihazlar için de geliştirmeleri mümkün olmuştur. Haliç Üniversitesi Bilgisayar Mühendisliği Bölümünde 2010 yılından [2] günümüze yapılan değişik araştırma projelerinde, YSA tekniklerini kullanan ve diyabet, iç hastalıkları ve kolesterol gibi değişik hatalıkların teşhisine yönelik cep bilgisayarları için çeşitli uygulamalar geliştirilmiştir. Bu araştırmalarda iyi sonuçlar alınmış ve günümüz mobil cihazlarının yeni uygulamalar geliştirmek için hazır olduğu görülmüştür.

Tezde kullanıcılara sunulan arayüzler birçok farklı yazılım platformlarında gerçekleştirilmiştir. Java ve .Net temel teknolojiler olmakla beraber bunlardan farklı olarak değişik teknolojilerden de yararlanılmıştır. Java teknolojileri kurumsal uygulama arayüzleri geliştirme ve kurumsal nitelikte kullanılabilecek uygulamalar hazırlamak için kolaylıklar ve karmaşık bir çalışmayı daha basitleştiren yazılım altyapıları (framework) sunmaktadır. Önemli Java kurumsal uygulamalar geliştirmede kullanılabilecek alt yapıları açıklayan ve bu konuyu aydınlatmaya çalışan bir çalışma Rod Johnson tarafından yapılmıştır [8]. Bu çalışmada Java Kurumsal Mimarisi ve Web yazılımları konusunda geniş bir bakış açısı sunulmaktadır. JEE'nin getirdiği avantajlar ve karşılaşılan bazı problemler verilmektedir. Ayrıca genel mimari yaklaşımlar hakkında bilgiler verilmekte ve yazılım altyapılarının kullanılmasındaki amaçlardan bahsedilmektedir. Yazılım altyapılarının kullanımı ve geliştirilmesi esnasında açık kaynak kodların sahip oldukları önem, kullanım alanları ve yöntemlerinden de bahsedilmektedir. Web teknolojilerinden Struts, Hibernate ve Spring gibi teknolojilere değinilmiş ve yeni nesil teknolojilerin getirebileceği yenilikler sunulmuştur.

Bu tezin konulardan birisi de geliştirilen yazılımların ve web uygulamalarının performanslarının kabul edilebilir ölçüt olup olmayacağıdır. Bu ölçütlerden önemli bir örnek, geliştirilen bir uygulamanın farklı platformlarda sergileyeceği çalışma süresidir.

Zaman faktörü performans hakkında karara varmak için önemli bir ölçüttür. Buna benzer olarak daha birçok ölçüt bulunmaktadır. Performans testi bir uygulamanın bilinen yükleme koşulları altında çalışabileceğini doğrulamak için kullanılan bir yöntemdir. Performans testleri uygulamayı özel koşullarla ve girişlerle test ederek performans ölçütlerini elde eder. O. Hamed ve N. Kafri yaptıkları çalışmada .NET ve Java EE yazılım altyapılarının performans analizlerini sunmakta ve iki önemli teknolojiyi karşılaştırmaktadır. Performans analizlerinde Java EE ile alınan sonuçların .NET ortamına göre daha iyi olduğundan yazılım geliştiriciler açısından Java EE teknolojilerinin daha cazip olabileceği sonucuna varmışlardır.

Bu tezde yapılan çalışmaların yukarıda verilen çalışmalara benzer birçok yönü vardır. Fakat bazı noktalardan farkları da bulunmaktadır. Bu farklara kısaca değinmek gerekirse, öncelikli olarak, geniş yapılı bir sistem olması, yani birçok farklı teknolojiyi bir araya getirerek tek bir sistem oluşturmayı hedeflemesidir. Örneğin, alt yapıda YSA motoru olarak hastalık teşhisinde kullanılan Delphi teknolojileri ile yazılmış bir programı, Java web teknolojileri ile kullanıcıya sunmakta ve Java dünyasının hızla gelişen yeniliklerinden faydalanabilmektedir. Diğer bir örnek olarak, C# diliyle programlanmış ve .Net teknolojilerini kullanan akıllı telefon uygulamaları, mobil cihaz kullanımı ihtiyacına karşılık vermek için geliştirilmiştir. İşte tüm bu farklı yapıdaki uygulamalar birleştirilerek meydana getirilen sistem, herhangi kurumun kullanabileceği nitelikteki arayüzlerle kullanıcıların kolay anlayabilecekleri ve sonuca rahat ulaşabilecekleri bir mimari sunmayı hedeflemektedir.

3. İLGİLİ KONULAR

3.1. Kurumsal Uygulamalar

Kurumsal uygulamalar büyük çaplı işyerlerinde karşılaşılan yüksek miktarda verinin işlenmesini gerektiren durumlar için geliştirilmiş olan uygulamalardır. Genelde bu uygulamalar karmaşık, ölçeklenebilir, dağıtık ve bileşen tabanlıdırlar. İş akışıyla ilgili bir görevi gerçekleştirmeyi hedeflemektedirler. Kurumsal uygulamalar birbirinden farklı platformlarda aynı anda çalışabilecek şekilde geliştirilmektedirler. Bu farklı platformlara örnekler olarak, bilgisayar ağları, Intranet'ler ve Internet üzerindeki farklı işletim sistemlerine, donanım ve yazılım özelliklerine sahip bilgisayarlar verilebilir.

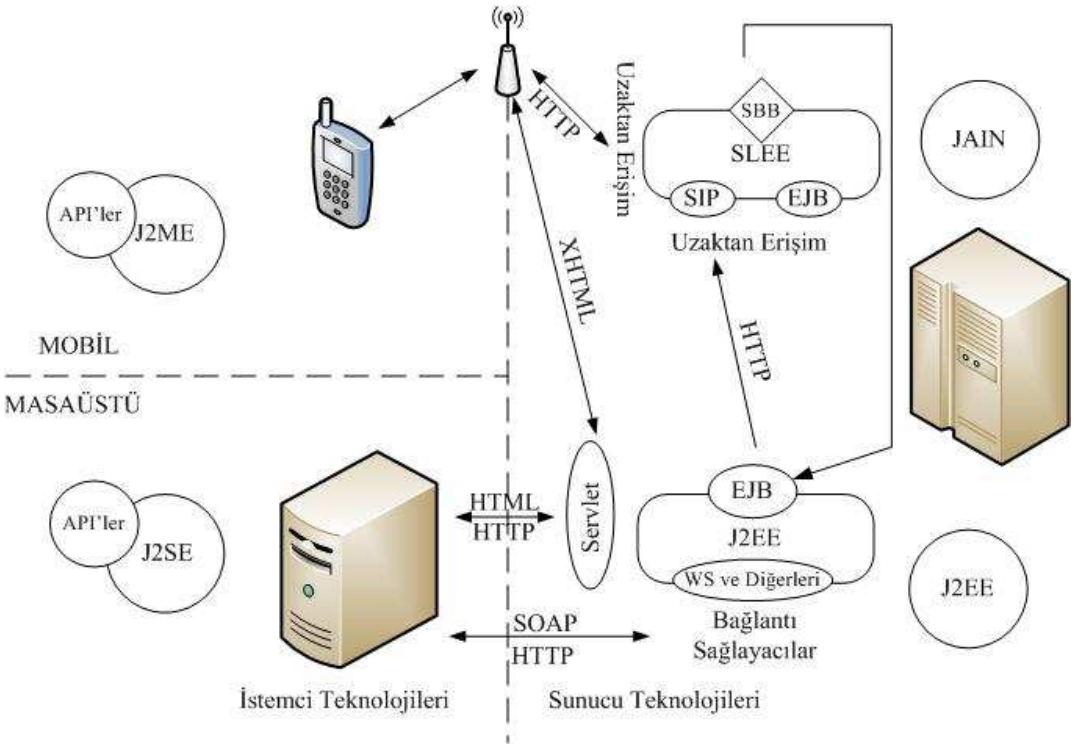
Kurumsal uygulamalar veriyi esas alan, kullanıcı dostu, güvenlik ölçütlerini karşılayabilecek nitelikte olan yazılımlardan meydana gelmektedir. Çok farklı gereksinimlere karşılık verebilirler. Kurumsal uygulamalar bir işyerinde insanların çok uzun sürelerde gerçekleştirebileceği karmaşık ve büyük miktarda veri içeren işlemleri kolaylaştırmayı hedeflemektedir.

Bu tezde seçilen kurum olan bir hastanedeki teşhis işlemlerinden elde edilen veriler aynı şekilde çok karmaşık ve yüksek miktarda veriler olacağından kurumsal bir uygulama geliştirme adımları takip edilmiştir.

3.2. Java Enterprise Edition (Java EE) ve Uygulama Modeli

Java Enterprise Edition yani Java Kurumsal Sürümü, kurumların özellikle büyük çaplı projelerde karşılaşacakları ihtiyaçlara karşılık verebilmek için geliştirilmiş, Java teknolojilerinin bütünün oluşturduğu standartların temel ismidir. Java EE servislerini sunan yazılımlar Java Uygulama Sunucusu (Java Application Server) olarak adlandırılır. Java EE'nin temel hedefi uygulama geliştiricilere güçlü uygulama programlama arayüzleri sunarak geliştirme süresini ve uygulama karmaşıklığını azaltmak ve uygulama performansını yükseltmektir. Java EE basitleştirilmiş bir programlama modeli kullanır. İsteğe bağlı olarak XML tabanlı tanımlayıcılar da eklenebilir [10].

Java EE uygulama modelinde birçok temel eleman bulunur. Java EE uygulama modeli Şekil 3.1'da görüldüğü gibidir [11].



Şekil 3.1. Java EE uygulama modeli.

İstemci teknolojileri masaüstü ve mobil olarak ikiye ayrılmaktadır. Masaüstü tarafında Java swing uygulamaları, web başlangıcı ve diğer uygulama programlama arayüzleri vardır. Mobil tarafında ise kablosuz mesajlaşma için uygulama programlama arayüzleri, diğer uygulama programlama arayüzleri, mobil aygıt bilgilendirme, Java mobil sürümü için sinyalleme protokolü ve kablosuz ağ vardır.

Sunucu teknolojilerinde ise Enterprise Java Bean, web başlangıcı bağlayıcısı, diğer bağlayıcılar, Servlet, servis çalıştırma ortamı ve servis çalıştırma ortamının etrafında sinyalleme protokolü bulunur. Servis inşa bloğu, karışık modlu çevirici ve JAIN yani akıllı ağlar için Java uygulama programlama arayüzleri de yine sunucu tarafında bulunan diğer elemanlardandır.

Java Bean, kurumsal Java uygulamalarında önemli bir yere sahiptir. Java Bean duruma göre fonksiyon veya metod gibi tanımlamaların yapılabileceği ve herhangi bir anda istenilen yerden çağırılıp programcının faydalanabileceği program parçalarıdır.

Java EE Sun firmasının kurumsal uygulama çözümüdür. Bu platform dağıtık sistem ortamında uygulama geliştirmek için meydana getirilmiş bir platformdur. Java EE yazılımlarında birçok yapı karmaşık bir biçimde içiçe çalışmaktadır. Veri depolama aracı olarak kullanılan veritabanı sunucuları ve veritabanları (JDBC, MYSQL, HSQLDB,

ORACLE), kullanıcıya arayüz ortamı sağlayan JSP web sayfaları, mesajlaşma sistemleri ve bağlantı sağlayıcılar gibi elemanlar birleşerek ana modeli oluşturmaktadırlar.

3.3. Windows Mobile İşletim Sistemi

Windows Mobile Microsoft firması tarafından cep bilgisayarları ve akıllı telefonlar gibi mobil cihazlar için tasarlanmış olan bir işletim sistemidir. Windows CE çekirdeği üzerine temellendirilmiştir [12].

3.3.1. Windows Mobile İşletim Sistemi İçin Uygulama Geliştirme

Mobil uygulama geliştirmek için birçok teknoloji ve yöntem mevcuttur. Bu tezde çalışılan problemler için Microsoft .Net teknolojileri kullanarak mobil uygulamalar geliştirilmiştir.

Windows Mobile işletim sisteminde uygulama geliştirebilmek için öncelikle Microsoft'un sitesinden Windows Mobile SDK (Software Development Kit)'i indirmek gerekmektedir. Windows Mobil işletim sistemlerine uygulama geliştirirken .Net Visual Studio derleyicisi yeterli olacaktır. Windows mobil işletim sisteminde .Net Framework kullanılmaktadır [12].

3.3.2. Windows Mobile Sürümleri

Windows Mobile işletim sisteminin Windows Mobile 2002, Windows Mobile 2003, Windows Mobile 2003 Second Edition (İkinci Sürüm), Windows Mobile 5.0, Windows Mobile 6 isimli sürümleri bulunmaktadır.

Windows Mobile 2003 Second Edition ile VGA (640×480) ve kare (240×240 ve 480×480), ekran desteği eklenmiştir. Windows Mobile 5.0 sürümü ile kalıcı hafıza desteği eklenmiştir. Windows Mobile 6 ise bir önceki sürüm temel alınarak hazırlanmıştır. Windows Mobile 6.1 ile oldukça kararlı bir hale gelmiştir. 6.5 ve 7.0 sürümleri de duyurulmuştur.

3.4. Tezde Kullanılan Yazılım Geliştirme Ortamları

3.4.1. Delphi

Tezin girişinde bahsedilen daha önceden gerçekleştirilmiş hastalık teşhis yazılımı, Delphi uygulaması "Teshis" programının geliştirilmesinde ve kodlarının yeniden düzenlenmesinde, Delphi yazılım dili ve Borland Delphi 7 IDE'si kullanılmıştır. Delphi

dilinin temelinde Pascal dili yatmaktadır. Delphi, Turbo Pascal dilinin bir nevi görsel bir sunumudur. Win32 ve .Net platformları üzerinde uygulama geliştirmeye yarar. Nesne tabanlı bir yazılım dilidir. Bu IDE güçlü bir grafik arayüz yapısına sahiptir ve geniş bir yazılım kütüphanesini uygulama geliştiricisine sunar [13].

3.4.2. NetBeans

Tezdeki Java web arayüzü yazılımlarını geliştirmek için seçilen derleme ortamı NetBeans'dir. Netbeans, çok geniş bir kullanıcı tabanı, büyüyen topluluğu, dünya çapında yüze yaklaşan ortakları olan başarılı bir açık kaynak kod projesidir [14]. Sun Microsystems, NetBeans açık kaynak kodlu projesini Haziran 2000 yılında başlatmış olup halen projelerin ana destekçisidir. Günümüzde iki ürün olarak bulunmaktadır: NetBeans IDE'si ve NetBeans Platformu. Her iki ürün de açık kodlu olup kar amaçlı olsun olmasın kullanılabilir. Kaynak kod CDDL (Common Development and Distribution License) koruması altındadır.

Netbeans IDE'si programcıların yazma, derleme, hata bulma ve yüklemelerini sağlayan bir araç olan geliştirme ortamıdır. Java ile yazılmış olmasına rağmen herhangi bir programlama dilini destekleyebilir. NetBeans IDE'sini genişletmek için çok sayıda modül bulunmaktadır. NetBeans IDE'si kullanımına ilişkin hiçbir sınırlaması olmayan ücretsiz bir üründür.

Ayrıca hali hazırda bulunan NetBeans Platformu, modüler ve genişletilebilir yapısıyla büyük masaüstü yazılımları üretmede kullanılır. Datasource katmanları için kullanılacak olan teknolojilerden Java DB (Java Veritabanı) Apache Derby Sun destekli, tamamen Java'da geliştirilmiş, açık kaynak kodlu olan ilişkisel bir veritabanıdır ve Apache 2.0 lisansı ile hakları korunmaktadır. Java, JDBC ve SQL standartlarına dayanır. PDA'lardan taşınabilir bilgisayarlara ve büyük bilgisayar sistemlerine kadar her yerde çalışır. Tüm özelliklerine rağmen boyutu sadece 2.5 Mb'dır.

3.4.3. MS .NET Ortamı, Visual Studio ve C#

Tezde mobil arayüzlerle ilgili çalışmalar tamamen .Net teknolojileri ile gerçekleştirilmiş olup meydana getirilen uygulama dosyası Windows Mobil işletim sisteminde çalıştırılabilir bir program dosyasıdır. Bu mobil uygulamayı derleyip geliştirirken .Net Visual Studio IDE'si kullanılmıştır. .Net Visual Studio içerisinde bulunan

ek bir özellik olan Smart Phone Programming, bu uygulamanın gerçekleştirilmesinde kullanılmıştır. Bu uygulama C# diliyle geliştirilmiştir.

Visual Studio, Asp.Net uygulamalarından Windows uygulamalarına, masaüstü uygulamalarından mobil uygulamalara kadar birçok uygulamayı yazılımcıya geliştirme imkanı sağlayan bir IDE'dir. Bünyesinde C++, C#, Visual J# dillerini barındırmaktadır. Görsel tasarımı kolaylaştıran arayüz araçlarını kullanıcıya sunar. Dolayısıyla kod odaklı geliştirmeden çok, iş odaklı geliştirme amacına hizmet verir. Ana sistem .Net Framework üzerine kuruludur. .Net framework Windows işletim sistemi için geliştirilmiş uygulamaların ve programların çalışabileceği bir platformdur [12]. Derlenen program .Net içerisinde bulunan bir emülator sayesinde herhangi bir mobil cihaz bağlı olmasa bile yerel makinede test edilebilir. Tezdeki görsel form tasarımları Visual Studio'nın içerisindeki araçlar kullanılarak yapılmıştır.

C# Microsoft tarafından üretilen nesne-yönelimli dillerin içerisinde en güçlü olanıdır. Uygulama geliştiricilere kısa zamanda büyük çaplı yazılımlar geliştirme imkanı sunmaktadır. Birçok hazır sınıf ve metod kütüphanesine sahiptir. Sun Microsystem'in Java teknolojileri temel alınarak, Java benzeri yapılara ve hedeflere sahip olarak Microsoft tarafından geliştirilmiştir. .Net'de derlenen kodlar, Java'daki bytecode benzeri, Microsoft Intermediate Language adı verilen bir orta seviyeli byte koduna dönüştürülmektedir [15].

3.5. Yazılım Performans Ölçme Ortamları

Yazılımlar geliştirildikten sonra birçok alanda performans analizi yapmak, yükleme anındaki durumları test etmek ve aşırı yüklenme durumlarında oluşacak olayları gözlemlemek gerekmektedir. Bu durumlara bazı örnekler olarak, hafıza kullanımı, işlemciye uygulanan yük, ağın üstünde meydana gelen yük, kullanıcıların isteklerine uygulamanın ve sunucunun vereceği cevabın sürelerinin ölçümü verilebilir.

Yazılım geliştirme, sadece kodu yazıp derleyip ve düzgün bir şekilde çalıştırmanın yanında, yazılım performans testleri konularını da içerir. Yazılım performansını test etmek için geliştirilmiş birçok performans analiz aracı mevcuttur. Bu yazılımlar farklı platformlar, işletim sistemleri ve amaçlar için farklı hizmetler sunmaktadır. Tezdeki yazılımların performans testleri için incelenen yazılımlar aşağıda kısaca sunulacaktır.

3.5.1. Apache JMeter

Fonksiyonel davranışları ve performansı ölçmek için geliştirilmiş tam anlamıyla bir Java uygulamasıdır. Asıl olarak web uygulamalarını test etmek için geliştirilmiştir, fakat daha sonra çalışma alanı başka fonksiyonların da testlerine doğru genişletilmiştir. Apache JMeter hem dinamik hem de statik kaynakları değerlendirmek ve test etmek için kullanılabilir. Test edilebilen kaynaklara örnekler olarak, dosyalar, servlet'lar, perl scriptleri, java nesneleri, veritabanları ve sorgular, ftp sunucuları verilebilir. Bir sunucuya aşırı yüklenme durumlarında neler olabileceğinin simülasyonunu yapma yeteneğine sahiptir. Yine farklı ortamlar için de yüklenme durumu ve performans analizi yapabilmektedir. Grafik analiz tekniklerine sahiptir ve farklı ortamlar için aşırı yüklenme durumlarındaki performansları test edebilmektedir [16].

3.5.2. CLIF

Clif modüler, esnek ve dağıtık sistem mimarisine uygun bir yük test platformudur. Herhangi bir Java programından ulaşılabilen her sistem için uyarlanabilir. Clif üç farklı kullanıcı arayüzü seçeneği sunar: Swing, Eclipse GUI ve komut satırı. Tüm bunları dağıtık bir sistemdeki hafıza kullanımı, işlemci kullanımı ve diğer performans kriterleriyle ilgili sorunları çözmek için kullanmaktadır. Bir Eclipse sihirbazını kullanarak yeni protokolleri de destekleme şansına sahiptir. Yük senaryoları XML dosyaları aracılığıyla, bir kullanıcı arayüzü aracılığıyla veya yakalama aracı ile yapılabilir. Senaryo motoru sisteme milyonlarca kullanıcı varmış gibi giriş yapıldığını varsayarak test yapabilme yeteneğine sahiptir [17].

3.5.3. Grinder

Grinder bir Java yük testi framework'üdür. Herhangi bir script'in birçok farklı makinede nasıl davranacağını, nasıl işlem göstereceğini grafiklere dökerek çözüm önerisi getirir [17].

3.5.4. Test Maker

Web tabanlı uygulamalar için ölçülebilirlik, fonksiyonellik ve performans gibi zengin bir alanda test yapabilme imkanı sağlayan bir uygulamadır. Kullanıcı dostu bir grafik arayüzle beraber gelmektedir. Nesne tabanlı bir dil olan Pyhton ile akıllı test nesneleri oluşturmaya imkan sağlar. HTTP, HTTPS, SOAP, XML-RPC, SMTP, POP3, IMAP gibi

protokolleri desteklemektedir. Birim testlerinden shell uygulamalarına kadar geniş bir yelpazede test yeteneklerine sahiptir [17].

3.5.5. Netbeans Profiler

Netbeans Profiler Netbeans IDE'sine isteğe bağlı olarak eklenebilen bir uygulamadır. Netbeans Profiler aracı uygulamanın çalışma zamanındaki performans kriterlerini belirlemekte kullanılan güçlü bir eklentidir. İşlem durumlarını, işlemci performansını, hafıza kullanımını gibi kriterleri ölçer. Web uygulamalarından Java uygulamalarına kadar geniş bir yelpazede test yapabilme yeteneğine sahiptir. Performans analizi yapılacak olan uygulamanın uzak bir masaüstünde ya da yerel makinede çalışması Netbeans Profiler için sorun oluşturmaz [17].

Tezde geliştirilen uygulamaların performans testi için Netbeans Profiler tercih edilmiştir. Bu uygulamanın tercih edilmesinin en büyük sebeplerinden birisi gerçekleştirilmesi düşünülen prototip uygulamanın Netbeans ortamında geliştirilmesidir. Netbeans Profiler Netbeans IDE'sine eklenen bir plug-in olduğundan dolayı sorunsuz bir şekilde performans analizi yapabilmektedir. Bu plug-in birçok uygulama geliştirici tarafından kullanılan bir performans analiz aracıdır. Ayrıca sadece web yük testi tezde geliştirilen uygulamanın performansını ölçmek için yeterli olmayacaktır. Sanal makine performansı, uygulama performansı, sunucu makinesi performansı gibi birçok elemanın birbirine paralel olarak test edilmesi gerekmektedir. Bu da yine Netbeans Profiler'ın diğer test araçlarının yerine tercih edilmesinin bir sebebidir.

3.5.6. CLR Profiler

CLR Profiler bir işlemin yığınsal durumunu gözlemlemeye ve çöp toplayıcısının (garbage collector) durumunu izlemeye yarar. Hafıza kullanımı, işlemci performansı gibi birçok anahtar faktör konusunda bilgi sahibi olarak analizler yapma imkanına sahip olunur. Dolayısıyla kodun daha etkin bir şekilde çalıştırılarak uygulamanın daha verimli olması için yapılan çalışmaya büyük katkı sağlar. Bu tezde uygulamanın performansını analiz ederken doğrudan CLR programından yararlanılmamıştır. CLR aracı bir kütüphane olarak analize dahil edilmiştir. Bu CLR Profiler uygulamasından veri çekmek ve görüntülemek için ek bir uygulama olan .NET Compact Framework Remote Performance Monitor uygulaması kullanılmıştır.

3.5.7. .NET Compact Framework Remote Performance Monitor

Bu test aracı .NET Compact Framework Versiyon 2.0 Service Pack 1 ile birlikte gelen bir analiz aracıdır. Bu araç kullanıcıya çalışmakta olan uygulamanın performans durumu ile ilgili önemli bilgiler vermeye ve gözlemler aktarmaya yarar. Uygulama .NET Compact Framework Remote Performance Monitor içerisinden başlatıldığı anda bu performans aracı doğrudan gerekli veriyi CLR kütüphanesinden alarak gözleme sunar. NET Compact Framework Remote Performance Monitor verisini PerfMon uygulaması ile birlikte kullanarak gerçek zamanlı grafikler ve performans sayaçları elde etmek mümkündür [12].

Remote Performance Monitor yazılımı aşağıdakileri yapabilme imkanı sunar:

a. Performans sayaç verilerini uygulama çalışırken izleyebilmek.
b. PerfMon.exe uygulaması aracılığıyla performans sayaç verilerini grafiksel olarak izleyebilmek.
c. Oturumları kaydederek farklı ortamlarda tekrar gözden geçirebilme imkanı sunmak.
d. Log dosyası yani kayıt defteri alarak gerçekleşmiş olayların kaydını tutmak [12].

3.5.8. Performans Analizi İçin Geliştirebilecek Kişisel Uygulamalar

Performansı test etmek için farklı kişiler veya grupların geliştirdiği paket yazılımlar yerine kişisel olarak geliştirilen yöntemlere gidilebilir. Bu konuyu daha detaylı olarak açmak gerekirse, C# dili temel alınarak derlenen bir uygulama için programın çalışma anlarında geçen süreleri ölçme yöntemi düşünülebilir. Bu yöntemle programın farklı ortamlarda çalışırken harcadığı süre ile bir performans karşılaştırma yöntemine gidilebilir. Bu yöntem uygulama geliştiricisine derlediği programın optimizasyonunda kontrol sağlar.

C# dilinde, hazır bir sistem sınıf olan System.Diagnostic.Stopwatch, bu sınıf kullanılarak herhangi bir anda bir zamanlayıcı başlatılabilir ve sonlandırılabilir. Bu yöntemin altında bir Windows API'si olan "QueryPerformanceTimer" çağırılmaktadır. "QueryPerformanceTimer" yüksek hızlı bir zaman sayacıdır ve basit Timer sınıflarından çok daha hassas değerlere sahiptir.

Bu kod hızı ölçüm metodu program geliştirme açısından, temsilci (delegate) kullanımıyla beraber istenilen metodlara ve program parçacıklarına adapte edilebilir. Bu yöntem kullanılarak geliştirilen bir sınıf aracılığıyla derlenilen uygulamaların çalışma anlarında işlemcilerde harcadıkları süreler başarı ile karşılaştırılabilir. Elde edilen sonuç değerleri kolaylıkla işlenebilir veriler olacağından kolay ve hızlı bir performans analizi imkanı sunmaktadır.

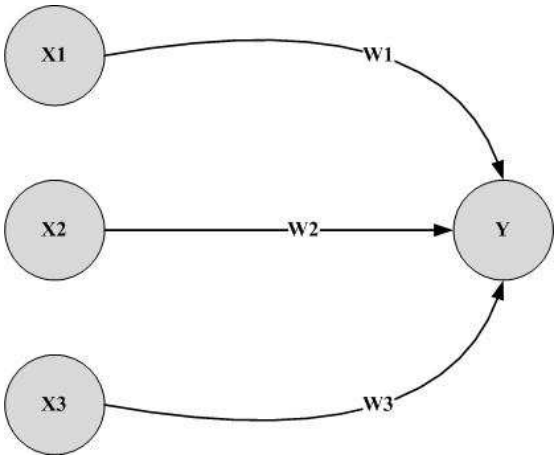
Bu tarz bir yöntem, herhangi bir uygulamanın farklı donanım ortamlarında çalışırken göstereceği farklılıkları somut olarak incelemeye yarar [18].

3.6. Yapay Sinir Ağları

Bu tezde geliştirilen teşhis uygulamalarının alt yapısında yapay sinir ağları modelleriyle geliştirilmiş programlar bulunmaktadır. Bundan dolayı bu bölümde yapay sinir ağlarının bu tezle bağlantılı olan kapsamlarına kısaca genel bir giriş yapılmaktadır.

İnsan beyni, gördüğü olayları çözümleme ve öğrenme yeteneğine sahip karmaşık bir sisteme sahiptir. Bu karmaşık sistem biyolojik sinir ağlarından ve belirli elemanlardan meydana gelmektedir. Yapay sinir ağları ise biyolojik sinir ağları ile birçok ortak özelliğe sahip ve biyolojik sinir ağlarında olduğu gibi bilgiyi işleme amacıyla kurulan sistemlerdir. Yapay sinir ağları biyolojik sinir sistemlerinin matematiksel modellerini örnek almaktadırlar.

Bilgi işleme olayı nöron (sinir) adı verilen elemanlarda meydana gelmektedir. Sinyaller, nöronlar arasında bağlantı yollarıyla transfer edilir. Her bağlantının ise bir ağırlığı vardır ve iletilen sinyali çoğaltmakla görevlidir [19].



Şekil 3.2. Basit bir yapay sinir ağı nöronu.

Şekil 3.2'de *X1, X2* ve *X3* ağın girişlerini, *W1*,*W2* ve *W3* ağırlıkları ve Y ağın çıkışını temsil etmektedir. Yapay sinir ağları, geleneksel programlama yöntemlerine oranla çok daha az kod yazarak problemlerin çözülmelerine imkan vermektedir. Yapay sinir ağları belirli bir düzene göre akış gösteren ve seriler halinde devam eden olaylardan meydana

gelen problemlerin çözümünde kullanılmak yerine, desenleri bulma, sınıflandırma ve tahmin gibi belirli bir tarzda ilerlemeyen problemlerin analizi için kullanılmaktadır [20].

Bu açıklamadan yola çıkarak, her hastanın kendisine göre farklı test sonuçları olacağından ve bu sonuçlar kişiden kişiye değişim göstereceğinden düzensiz bir dizilim meydana gelecektir. Dolayısıyla hastalık teşhisi probleminin yapay sinir ağları ile modellenmesi çok uygun olmaktadır.

Yapay sinir ağlarının belirli bir problem alanında gerçekleşen olayları çözümleyebilmesi ve anlamlandırabilmesi için düzgün ve doğru bir şekilde eğitilmesi gerekmektedir. Yapay sinir ağlarının eğitim işleminin gerçekleştirilebilmesi için düşünülmüş farklı algoritmalar mevcuttur. Bu algoritmalar geri yayınım, radyal tabanlı fonksiyon, öğrenme vektörü niceleme algoritmaları olarak adlandırılmışlardır. Bu tezde kullanılan YSA eğitim algoritması geri yayınım algoritmasıdır. Çünkü geri yayınım algoritması ileri beslemeli yapay sinir ağlarını eğitmek için kullanılan algoritmalar arasında en yaygın kullanılanıdır [21].

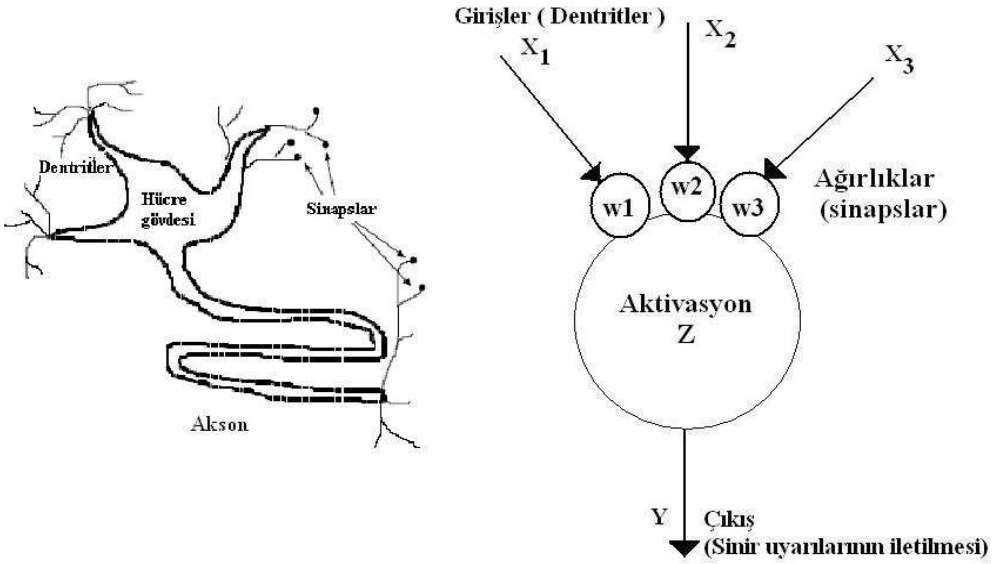
Geri yayınım algoritmasının diğer algoritmalara göre daha çok tercih edilmesinin sebeplerinin altında yatan gerçeklerden birisi basit ve güçlü olmasıdır. Gerçek hayata uyarlanabilecek ve lineer olmayan ağların eğitiminde kullanılabilir. Ayrıca algoritma sadece birkaç satır koda dökülebilecek kadar kolay uygulanabilir bir seviyededir [21]. Bu gibi avantajlarından dolayı bu tezde yapay sinir ağını eğitmek için geri yayınım algoritmasından yararlanma yoluna gidilmiştir.

3.7. Yapay Sinir Ağları ile Hastalık Teşhisi

Tam anlamda doğru bir teşhisi gerçekleştirebilmek için sağlık konusuyla ilgilenen bir bilim adamının uzun dönemli tecrübeler edinmesine ve birçok konuda öğrenim görmesine gerek vardır. Bunun yanısıra öğrenilen medikal bilgiler ve veriler kısa zamanda değişim gösterebilmektedirler. Bir bilim adamının bu değişimleri sürekli takip etmesi, her değişimde yeni konular hakkında araştırma yapması ve öğrenme yoluna gitmesi şarttır. Bir teşhisin geçerli sayılabilmesi için sağlık konusunda kendini yetiştiren bir bilim adamının kariyerinde uzun bir yol katetmiş olması gerekmektedir. Yani bu bilim adamının mesleki açıdan tecrübeli ve kendini kanıtlamış olması beklenmektedir. Ancak bazen tıp dünyasında yeni baş gösteren hastalıklarda tecrübeli doktorlar bile hatalar yapabilmekte ve teşhis koyma konusunda problemlerle karşılaşabilmektedir. Özet olarak insanlar tanımlama işlemini gerçekleştirebilir fakat bunu bilgisayarlar kadar başarılı bir şekilde yerine

getiremezler. İnsanlar nesneleri ve desenleri kolaylıkla tanıyabilir fakat olasılıkları incelemeye dökerken bilgisayarlar kadar hatasız olamazlar. Buradan yola çıkarak hastalık teşhisi gibi hata oranının çok düşük hatta sıfır oranında olması gerektiği bir alanda, bilgisayarların analiz ve örüntü tanımlama yeteneklerinden faydalanmak çok doğru ve yerinde bir karar olacaktır.

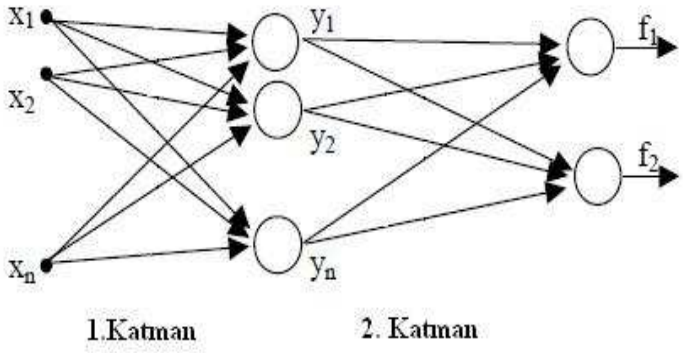
Doğal sinir ağlarında olduğu gibi yapay sinir ağları da birçok küçük arabirimden meydana gelmektedirler. Birbirleri arasında bağlantılıdırlar ve hep birlikte çalışmaktadırlar. Şekil 3.3'te görüldüğü gibi her nöronda birçok giriş vardır fakat sadece bir çıkış bulunmaktadır [22].



Şekil 3.3. Gerçek ve yapay sinir modelleri.

YSA modelinde gerçek sinirdekine benzer olarak girişler vardır ve bu girişler ağırlıklara bağlanmışlardır. Her ağırlık için *W* ve her giriş için *X* sembolü kullanılmıştır. *Y* ise çıkışı temsil etmektedir. Burada aktivasyon 1'den *n*'e kadar olan tüm ağırlıkların ve girişlerin çarpımlarının toplamı olacaktır.

Eğer farklı birkaç nöron paralel olarak farklı katmanlarda birbirine bağlanırsa girişten çıkışa doğru olan bir adresleme meydana gelecektir. Bu olay ileri beslemeli ağ modelini oluşturmaktadır. Bu tezde kullanılan YSA modeli ileri beslemeli bir yapay sinir ağı modelidir. İleri beslemeli bir yapay sinir ağı örneği Şekil 3.4'te verilmektedir [22].



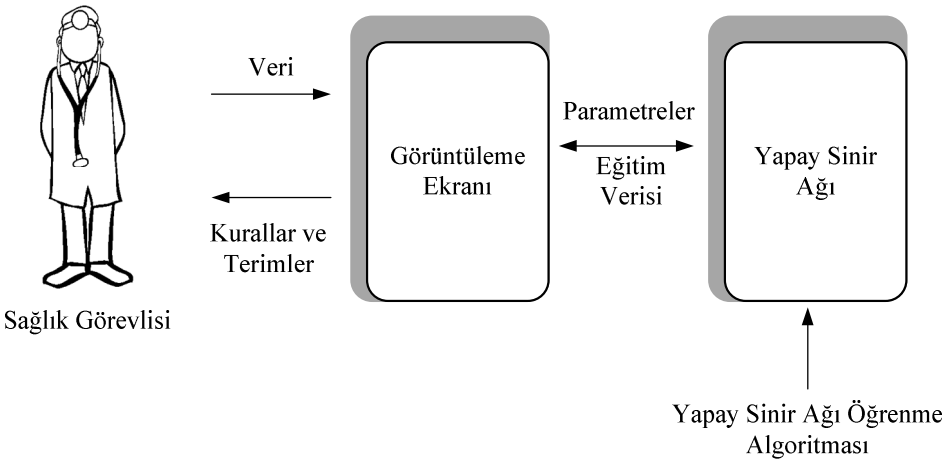
Şekil 3.4. Örnek bir ileri beslemeli yapay sinir ağı modeli.

Şekil 3.4 görülen YSA modeline göre *X1, X2, …, Xn* 1. katmandaki girişleri *Y1, Y2, …., Yn* 2. katmandaki girişleri ve *f1, f2* ağın çıkışlarını temsil etmektedir. Burada 1. katmandaki *X1* girişi sırasıyla 2. katmandaki *Y1,Y2, …, Yn* ile çarprazlanmış ve aynı işlem diğer *X* girişleri için de gerçekleştirilmiştir. Benzer şekilde *Y* girişleri de *f* çıkışları ile eşlenerek sistemin son çıkış değerleri elde edilmektedir.

Bu tezde hastalık teşhisiyle ilgili sağlık verileri toplandıktan ve derlendikten sonra, bu yapay sinir ağına benzer, ilgili verilerin modellenmesi için uygun yeni bir yapay sinir ağı tasarlanır ve teşhis yapacak uygulamanın programlanması fazına geçilir.

Teşhis yapılmadan önce gerçekleştirilecek adımlardan birisi de, verinin ön işlemlerden geçirilmesidir. Veriler hakkında bazı zorluklarla karşılaşılabilineceği bilinmelidir. Bu zorluklara örnek olarak veri kümesinin çok küçük olması verilebilir. Küçük veri kümelerinden elde edilecek sonuçlar güvenilir olmayacaktır. Bundan ayrı olarak, sağlık verileri homojen olmayabilir. Dolayısıyla tüm bu verilerin bir normalizasyon işleminden geçirilerek belirli bir standardı yakalamaları gerekmektedir.

Yapay sinir ağlarını kullanarak hastalık teşhisi yapmadaki son faz sistemin eğitilmesi ve test edilmesidir. Tüm yukarıdaki anlatılan işlemlerin genel bir krokisi Şekil 3.5'te verilmiştir. Şekilde sağlık çalışanının uyguladığı fiziksel testten sonra elde ettiği sağlık verileri, bir arayüz aracılığıyla sisteme girilir. Bu verilerin girişini yaparken hastalıkla ilgili kurallara ve terimlere dikkat edilir. Buradaki veri girişleri bir arayüz ekranı, yani görüntüleme ekranı aracılığı ile gözden geçirilir ve daha sonra girişler yapay sinir ağına gönderilir. Yapay sinir ağına gönderilen veriler test işlemini gerçekleştirmek için yapay sinir ağları öğrenme algoritmasına aktarılır. Yapay sinir ağı eğitim algoritmasında eğitilen veri test işlemi için hazır hale getirilecektir.



Şekil 3.5. Hastalık teşhisi sırasında gerçekleşen olayların genel modeli.

Buradaki mimari farklı uygulamalara aktarılarak değişik hastalıkların teşhisinde aynı çalışma tekniğiyle kullanılabilir. Bu tezde aynı model mobil ortamlarda çalışan uygulamalar için de tasarlanmış ve uygulamaya konulmuştur.

Farklı yapay sinir ağı yapıları, tıp alanında karar destek sistemlerinin gelişmesinde kullanılmakta olan önemli modellerdendir. Sağlık ve tıp alanında bir hastalığın teşhisi konusunda karara sahip olabilmek için verilerin derin ve doğru bir şekilde analiz edilmesi gerekmektedir. Doktorlar karar verme aşamasında bu verilerle ilgili bileşenleri incelemektedirler. Bu incelemeyi istatiksel çalışmalarla gerçekleştirmektedirler. Bu veriler bazen çok büyük ve geniş çaplı olabileceğinden bilgisayarlar bu anda devreye girebilirler. Hastalık teşhisleri veri sınıflama işlemi olarak incelenebilmektedir.

4. İÇ HASTALIKLARI İÇİN WEB ARAYÜZÜ

4.1. Giriş

Bu tez çalışmasında bir hastanede var olduğu varsayılan iç hastalıkları teşhis eden eski bir yazılımın web arayüzüne kavuşması için Java EE ortamında geliştirilen bir sistem tasarlanmıştır. Bu sistemin alt yapısında farklı eski bir teknolojiyle oluşturulmuş, diğer uygulamaların temel motoru olarak çalışan bir yapay sinir ağları uygulaması mevcuttur. Bu motor yapay sinir ağlarını kullanarak birçok hastalığın teşhisi için gerekli ortamı sunmaktadır. Bu hastalık teşhis motorunu modern web ortamına sunmak için Java Server Faces teknolojisi kullanılmıştır.

Günümüzde akıllı telefon ve cep bilgisayarı gibi mobil cihazlardaki gelişmeler sonucu, bu cihazlar zaman alıcı eskiden mümkün olmayan matematiksel hesaplamaları yapabilecek duruma gelmiştir. Bu sayede sunucu tarafındaki yükü de azaltma imkanı doğmuştur. Mesela kullanıcı yani istemci tarafında büyük veri girişleri gerektirmeyen ve kısa süreli işlemler için sunucuya çok fazla ihtiyaç duymadan istemci tarafında yerel işlem yapmak, daha büyük çapta veri girişi olacak durumlar için sunucu ağırlıklı bir istemci-sunucu hesaplama mimarisi geliştirmek, sistem kaynaklarını dengeli bir biçimde kullanmak, daha kaliteli ve hızlı hizmet verme imkanı sunacaktır.

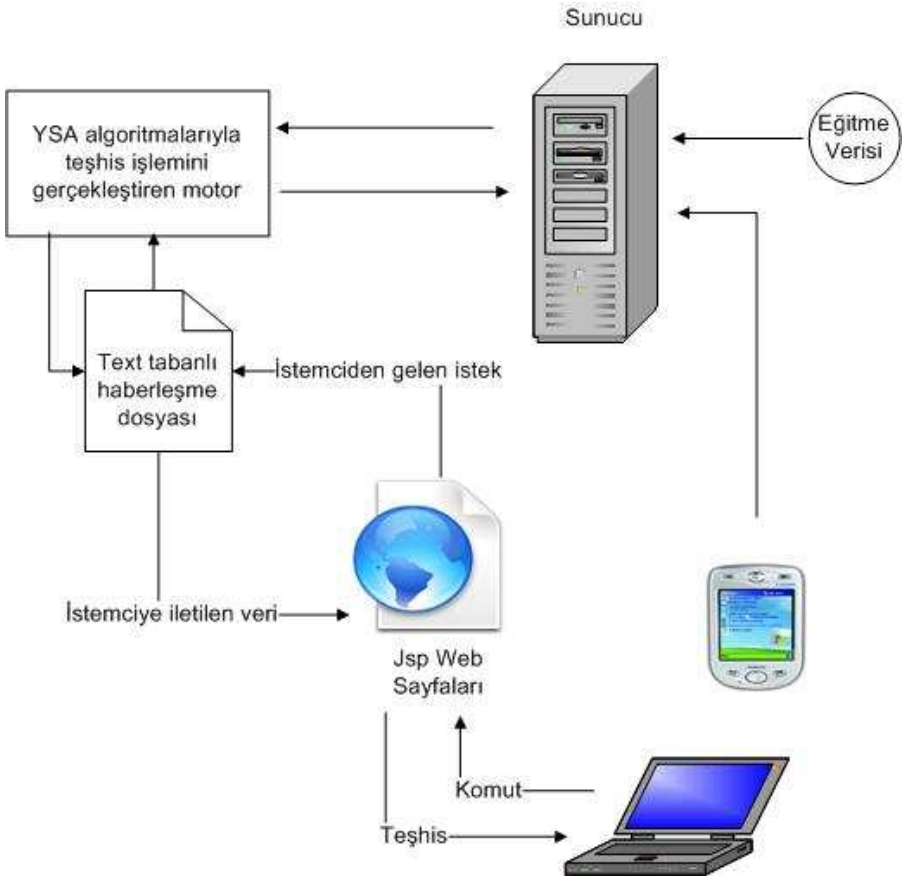
Kullanıcı web sitesinden telefonuna ya da benzeri mobil cihazına indireceği uygulama ile yerel olarak gerekli işlemi gerçekleştirecek ve sunucuya sadece bazı sonuçları döndürecektir. Sunucu tarafında ise bu veri girişlerinin bir veritabanı aracılığıyla kaydı tutulup bir sonraki işlemlerle daha önceki girişleri karşılaştırarak sistemin öğreniminin daha da geliştirilmesi amaçlanmaktadır. Kullanılan YSA motoru ile Tablo 4.1'de görüldüğü gibi birçok hastalığının teşhisi yapılabilmektedir. Hastalıkların teşhisi için geliştirilen programlarda Geri Yayınım (Back Propagation) eğitim algoritması kullanılmaktadır.

Tablo 4.1. Tezde kullanılan YSA programının teşhis edebildiği hastalıklar.

Teşhis Edilen Hastalık	**Açıklama**
Akut Böbrek Yetmezliği (A.K.I)	Hastalık, enfeksiyon veya sakatlık gibi durumlar böbreklere hasar verdikleri zaman ortaya çıkmaktadır. Geçici olarak böbrekler, yararsız sıvıları ve atıkları atamaz ve kan akışından uygun miktarda böbrek ayarlaması sağlanmış kimyasalları elde edemez.
Demir Eksikliği (Iron Deficiency)	Beslenme eksikliklerine bağlı olarak meydana gelmektedir. Çabuk gelişir ve vücudun yüksek miktarda demir kaybetmesine sebep olur.
Delta Hepatit (Delta Hepatitis)	Sadece hepatit B virüsü bulaşan hastalarda oluşan bir karaciğer iltihaplanması şeklidir.
Hepatit A (Hepatitis A)	Hepatit A bulaşması yüzünden meydana gelen bir karaciğer iltihaplanması şeklidir. İçme suyuna hasta bir insanın dışkısından virüs bulaştığı zaman salgınlara neden olabilir.
Hepatit B (Hepatitis B)	Hepatit B virüsü bulaşması yüzünden meydana gelen karaciğer iltihaplanmasının potansiyel olarak ciddi bir şeklidir.
Hepatit C (Hepatitis C)	Uzun süren bir kronik hastalığa sebep olan karaciğer iltihaplanmasıdır. Genellikle Hepatit C virüslü kanla temas halinde bulaşmaktadır. Hepatit A ve Hepatit B virüsüne bağlı olmadan birçok viral karaciğer iltihaplanmasına sebep olur.
Hepatit E (Hepatitis E)	Hepatitin bağırsak yoluyla taşınan virüsler tarafından bulaşması durumunda oluşan bir hastalık türüdür. Hepatit A virüsü yüzünden meydana gelmez. Genellikle kirli içme suları tarafından yayılmaktadır.
Hipertansiyon (Hypertension)	Yüksek kan basıncıdır. Kan basıncı, kanın damar çepherlerini akış anındaki itme gücüdür.
Hipotansiyon (Hypotension)	Düşük kan basıncıdır. Kanın beyine ulaşamamasına sebep olacak derecede düşük kan basıncı olduğu durumlarda meydana gelmektedir. Baş dönmesi ve bayılmalara sebep olmaktadır.
Hiper Tiroid (Hyper Thyroidy)	Tiroid hormonlarının aşırı faal bir tiroid bezi tarafından aşırı derecede üretilmesi sebebiyle meydana gelen bir hastalıktır.
Hipo Tiroid (Hypo Thyroidy)	Hipo tiroid, tiroid bezi çökünce vücudun ihtiyacı olan tiroksini üretemediği ve salgılayamadığı zaman oluşan bir hastalıktır.
Kalp Embolisi (Heart Embolism)	Tıkanıklık bir kan pıhtısının, bir dokunun ya da timörün, gaz balonunun veya vücuda yabancı bir maddenin kanda dolaşırken herhangi bir damarı tıkamasıyla meydana gelen bir hastalıktır. Bu tıkanıklık kalple ilgili damarlardan birisinde de gerçekleşebilir.
Üst Sindirim Sistemi Kanaması (Gastric Bleeding of Upper Part)	Sindirim sistemi yolunda başlayan kanamadır, ağızdan mideye kadar uzanan boruyu, mideyi ve ince bağırsağı da içermektedir.
Alt Sindirim Sistemi Kanaması (Gastric Bleeding Of Bottom Part)	Bu kanama şekli sindirim sistemi yolunun alt kısmı ince bağırsağın büyük bir kısmını, kalın bağırsağı veya iç kısımları, rectum ve anüsü içermektedir.
Pankreas İltihabı (Pankreas Inflammation)	Bu hastalığın iki ana sebebi, safra taşı oluşumu ve alkolizmdir. Alkolizm sebebiyle gelişen cinsi, uzun süreli olarak alkollü içeceklerin alımıyla gerçekleşmektedir. Safra taşlarının safra kesesinde oluşması pakreatik kanalda tıkanmaya sebep olabilir. Bu bölge sindirilebilir suların toplanmasıyla sorumludur ve burada oluşan bu çeşit bir sorun pankreas iltihaplanmasına sebep olur.
Şeker Hastalığı (Diabetes)	Pankreasın yeterli derecede insülin salgılayamadığı veya hücrelerin üretilen insüline tepki vermediği durumlarda kandaki glukozun hücreler tarafından emilememesiyle meydana gelen hastalıktır.

4.2. Sistemin Genel Görünüşü

İlk olarak bir kullanıcı sisteme bağlandığında mevcut iç hastalıkları teşhis uygulaması yani "YSA motoru" sunucudan aldığı komutla işlem yapmaya hazır duruma gelmektedir. Daha sonra sunucu tarafından sunulan web arayüzünde kullanıcının mobil ya da PC erişim seçeneklerinden hangisini kullandığı tespit edilir. Mobil bir cihazla sisteme bağlanan istemci teşhis sonucunu mobil cihazda çalışabilecek bir yazılım aracılığı ile öğrendikten sonra bu sonucu web arayüzü üzerinden tekrar sunucuya iletir. Sistemin genel yapısı Şekil 4.1'de görülmektedir.



Şekil 4.1. Sistemin mimari görünüşü.

Sunucuda YSA algoritmalarının çalıştığı Delphi uygulaması ile web arayüzünün iletişimi için ASCII metin dosyaları aracı elemanlar olarak düşünülmüştür. İki teknoloji tamamen birbirinden farklı olduğu için böyle bir yöntem tercih edilmiştir. Sunucu tarafında çalışan YSA motoru Delphi ile yazılmış bir çalıştırılabilir komut dosyasıdır. Dolayısıyla web arayüzünü bir çalıştırılabilir komut dosyasına bağlamak bir iletişim problemi doğurmuştur. Bu problem iki uygulama arasında bir köprü görevi üstlenecek ASCII metin

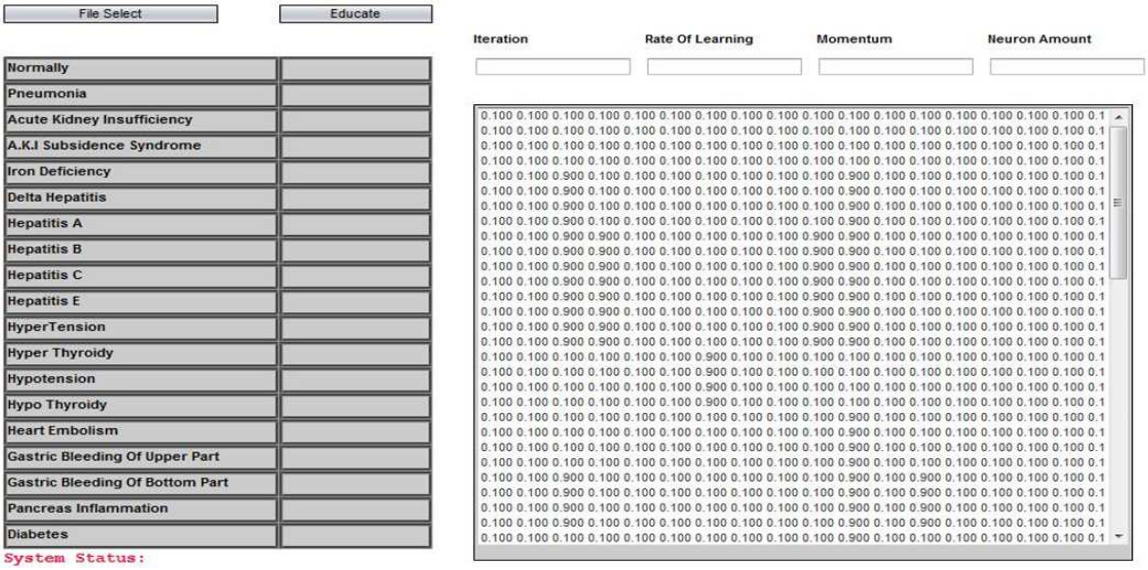
bazlı haberleşme yöntemi ile çözülmüştür. Bu haberleşme için ASCII metin dosyalarının tercih edilme sebebi ise byte düzeyinde iletişim kurmak zorunda kalınmasıdır. Bu haberleşme dosyasında, web arayüzünden girilen Iterasyon, Öğrenme Oranı, Momentum ve Nöron miktarı değerleri bu metin dosyasına yazılır.

Web arayüzünden gelen komut, metin dosyasına yazılır ve çalıştırılabilir bir program dosyası olan Teşhis.exe programı bu komutu alır. İşlemi gerçekleştirdikten sonra sonucu tekrar bu metin dosyasına yazar ve tüm teşhis sonuçlarını yine ayrı ayrı raporlar halinde farklı metin dosyalarına kaydeder. Kullanıcı herhangi bir veriyi test etmek istediğinde hazırlanan bu rapor dosyaları web arayüzü aracılığıyla kullanıcıya sunulur. Dolayısıyla web arayüzünden tüm olay akışını izleme imkanı sağlanmış olur.

Sisteme bir kullanıcı adı ve şifreyle girilmektedir. Veritabanında kayıtlı tutulan tüm kullanıcılar için uzman ya da normal kullanıcı olmak üzere iki tip belirlenmiştir. Uzman ve normal kullanıcılar için iki farklı ekran olacaktır. Bunun sebebi uzman olmayan bir kişinin çok karmaşık verileri giremeyeceği ve sorun yaşama riskinin olmasıdır. Dolayısıyla normal kullanıcılar için daha kullanıcı dostu bir ekran olmalı ve olabildiğince doğrudan sonuca ulaşmaları sağlanmalıdır. Uzman kişiye sunulan grafik arayüz uygulaması Şekil 4.2'deki gibidir. Bu arayüzün temeli daha önce bahsi geçen Delphi programından gelmektedir [1]. Bu program sadece sunucuda çalışmaktadır.

Alanda uzman olan kullanıcı İterasyon, Öğrenme Oranı, Momentum ve Nöron sayısı değerlerini girdikten sonra, Educate tuşuna basınca YSA motoru sunucuda aktif hale gelir. Geri Yayınım (Back Propagation) algoritmasını kullanarak eğitme işlemini bitirir ve raporların oluşturulmasını tamamlar.

YSA motoru sunucuda teşhis işlemini gerçekleştikten sonra Java kodundan gelen bir komut ile kendisini sonlandırmaktadır. Bunun sebebi geçici hafızayı ve işlemciyi gereksiz yere işgal etmemesidir. Dolayısıyla program çalışma anında sadece arka planda çalışacak ve işi bitince kendisini sonlandıracaktır.

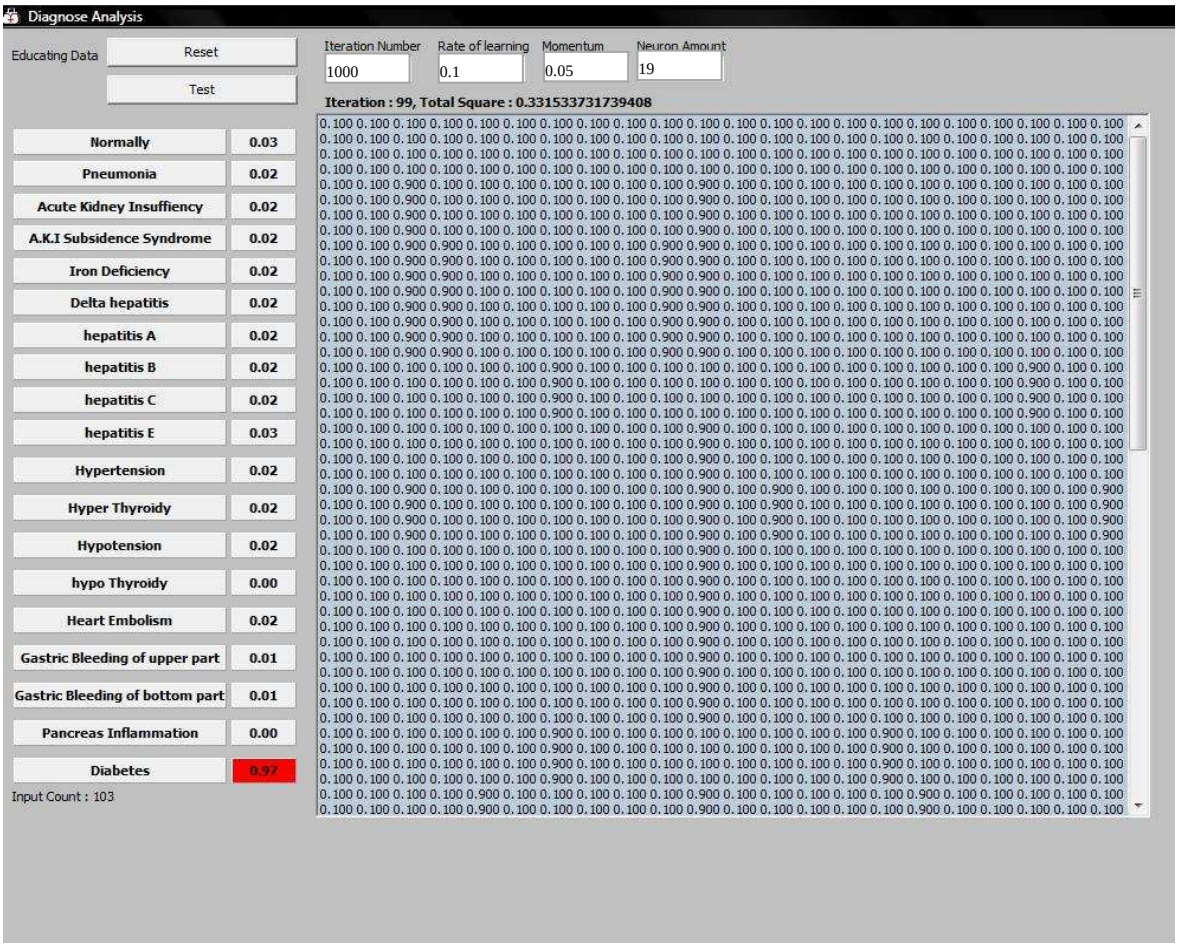


Şekil 4.2. İç hastalıkların teşhisinde kullanılan uzman kullanıcı için tasarlanan web arayüzü.

Delphi programında kullanılan Geri Yayınım algoritmasında gerçekleşen işler aşağıdaki gibidir:

- Eğitilecek giriş verilerinin verilmesi,
- Çıkışların hesaplanması,
- Son katman için beklenen değerler ve çıkışlar karşılaştırılarak deltaların bulunması,
- Gizli katmanlar için deltaların bulunması,
- Ağırlıkların yenilenmesi.

Tezimizde YSA motoru olarak kullanılan ve Delphi programlama dili ile geliştirilmiş programın arayüzü Şekil 4.3'te verilmiştir.

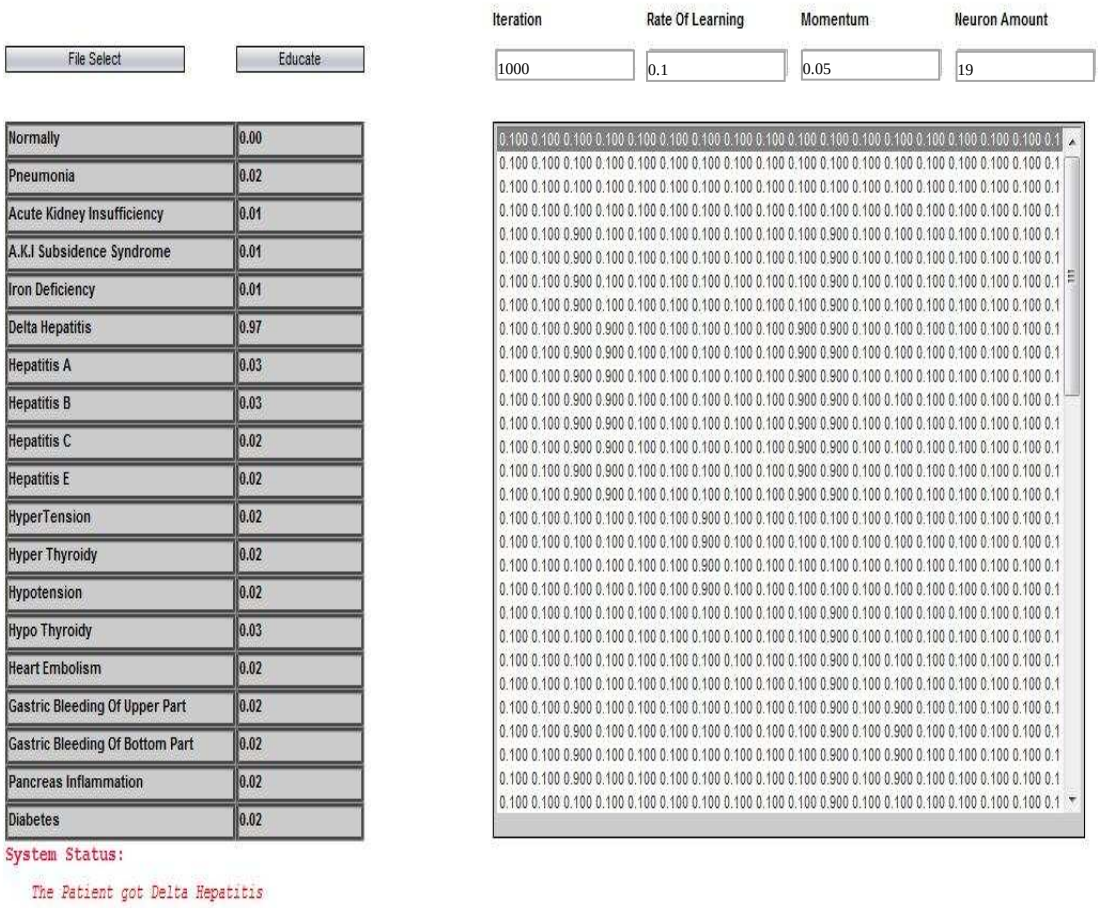


Şekil 4.3. Delphi ile yazılımış Teşhis programının arayüzü.

Kullanıcı listedeki hangi değeri test etmek isterse o satırın üstüne tıkladıktan sonra Sistem Durumu (System Status) satırında hangi hastalığın riskini taşıdığını görebilir.

Sisteme girilecek eğitim verisi, kullanıcı tarafından oluşturulmuş veriler değildir. Bu verilerin kaynağı hastane veritabanından gelmektedir. Sisteme yüklenen bu veriler test edilerek kullanıcıya bir sonuç raporu döndürülür. Daha önce bahsedildiği gibi normalizasyon sonucunda ortaya çıkan hangi değer 1.00 değerine en yakın ise hastada o hastalığın riski bulunmaktadır.

Aşağıda Şekil 4.4'de verilen örnekte görüleceği gibi, sistem durumu kısmında rapor sunulmuş ve elde edilen veriler ışığında sistemin aldığı karara göre kişinin Delta Hepatit olma riski bulunmuştur.



Şekil 4.4. Teşhis sonucu için bir örnek.

Uzman olmayan kullanıcı yani hasta için tasarlanan ekranda ise yine yukarıdaki işlemlerin aynısı işleyecek, fakat detaylar kullanıcıya yansıtılmayacaktır. Sadece kan tahlili veya idrar tahlilindeki anahtar değerlerin girişini yapması istenir. Örneğin, normal kullanıcı özel olarak Şeker Hastalığı teşhisi yapmak istediği zaman karşısına basit bir ekran çıkar. Bu ekranda sadece isim, yaş, şeker hastalığı olan akraba bilgisi, idrar oranı, plasma glukoz miktarı, kan basıncı, vücut ağırlığı, kolestrol ve boy gibi temel bilgilerin girilmesi istenecek ve sistem bunları teşhis YSA motoruna aktaracaktır. Daha sonra kullanıcıya şeker hastası olup olmadığı raporu sunulur. Bu işlemden elde edilen sonuçlar MySql veritabanında tutulan bir tabloda saklanarak ilerdeki sistem eğitmeleri için referans alınacaktır. Tüm bu teşhis yaklaşımları kesin sonuçlar olmamakla beraber testi yapan kişinin tekrar doktoruyla görüşmeye gittiği zaman ana bir fikir edinmesine katkı sağlayacaktır.

5. HASTALIK TEŞHİSİ İÇİN CEP BİLGİSAYARI ARAYÜZLERİ

Hastalık teşhisi için Microsoft .Net ortamında akıllı telefon ekranları tasarlanmış ve geliştirilmiştir. Bu uygulamanın çalışma prensibi önceki bölümde sunulan web arayüzünün çalışma prensibinden farklıdır.

Önce kullanıcı bu programı cep telefonuna kurar. Kurulacak işletim sistemi minimum Windows Mobile 5.0 olmalıdır. Günümüzde mobil cihazların teknik kapasitelerinin oldukça ileri bir noktada olduğu düşünüldüğünde, modern bir akıllı telefonun yapay sinir ağları algoritmalarını lokal olarak çalıştırabilme imkanı olduğu fikri ortaya çıkmaktadır.

Yapay sinir ağlarının yoğun hesaplamalı işlemleri gerçekleştirdiği, RAM ve işlemci gibi elemanları üst düzeyde kullanması gerektiği düşünüldüğünde, her problemin uygulanmasının mümkün olamayacağı ya da istenilenden çok daha uzun ve kabul edilemez bir sürede işlemi bitireceği ön görülmektedir. Bundan dolayı başlangıçta diabet (şeker hastalığı) ve kolestrol teşhisleri için programların geliştirilmesi düşünülmüştür. Bu mobil uygulamaların çalıştırıldığı ve performans testinin gerçekleştirildiği mobil cihaz olarak HTC Mini aygıtı kullanılmış ve donanımsal özellikleri Tablo 5.1'de verilmiştir. Tablo 5.2'de ise mobil uygulamalarla teşhis edilen Kolesterol ve Şeker Hastalığı hakkında kısa bilgi verilmiştir.

Tablo 5.1. HTC Mini aygıtının donanımsal özellikleri.

CPU	QUALCOMM(R) 7227, 600 MHz
RAM	384 MB
Flash Hafıza Boyutu	512 MB
Veri Yolu	32 Bit
Depolama Boyutu	223.93 MB
Ekran Çözünürlüğü	320 x 480
Renk Sayısı	65536
Bluetooth Versıyonu	2.1/2.1 + EDR

Tablo 5.2. Mobil uygulamayla teşhis edilen hastalıklar.

Teşhis Edilen Hastalık	**Açıklama**
Kolesterol (Cholesterol)	Çok fazla kolestrol, damarların çepherlerinde bir plak toplanmasına sebep olduğu zaman kanın kalbe ve diğer organlara akışını engellemektedir.
Şeker Hastalığı (Diabetes)	Pankreasın yeterli derecede insülin üretmemesiyle veya hücrelerin üretilen insüline tepki vermemesi sonucunda kandaki glikoz hücreler tarafından emilemez ve diabet rahatsızlığı meydana gelir.

Diyabet hastalığına teşhis yaklaşımı getirebilmek için öncelikle diyabet hastalığı ile ilgili bazı bilgilerin elde edilmesi gerekmektedir. Bu bilgiler, diyabet hastalığının belirtilerinin neler olduğu, diyabet hastalığının doktorlar tarafından nasıl teşhis edildiği, diabet hastalığının çeşitleri ve ne tür girişlerden test verileri olarak yararlanılması gerektiği ile ilgili bilgilerdir.

5.1. Diyabet Hastalığının Teşhisi İçin Mobil Program Arayüzleri

Diyabet hastalığının teşhisi için geliştirilen yazılım sunulmadan önce bu bölümde hastalık hakkında bilgi verilecektir. Daha sonra hastalıkta kullanılan teşhis veri girişlerinin programa nasıl entegre edileceği açıklanmaktadır. Veriler yapay sinir ağı ile eğitildikten ve test edildikten sonra elde edilen çıkış verisi aracılığıyla hasta kişiye sonucun nasıl sunulacağı yine burada anlatılmaktadır.

5.1.1. Diyabet Hastalığının Genel Tanımı ve Teşhisi

Diabet hastalığı 3 bölüm olarak ele alınmalıdır [23]. Ön-Diabet (Pre-Diabetes), Tip 1 ve Tip 2.

- Ön-Diabet (Pre-Diabetes)

Tip 2 diyabet gelişmeden önce "ön-diyabet" evresi göz önüne alınmaktadır. Ön diyabette kandaki glikoz miktarı yüksektir, fakat diyabete karar verilecek kadar yüksek değildir. Amerika Birleşik Devletlerinde 79 Milyon insan diyabet hastasıdır. Yakın zamanda yapılan araştırmalar, ön-diyabet evresi süresince vücudun, özelikle kalbin ve dolaşım sisteminin uzun süreli bir zarar görebileceğini göstermektedir [23].

- Bir Kişinin Diabet Olduğuna Nasıl Karar Verilir ?

Doktorların kullandığı iki tip test vardır. FPGT (Fasting Plasma Glucose Test) yani aç karnına yapılan plazma glikoz testi ve OGTT (Oral Glucose Tolerance Test) Oral Glikoz Tolerans Testidir.

FPG testinde, normal durumda aç karnına alınan kan örneğindeki glikoz seviyesi 100 mg/dl'nin altındadır. Ön diyabet evresindeki bir kişide bu seviye 100 mg/dl ile 125 mg/dl' dir. Eğer kandaki glikoz seviyesi 126 mg/dl veya daha da yukarı çıkarsa kişide diyabet olduğuna karar verilmektedir.

OGT testinde ise, kişinin kan tahlili aç karnına ve glikoz bakımından zengin bir içeceğin içildiği andan 2 saat sonra yapılmaktadır. Normal durumda içeceği içtikten 2 saat sonraki glikoz oranı 140 mg/dl' dir. Ön diabette 2 saatlik kan glikoz oranı 140'tan 199

mg/dl’ ye kadar değişim göstermektedir. Eğer iki saatlik kan şekeri 200 mg/dl ya da daha üst seviyelere çıkarsa kişinin diyabet hastası olduğu söylenebilir.

Diyabet hastalığı için risk grubunda olan ve diyabet testi yaptırması gereken kişiler, obezite problemi yaşayan, aşırı kilolu ve 45 yaşın üstünde olan kişilerdir. Yine bu grubun dışında kalan bireylerinde doktor danışmanlığında test yaptırması gerekebilir. 45 yaşından küçük fakat aşırı kilo problemi olan kişilerde eğer ön-diyabet veya diyabet belirtileri varsa doktor test önerebilir. Bu belirtiler; yüksek kan basıncı, düşük kolesterol ve yüksek trigliserid seviyesi, diyabet hastalığı bulunan bir akraba, 4 kilo 77 gr’dan daha yüksek bir ağırlıkta bebek doğurma veya diyabet hastalığının bol görüldüğü bir etnik gruba ait olma gibi belirtilerdir [24].

- Diabet Hastalığından Korunmanın Yolları

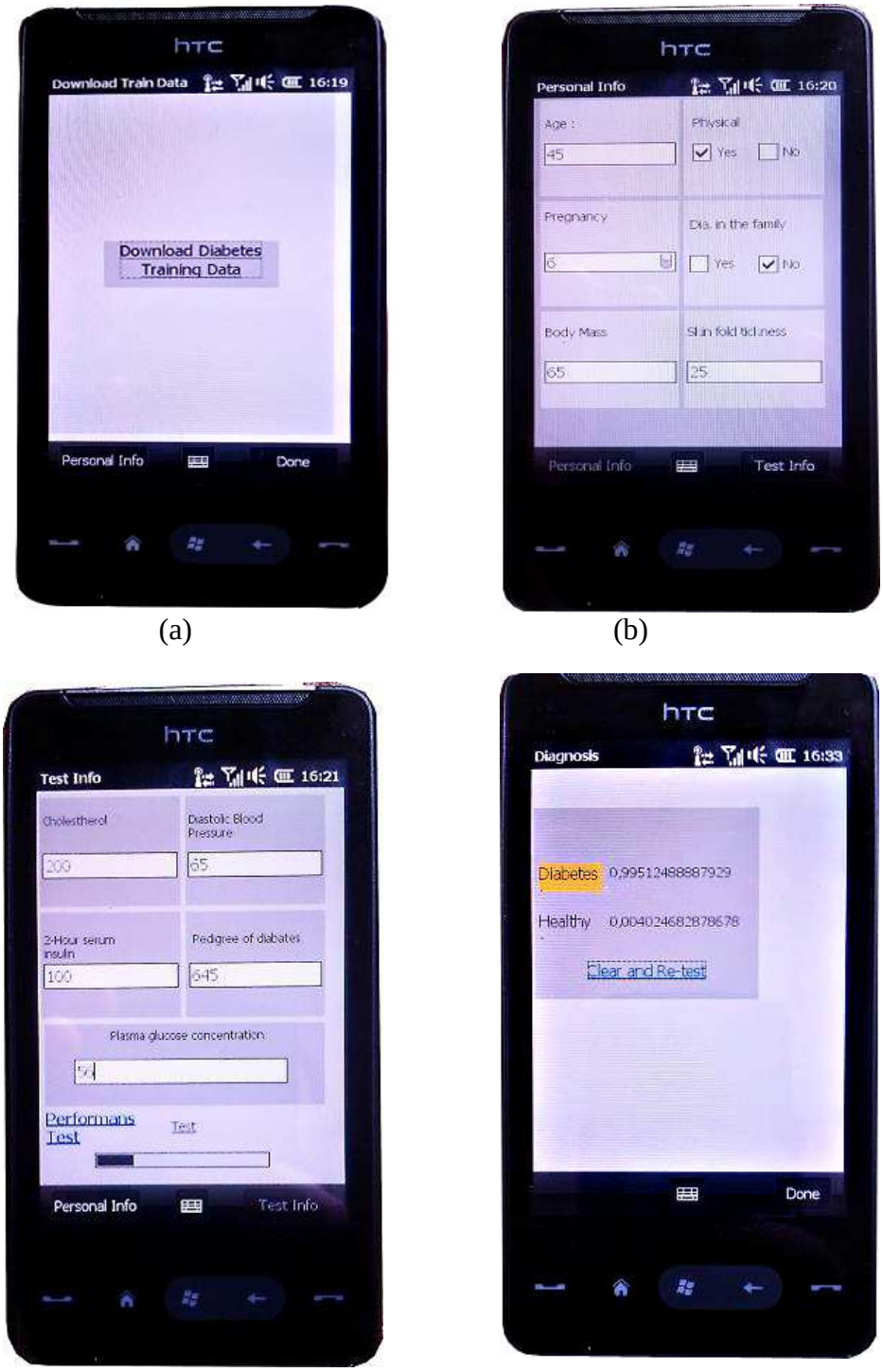
Bu hastalıktan korunmayla ilgili yapılan çalışmalar, insanların Tip 2 diyabetten sağlıklı beslenerek, fiziksel olarak aktif bir yaşam sürerek ve kilolarına dikkat ederek korunabileceklerini göstermektedir [24].

5.1.2. Diyabet Teşhisi için Geliştirilen Mobil Program Arayüzleri

Diyabet teşhisi için geliştirilen mobil program arayüzleri Şekil 5.1’de görülmektedir. Bu arayüzler arkasında çalışan kodlarda, teşhis işleminin yapay sinir ağlarına uyarlanması yapılarak bu arayüzlerin kullanıcılara test sonuçlarını kolayca anlaşılabilir şekilde sunması hedeflenmiştir. Aşağıda Şekil 5.1’de verilen ekranlar açıklanacaktır.

Eğitim Verisinin Sunucudan İndirilmesi: Yapay sinir ağının eğitilmesi için bir eğitim veri seti gereklidir. Bu eğitim veri seti herhangi bir hastanenin veritabanında bulunan kayıtlardan diyabet hastalığı ile ilgili verilerin toplanmasıyla oluşturulmaktadır. Bir grup hastanın diyabet test sonuçlarından elde edilen sayısal değerler kullanılarak bu eğitim dosyası meydana getirilmiştir. Bu eğitim dosyasının oluşturulduğu hastalardan bir kısmı sağlıklı bir kısmı ise hastadır.

Eğitim dosyası belirli bir formata göre düzenlenmiş bir metin (text) dosyasıdır. Eğitim veri setinin tutulduğu dosya, sunucu olarak çalışan bir bilgisayarda saklanmaktadır. Bu bilgisayara ağ üstünde sabit bir IP vererek, telefonların ağ üstünden bu bilgisayara ulaşabilmelerinde sorun yaşamamaları sağlanmıştır. Telefonlar ile sunucu görevi gören kişisel bilgisayar arasındaki iletişim soket programlama tekniği kullanılarak gerçekleştirilmiştir. Sunucu bilgisayarda bir TCP dinleyicisi sürekli bağlantı isteklerini dinleyerek sisteme ulaşan bağlantı isteklerine cevap vermektedir.



(a) (b)

(c) (d)

Şekil 5.1. Diabet hastalığı teşhisi için geliştirilen program arayüzleri: (a) Eğitim verilerinin sunucudan indirilmesi, (b) Kişisel bilgilerin girişi (c) Test bilgisi girişi (d) Test sonucunun görüntülenmesi.

IP numarası “192.168.1.7” olan sunucu bilgisayarın ağ üzerinde kullandığı port numarası olan 1453 aracılığıyla telefonlar sisteme bağlanmaktadır. İstemci yani telefon sisteme bağlandığı anda sunucu, eğitim giriş dosyası ve eğitim çıkış dosyasını istemciye gönderir. İstemci eğitim dosyalarının indirme ve kullanıma hazır duruma getirme işlemini tamamlamış olur ve program veri girişine hazır duruma gelir.

Kişisel Bilgilerin Girişi: Diyabet hastalığının teşhisinde, test aşamasında alınan yaş, hamilelik sayısı, vücut kütle oranı, fiziksel faaliyet durumu, diyabet hastalığına sahip bir akrabanın olup olmama durumu, deri kıvrımının kalınlığı gibi verilerin yapay sinir ağının eğitilmesi için uygun bir şekilde girilmesi gerçekleştirilmektedir. Buradan elde edilen verilerden sonra test değerlerinin de yapay sinir ağlarına girilmesi gerekmektedir. Bunun için de Test ekranına geçilir. Test ekranına geçmek için alt menu’deki Test info tuşuna basılır.

Test Bilgisi Girişi: Bu ekranda ise kolesterol, iki saatlik insulin serum değeri, kan basıncı ve diyabet pedigrisi gibi veriler girilmektedir. Kişisel bilgi ve Test bilgisi ekranından gelen veriler yapay sinir ağı modelinin giriş katmanına aktarılan verilerdir. Bu ekran aynı zamanda test işleminin gerçekleşeceği ve en önemli kısımdır.

Programda gerçekleşen olayları detaylı bir şekilde açıklamak gerekirse, önce sistemde tasarlanan yapay sinir ağı modelini anlatmak gerekmektedir. Bu çalışmada 3 katmandan oluşan, bir çok katmanlı perseptron ileri beslemeli yapay sinir ağı modeli kullanılmıştır. Bu yapay sinir ağı modeli Geri Yayınım algoritmasını kullanarak eğitilmektedir. Buradaki YSA modelinde, bir giriş, bir gizli ve bir çıkış katmanı kullanılmıştır. Daha önceki ekranların anlatımında olduğu gibi YSA modelinin girişleri Tablo 5.3’de verilmiştir.

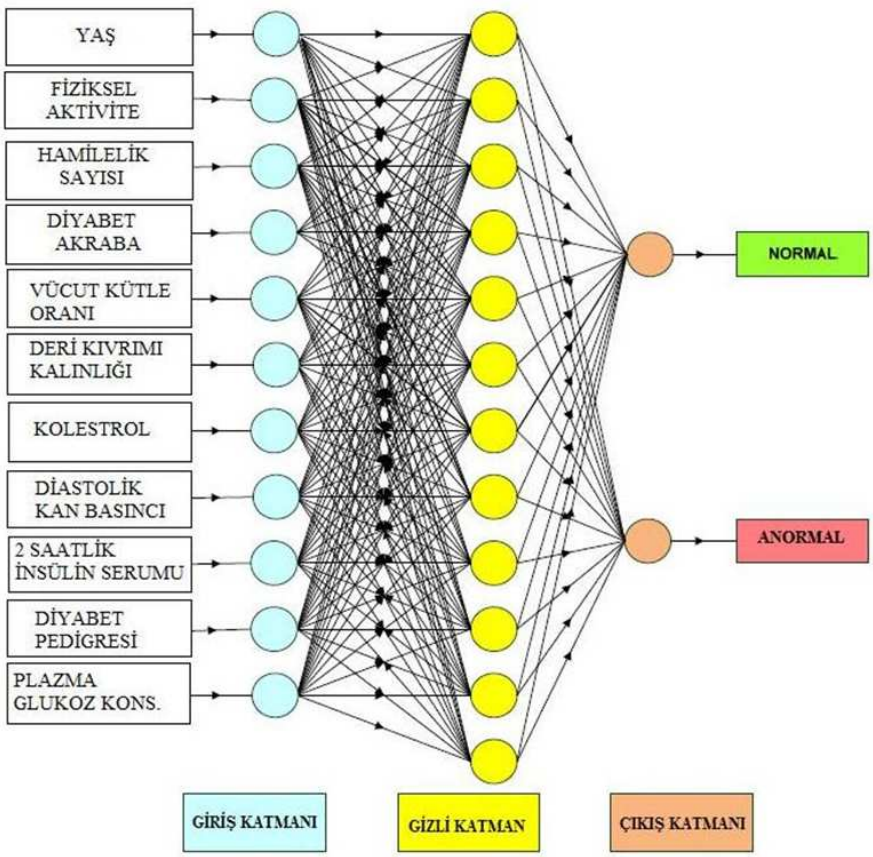
Diyabet hastalığının teşhisinde iki çıktı vardır. Bunlar “Hasta” ve “Sağlıklı Olma” durumlarıdır. Eğitim evresinde sistem 5000 iterasyonla, 0.95 değerinde bir öğrenme oranıyla ve 0.05’lik bir momentumla eğitilmektedir. Diyabet için kullanılan YSA modeli Şekil 5.2’de verilmiştir.

YSA modelindeki değerlerin 0 ile 1 arasında girilmesi gerekmektedir. Bundan dolayı ekrandan girilen veriler ortak bir sayı ile çarpılıp 0 ile 1 arasında değerlere çekilerek normalize edilir. Bu katsayı 1/1000’dir. Bu işlem programda aşağıda gibi yapılır.

```
cholesterol = double.Parse(m_textBox_Cholesterol.Text)/1000;
```

Tablo 5.3. Diyabet hastalığının teşhisi için tasarlanan yapay sinir ağında kullanılan veriler.

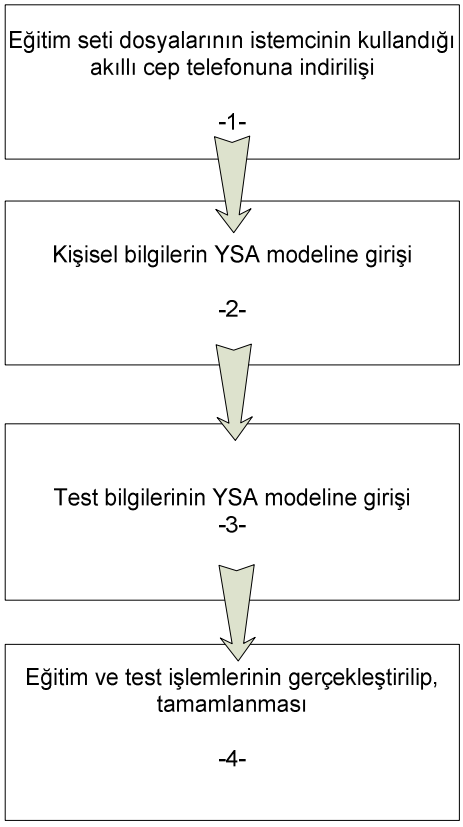
Veri	**Birim**
Yaş	Sene
Fiziksel Aktivite (Hayır/Evet)	(0/1)
Hamilelik Sayısı	Rakam
Ailede diyabet olan kişi var mı?	(0/1)
Vücut kütle oranı	Ağırlık (kg) / Uzunluk (m)
Deri kıvrımı kalınlığı	Mm
Kolesterol	Mg/dl
Dialostik kan basıncı	MmHg
2 Saatlik insülin serumu	mU / ml
Diyabet Pedigresi	Rakam
Plazma glukoz konsantrasyonu	mg / dl



Şekil 5.2. Diyabet hastalığı için kullanılan yapay sinir ağı modeli.

Programda tanımlanmış olan kolesterol değişkeninin değeri, arayüzdeki kolestrolü temsil eden *textbox* aracılığıyla atanmıştır. Bu değişkenin değeri tayin edildikten sonra normalizasyon işlemi 1/1000 ile çarparak gerçekleştirilir.

Test Sonucunun Görüntülenmesi: Test tuşuna basıldığı anda sistem önce eğitilmekte, daha sonra eğitilmiş veri test fonksiyonuna aktarılarak test işlemi gerçekleştirilmektedir. Test evresi bittikten sonra kullanıcıya bir rapor sunulmaktadır. Hasta ya da sağlıklı olması hakkındaki bilgilendirme yeni bir ekranda gösterilmektedir. Buradaki test sonucunda ortaya çıkan iki değer vardır. Daha önce bahsedildiği gibi diyabet hastalığının teşhisinde iki çıktı vardır. Yani normal ve anormal. Bu değerlerin her ikisi de 1 ve 0 arasında değerlerdir. Bu çıktılardan hangisi 1 değerine daha yakın ise o geçerli çıktı değeri olarak kabul edilecek ve kişinin hasta ya da sağlıklı olacağına karar verilecektir. Eğer sonuç normal ise sağlıklı, anormal ise hasta anlamına gelmektedir. Diyabet teşhisinde gerçekleşen adımların şematik gösterimi Şekil 5.3'de verilmiştir.



Şekil 5.3. Diabet teşhisinde gerçekleşen adımlar.

Test sonucunun görüntülenmesi ile ilgili olan ekranda yukarıda bahsi geçen adımlar başarıyla gerçekleştirildikten sonra kullanıcıya kolaylıkla anlayabileceği bir durum görüntüleme arayüzü sunulmaktadır. Sağlıklı ya da Hasta olma durumlarında ilgili sonuç kısmı sarı renk alarak kolaylıkla farkedilebilir. Bu ekranda ileri seviyede bir işlem yapılmamaktadır. Sadece önceki işlemlerden elde edilen sonuçları kullanıcının basit bir şekilde anlayabilmesi için gerekli çevirme işlemleri yapılmaktadır. Clear and Re-Test tuşu, yeni bir test yapılmak istenildiğinde sistemin yeniden kullanılabilir hale getirilmesi için düşünülmüştür.

5.2. Kolesterol Hastalığı Teşhisi İçin Mobil Program Arayüzleri

Bu bölümde geliştirilen kolesterol hastalığı teşhis programı hakkında bilgi sunulmadan önce kolesterol ile ilgili kısa bilgi sunulmaktadır. Bu hastalığın teşhisinde gereken giriş verilerinin yine diyabet hastalığının teşhisinde olduğu gibi yapay sinir ağlarının eğitiminde nasıl kullanıldığı anlatılmaktadır. Bu eğitimden sonra hasta kişiye bilgilendirmenin nasıl yapılacağı da bu bölümde anlatılmıştır.

5.2.1. Kolesterol Hastalığı ve Teşhisi Hakkında Genel Bilgiler

Kolesterol hayvansal dokularda bulunan yağlı bir maddedir ve insan vücudu için önemli bir bileşendir. Karaciğerde üretilir ve vücuda kan akışıyla taşınır. Çok fazla kolesterol kan damarlarının çeperlerinin etrafında bir plak toplanmasına sebep olursa, problemler meydana gelmeye başlar. Bu problemler kanın kalbe ve diğer organlara akışını engellemektedir. Kolesterol en çok ette, tavukta, midyede ve süt ürünlerinde bulunmaktadır. Kolesterolün hem iyi hem de kötü yanları vardır. Yiyeceklerden yağın sindirilmesini sağlar, hormonların üretimine katkı verir, hücre duvarlarını oluşturur ve sağlıklı bir vücut için gereken fonksiyonlara katılır. Kolesterol rahatsızlıklarından bahsedilirken genellikle yüksek kolesterol söz konusu olur [24].

Bu bilgiyi daha da açmak gerekirse kolesterolle ilgili 4 önemli bileşenden bahsetmek gerekir:

- LDL (Low Density Lipoprotein): Düşük yoğunluktaki Lipoprotein. Kötü kolestrol olarak da bilinir.
- HDL (High Density Lipoprotein): Yüksek yoğunluktaki lipoprotein. İyi kolestrol olarak da bilinir.
- Trigliserid : Kalp hastalıkları riskini arttıran bir kan lipididir.

- Toplam Kolestrol.

Kolesterolün damarları tıkaması sonucunda kalbe kan gidişinin engellenmesi, kalp krizine yol açabilir, kalbe ciddi zarar verebilir ve hatta ölüme sebebiyet verebilir. Bazı bilim adamları kolesterolün beyindeki sinirler için gerekli olan maddelerin taşınmasına engel olduğu için Alzheimer hastalığına sebep olduğunu düşünmektedir [24].

- Kolestrol Riski:

Kolesterolün risk grubunda olan kişilerde ailelerinde de kolesterol problemi yaşayan kişilerin olup olmadığı dikkate alınmaktadır. Bunların yanında obezite, alkolizm ve düzenli egzersiz eksikliği yine kolesterol testlerinde ilgi alanına giren konulardır. Yüksek kötü kolesterolün sebeplerinden birisi de beslenme esnasında alınan aşırı yağlı yiyecekler veya şekerli yiyeceklerdir. Kolesterol doğal yollardan karaciğerde üretilmekte ve yüksek kolesterol içeren besinlerden kaçınan kişilerde bile aşırı miktarda üretilebilmektedir. Düşük HDL ve yüksek trigliserid seviyeleri yine risk faktörlerini oluşturmaktadır [24].

- Sebepleri ve Semptomları :

Yüksek kolesterol genellikle aile hekimleri veya genel sağlık çalışanları tarafından teşhis ve tedavi edilmektedir. Bazı durumlarda endokrinolojist veya kardiolojist tarafından da teşhis ve tedavi gerçekleştirilebilir. Lipid Panel adı verilen bir test ile Toplam kolestrol, LDL, HDL ve trigliserid seviyeleri ölçülmektedir. Eğer toplam kolesterol değeri yüksek veya düşük ise buradan elde edilecek sonuçlar sadece rehber olarak kullanılmaktadır. Birçok erişkin için normal kabul edilen değerler şöyle olacaktır;

- Birim mg/dl kanda,
- Toplam kolestrol 200’ den düşükse,
- LDL 130’dan düşükse,
- HDL 35’den daha yüksekse,
- Trigliserid 30-200 arasında ise,
- Kolestrolün HDL’ye olan oranı 4’e 1 ise.

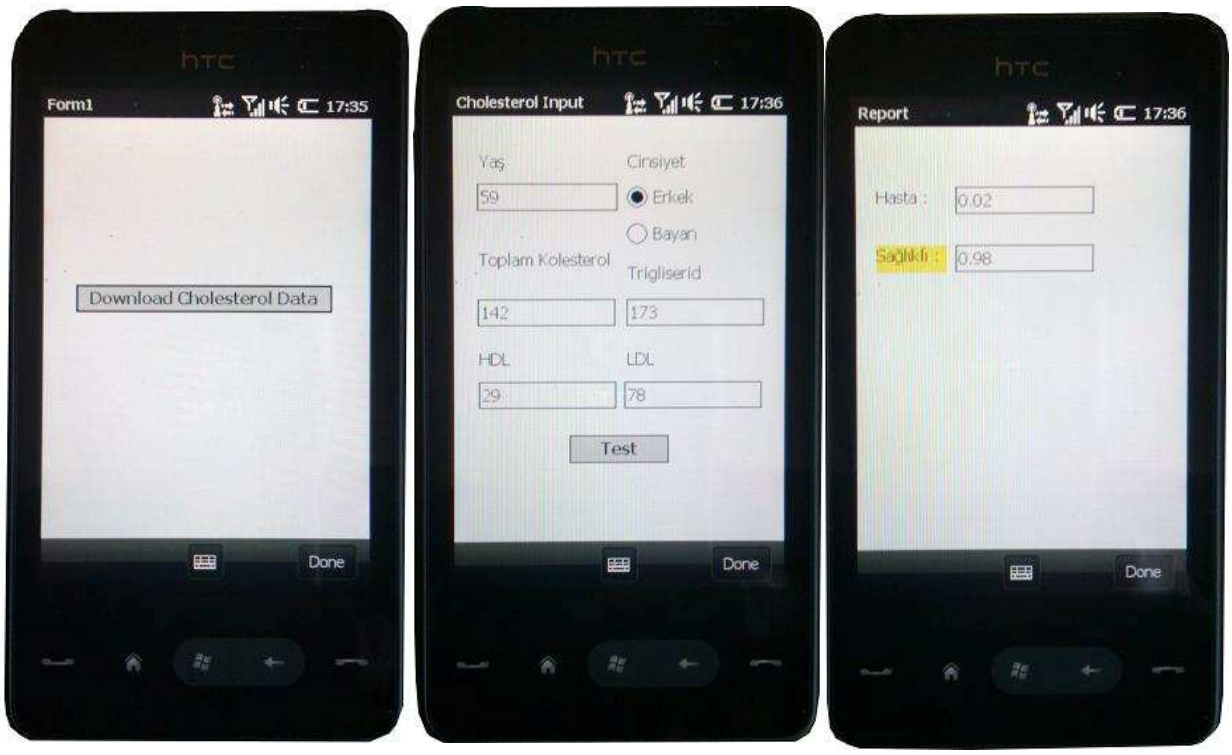
Tavsiye edilen kolesterol seviyeleri, hipertansiyona, ailede kalp rahatsızlığı geçmişine, diyabete, yaşa, alkolizme ve sigara içmeye bağlı olarak değişim gösterebilmektedir. Kolesterol tipleri ve ilgili değerler Tablo 5.4’de verilmiştir [24].

Tablo 5.4. Kolestrol çeşitleri.

Kolesterol tipleri	İstenilen Düzey	Sınır Düzeyi	İstenmeyen Düzey
Toplam Kolesterol	<200	200-240	>240
HDL Kolesterol	>45	35-45	<35
LDL Kolesterol	<130	130-160	>160
Toplam Kolesterolün HDL Kolesterole Oranı	<3	3-4	>4

5.2.2. Kolesterol Hastalığının Teşhisi İçin Geliştirilen Mobil Program Arayüzleri

Kolesterol hastalığının teşhisi için tasarlanan ve geliştirilen programın arayüzleri ve açıklamaları Şekil 5.4'de görülmektedir.



(a) (b) (c)

Şekil 5.4. Kolestrol hastalığı için geliştirilen programın arayüzleri: (a) Eğitim verilerinin indirilmesi, (b) Test için gereken verilerin girilmesi, yapay sinir ağının eğitilmesi ve test edilmesi (c) Test sonucunun raporlanarak kullanıcıya sunulması.

Diyabet hastalığında olduğu gibi kolestrol hastalığının teşhisine başlamak için de eğitim dosyalarının sunucudan indirilmesi gerekmektedir. Bu işlem yine sunucu ile bir soket bağlantısı kurarak gerçekleşmektedir. Eğitim dosyalarının sunucudan indirilmesinin sebebi diğer kullanıcıların girdikleri test verileriyle eğitim dosyasının sürekli güncellenebilmesidir. Eğitim dosyasındaki verilerin sayısı arttıkça yapay sinir ağlarının öğrenme yeteneği artacaktır ve bu kullanıcılara daha hassas ve doğru sonuçlar alma şansı doğuracaktır.

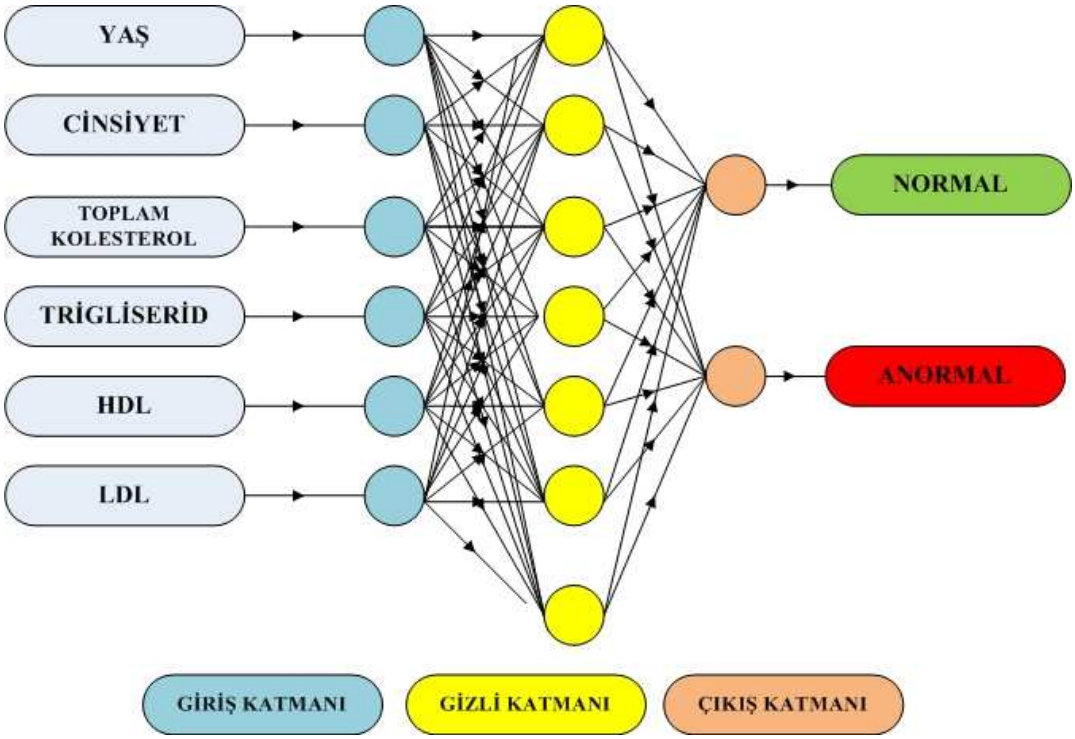
Kolesterol verilerinin girildiği ekranda, daha önce diyabet verilerinin girilmesinde olduğu gibi, burada da yine yapay sinir ağlarının test işleminde ihtiyaç duyduğu veriler kullanıcılardan bu arayüz sayesinde alınmaktadır. Kullanıcı bu verileri sisteme girer, test düğmesine basar ve YSA modeli daha önceden sunucudan indirilen eğitim verileri ile eğitilir. Veriler girilirken daha önce bahsedilen normalizasyon işlemi burada da gerçekleşmektedir. Girilen değerler sayısal ifadelere çevrilir ve 1 ile 0 arasında değerlere dönüşür. Eğitim tamamlandıktan sonra kullanıcının bu ekranda girdiği verilere göre sistem test edilir ve program test çıktısını oluşturur. Oluşturulan bu test çıktısı son ekran olan rapor ekranına sonucun aktarılması için kullanılır. Tüm bu işlemler bittikten sonra otomatik olarak rapor penceresi açılacaktır. Rapor penceresi kullanıcının kolayca anlayabileceği durum raporundan meydana gelen bir arayüz olmalıdır ve bu tasarımla bu olayın gerçeklenmesi hedeflenmektedir.

Kolesterol teşhisi için de 3 katmandan oluşan, bir çok katmanlı perseptron ileri beslemeli yapay sinir ağı modeli kullanılmaktadır. Bu YSA modeli Geri Yayınım algoritmasını kullanarak eğitilmektedir. YSA modelinde, bir giriş, bir gizli ve bir çıkış katmanı kullanılmıştır. Yapay sinir ağının testi için kullanılan veriler Tablo 5.5'de verilmiştir.

Tablo 5.5. Kolestrol teşhisinde kullanılan YSA modelindeki veriler.

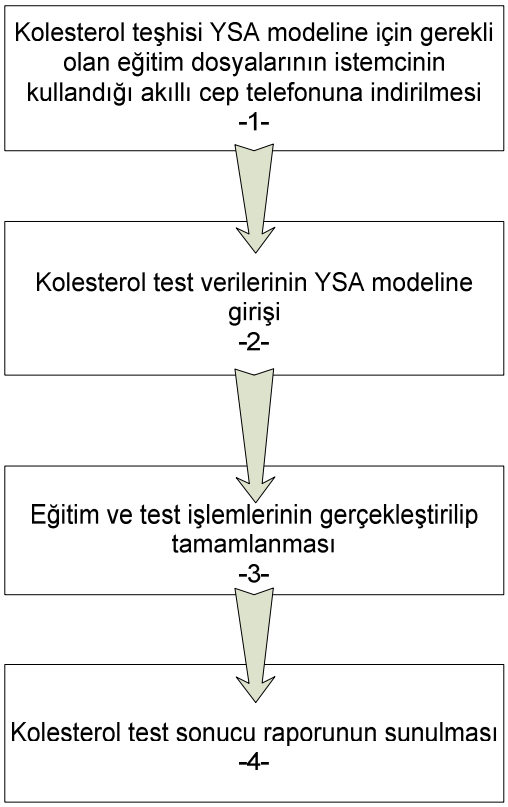
Veri	**Birim**
Yaş	Sene
Cinsiyet	Erkek/Bayan (0/1)
Toplam Kolestrol	Rakam
Trigliserid	Rakam
HDL	Rakam
LDL	Rakam

Kolesterol hastalığının teşhisinde de yine 2 çıktı vardır. Bunlar “Hasta” ve “Sağlıklı Olma” durumlarıdır. Eğitim evresinde sistem 5000 iterasyonla, 0.95 değerinde bir öğrenme oranıyla ve 0.05’lik bir momentumla eğitilmektedir. Kolesterol için kullanılan YSA modeli Şekil 5.5’de verilmiştir.



Şekil 5.5. Diyabet hastalığı için kullanılan yapay sinir ağı modeli.

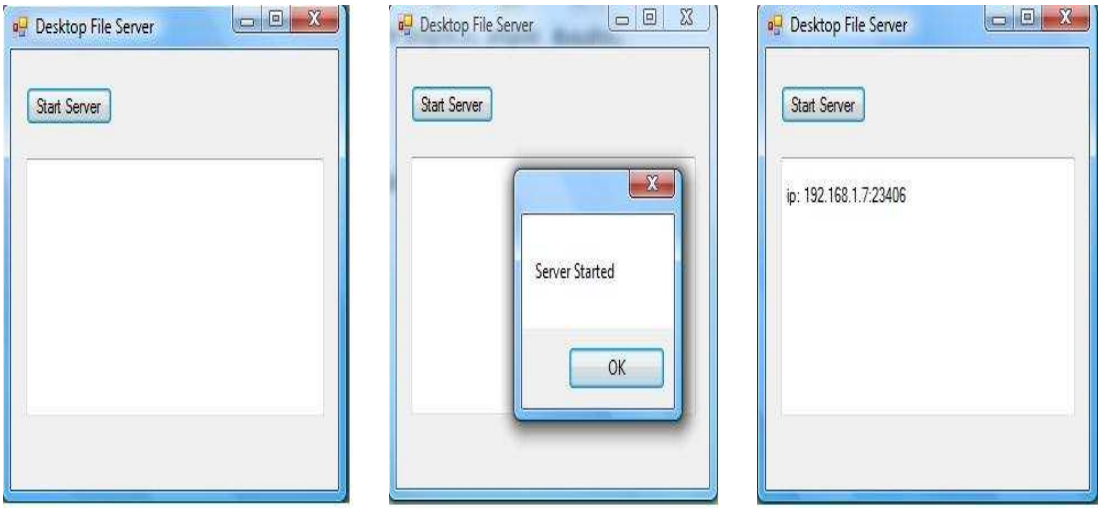
Bu YSA modelinde de yine diyabet hastalığında olduğu gibi, girişler, gizli katman ve çıkış katmanı modellenmiştir. Giriş katmanındaki giriş sayısı kolesterol hastalığının teşhisinde gösterilen parametrelerin sayısı olan 6’dır. Bu girişler sırasıyla, yaş, cinsiyet, toplam kolesterol, trigliserid, HDL ve LDL’dir. Bu girişler kullanılarak ağ eğitilmekte ve eğitim tamamlandıktan sonra istenilen çıkış değeri elde edilmektedir. Ağın sadece 2 çıktısı vardır. Bu çıktılar Normal ve Anormal olarak adlandırılmıştır. Normal sağlıklı kişiyi temsil etmekte ve Anormal ise hasta kişiyi temsil etmektedir. Sistemin çalışmasının şematik bir modeli Şekil 5.6’daki görülmektedir.



Şekil 5.6. Kolesterol testinde gerçekleşen adımlar.

5.3. Mobil Uygulamaların Eğitim Dosyalarına Erişimini Sağlayan Sunucu Programı

Bu tezde geliştirilen mobil uygulamaların, eğitim giriş ve eğitim çıkış adlı iki veri seti dosyasına ilgili yapay sinir ağını eğitmeleri için ihtiyaçları vardır. Mobil cihazların hafıza kapasitelerinin masaüstü bilgisayarlara göre daha sınırlı bir yapıda olduğu bilindiği için bu dosyaları sürekli olarak mobil cihaz içerisinde saklamak yerine teşhis işlemi süresince kullanabilmesi için bir sunucu bilgisayardan indirmesi düşünülmüştür. Ayrıca yine bu yöntem eğitim dosyalarının sık sık güncellenmesine mümkün kılmaktadır. Güncel eğitim dosyaları sistemi daha doğru teşhis yapabilme konusunda daha yetenekli hale getirecektir. Sistemin eğitim verileri sürekli olarak yenilenebilir. Bu işlem sadece sunucu programının içerisinde saklanan metin tabanlı eğitim dosyalarını değiştirerek gerçekleşebilir. Geliştirilen sunucu programının arayüzü Şekil 5.7'da görülmektedir.



(a) (b) (c)

Şekil 5.7. Mobil uygulamaların eğitim dosyalarına erişimini sağlayan sunucu programı arayüzleri.

Sunucu açıldığı anda kullanıcının karşısına çıkan arayüz (a) basit bir şekilde sadece başlat tuşuna basarak sunucuyu aktif ve ağı dinler duruma getirmektedir. Sunucunun sorunsuz çalıştığını bildiren arayüz (b) başlat tuşuna basıldıktan sonra sunucu aktif olurken düzgün bir şekilde başlayıp başlamadığını kullanıcıya bildiren ekrandır. Bağlantı durumunu gösteren arayüz (c) ise herhangi bir mobil cihazdan sunucuya bağlantı olduğu zaman bağlanan istemcinin IP numarasını ve bağlantının kurulduğu port numarasını bildirmektedir.

Sunucu programının çalışma prensibinin altında Soket Programlama tekniği yatmaktadır. Bu sunucu programında bir TCP dinleyicisi çalışma anı başladıktan itibaren ağı dinlemeye başlar. 1453 numaralı port'u açık tutarak buradan gelen istemci bağlantılarını kabul eder. Bu programda kullanılan soket tipi 'stream'dir. Dolayısıyla büyük dosyaların ve verilerin gönderilmesinde sorun teşkil etmez.

İstemci sunucuya bağlandığı anda, sunucu kendi klasöründe bulunan "Train_input.txt" ve "Train_output.txt" dosyalarını byte dizilerine dönüştürerek soket üzerinden istemciye gönderir. Ve aynı şekilde istemci üzerinde bulunan soket üzerinden da gelen bu geri dönüşü kabul eder ve ağ üzerinden aldığı bu byte dizilerini tekrar kendi üzerinde "Train_input.txt" ve "Train_output.txt" dosyalarına dönüştürür ve yapay sinir ağının eğitimi için hazır duruma getirir. Sunucu birden çok istemci bağlantısını kabul edebilecek yeteneğe sahiptir ve asenkron olarak çalışabilir.

6. PERFORMANS ANALİZLERİ

Bu tezde geçen sistem PC, web ve mobil ortam olmak üzere 3 farklı platform üzerine yayılmaktadır. Her ortamın kendine göre farklı ve özel yetenekleri, avantajları ve dezavantajları bulunmaktadır. Bu avantaj ve dezavantajlara örnek vermek gerekirse, genellikle modern bir kişisel bilgisayarın performansı herhangi bir mobil cihazdan daha yüksek olacaktır, fakat mobil cihazın sağlayabildiği portatifliği bir bilgisayar sağlayamayacaktır. Benzer bir şekilde, bir web arayüzü herhangi bir işletim sistemine bağımlı olmadan her ortamda çalışabilme şansına sahip olduğu için avantajlar sağlayacaktır, fakat web sitelerinin de bir bilgisayara yüklenen programların çalışması gibi işlem göremeyeceği alanlar vardır. Bir web sitesinin sadece browser içerisinde operasyonunun gerçekleştiği düşünülürse, bilgisayar üzerinde çalışan bir program kadar yetenekli olamayacağı ve uygulamayı geliştirirken buna göre yazılım teknikleri geliştirmek gerekeceği anlaşılmaktadır.

Tüm bu farklı ortamlardan alınacak verimliliği nicel olarak karşılaştırabilmek ve performans analizleri yapabilmek için en uygun teknik olarak çalışma sürelerini karşılaştırmak düşünülmüştür. Bunun için geliştirilen bir performans analiz kodu geliştirilen ana programların içine yerleştirilerek, buradan elde edilen sonuç kayıt dosyasından performans hakkında sonuçlar alınmıştır.

Bunun yanında web arayüzlerinin analizi için de Netbeans derleyicisine eklenen bir performans izleme aracı olan Netbeans Profiler kullanılmıştır. Ayrıca geliştirilen web uygulaması, alt yapı olarak Delphi programını kullandığı için yine bu web sitesinin toplam cevap verme süresi anahtar değer olarak alınmıştır. Temel çalışma prensibi daha önce sunulduğu gibi, burada tasarlanan web sitesi Delphi programına sadece bir arayüz sağlamaktadır. İç hastalıklarının yapay sinir ağları aracılığıyla teşhis edilmesi için gereken karmaşık işlemler ve kodlar web sitesinin içerisinde değil web sunucusunda bulunan Delphi uygulamasında çalışmaktadır.

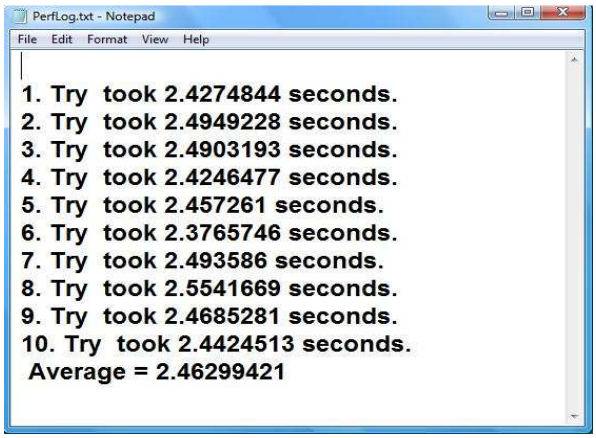
6.1. Performans Analizi İçin Geliştirilen Sınıf ve Kullanımı

Peformans analizi bu tezin ilgi alanına giren konulardan birisi olduğu için, tezde geliştirilen uygulamaların çalıştıkları tüm ortamlardaki performansları ölçülmüştür. Bunu gerçekleştirebilmek için programların çalışma zamanı performans kriterlerini ölçmeye yarayan bir kod bloğu geliştirilmiştir. C# dilinde performans ölçümlerine yardımcı olabilecek yeterlilikte sistem sınıfları mevcuttur. Bu sınıflar aracılığı Microsoft Windows

işletim sisteminin çekirdek yapısında bulunan API'lere ulaşabilerek detaylı bir zaman ölçümü algoritması geliştirmek mümkündür.

Öncelikle zamanı ölçmek için kullanılan sınıflardan birisi olan **`System.DiagnosisStopWatch`** sınıfından yararlanılmıştır. Bu sınıf zamanı milimetrik düzeyde ölçebilecek düzeyde kapasiteye sahiptir. Dolayısıyla çok derin ve detaylı bir şekilde harcanan zaman/performans ilişkisi üzerinde fikir vermekteki anahtar elemanlardan birisi olmuştur.

Yukarıda kısaca sunulan kavramların ışığı altında, sistemin farklı platformlarda çalışma zamanları ölçülerek programların performans analizi yapılmıştır. Zaman ölçümü metoduyla performans analizine gidilirken, öncelikle C# dilinde önceden tanımlı bir sınıf olan **`TimeSpan`** sınıfından oluşturulan **`TimeSpan`** nesnesi ile bir zaman sayacı çalışmaya başlatılıyor ve daha sonra performansının test edilmesi istenilen metotlar kullanıcı tarafından belirlenen sayıda çalıştırılıyor. Çalıştırma işlemi bittikten sonra zaman sayacı durduruluyor. Kaç saniye geçtiği bilgisi buradan alınarak bir rapor dosyası oluşturulmaktadır. Her çalıştırılma ile ilgili toplam zaman ve ortalama zaman bilgisi bu kayıt dosyasına işlenir ve kod bloğu çalışmayı sonlandırır. Örnek olarak mobil diyabet teşhis programının PC'de çalışırken harcadığı zamanları gösteren kayıt dosyası ve içerisindeki veriler ve bu değerlerin ortalama değeri Şekil 6.1'de görülmektedir. Bu şekilde mobil diyabet programında hastalığın teşhisini yapabilmek için anahtar metotlar olan Train ve Test metodlarını 10'ar kere çalıştırarak kaç saniye harcadıkları hesaplanır. Bu değerler görüntülendikten sonra tüm bu denemelerin ortalaması hesaplanarak yazılır.



Şekil 6.1. Performans raporu kayıt dosyası.

Diyabet teşhisi mobil uygulamasının bir masaüstü bilgisayarda ve bir mobil cihazda çalıştırıldıktan sonra elde edilen çalışma sürelerinin karşılaştırılması Tablo 6.1'de verilmiştir.

Tablo 6.1. Diyabet teşhisi programının iki farklı ortamda performans karşılaştırması.

Deneme	Akıllı Telefon	Bilgisayar
1	5.185195 sn	3.0689015 sn
2	5.242278 sn	2.7400436 sn
3	5.5351076 sn	3.12526665 sn
Ortalama	5.25952 sn	2.9872038 sn

Diabet Teşhisi uygulamasındaki yapay sinir ağlarının eğitimini ve test işlemlerini gerçekleştiren Train ve Test Thread'leri üçer kez üstüste çalıştırılarak ortalama bir değer alındığında telefonun performansının bilgisayara göre daha düşük olduğu Tablo 6.1'de görülmektedir. İlgili cihazların teknik özellikleri de göz önünde bulundurulduğunda ve aradaki zaman farkının çok yüksek boyutlarda olmaması nedeniyle bu uygulamanın mobil cihazlarda diyabet hastalığının teşhisi için kullanılabilmesi mümkün gözükmektedir.

6.2. Web uygulamasının performans analizleri

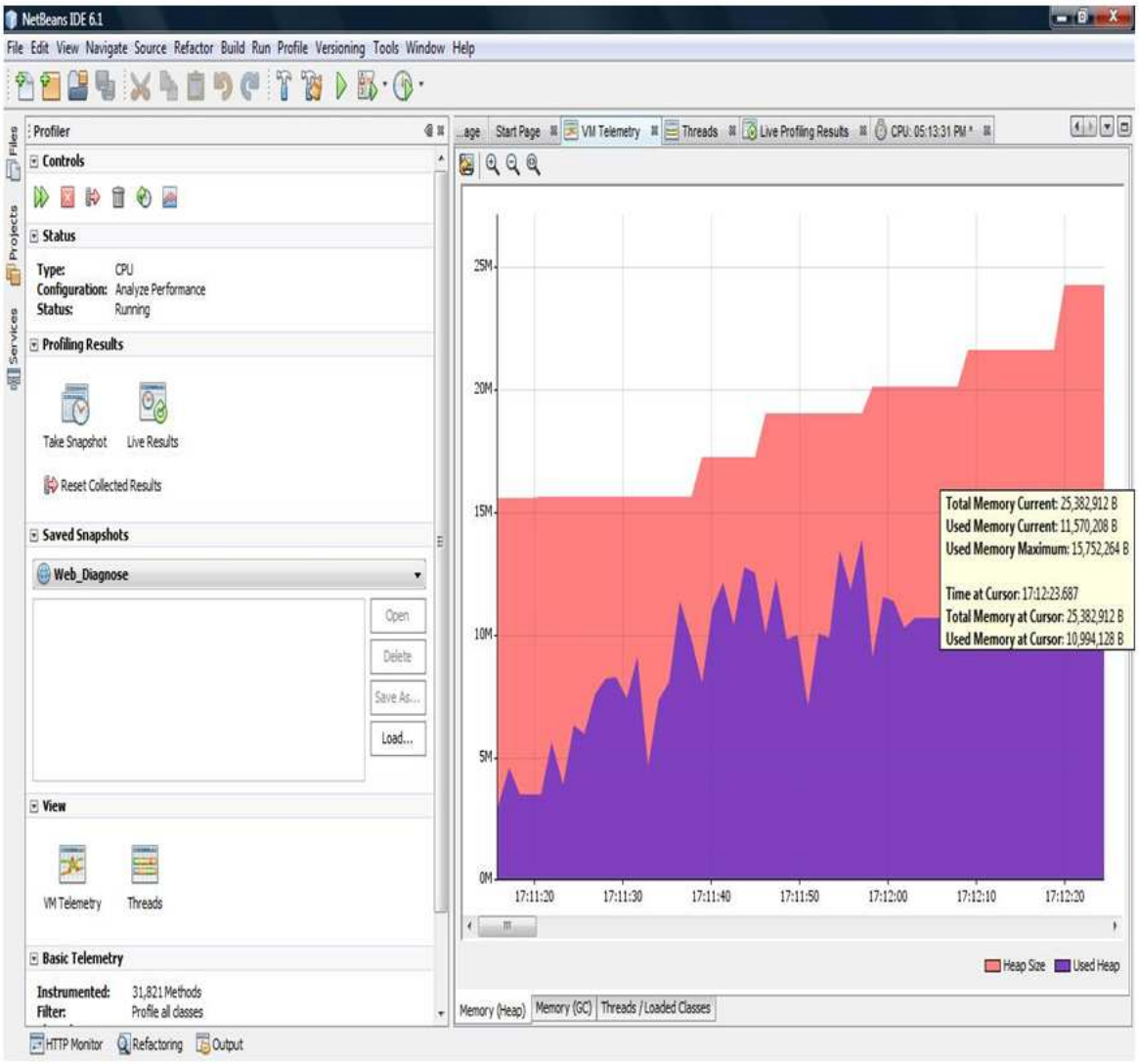
Web uygulamasının performans analizlerinde iki kategori vardır:

1. Web uygulamasının teknik ve yük performans analizi.

2. Tepki zamanlarının ölçülerek grafikle izah edilebilir hale getirilmesi.

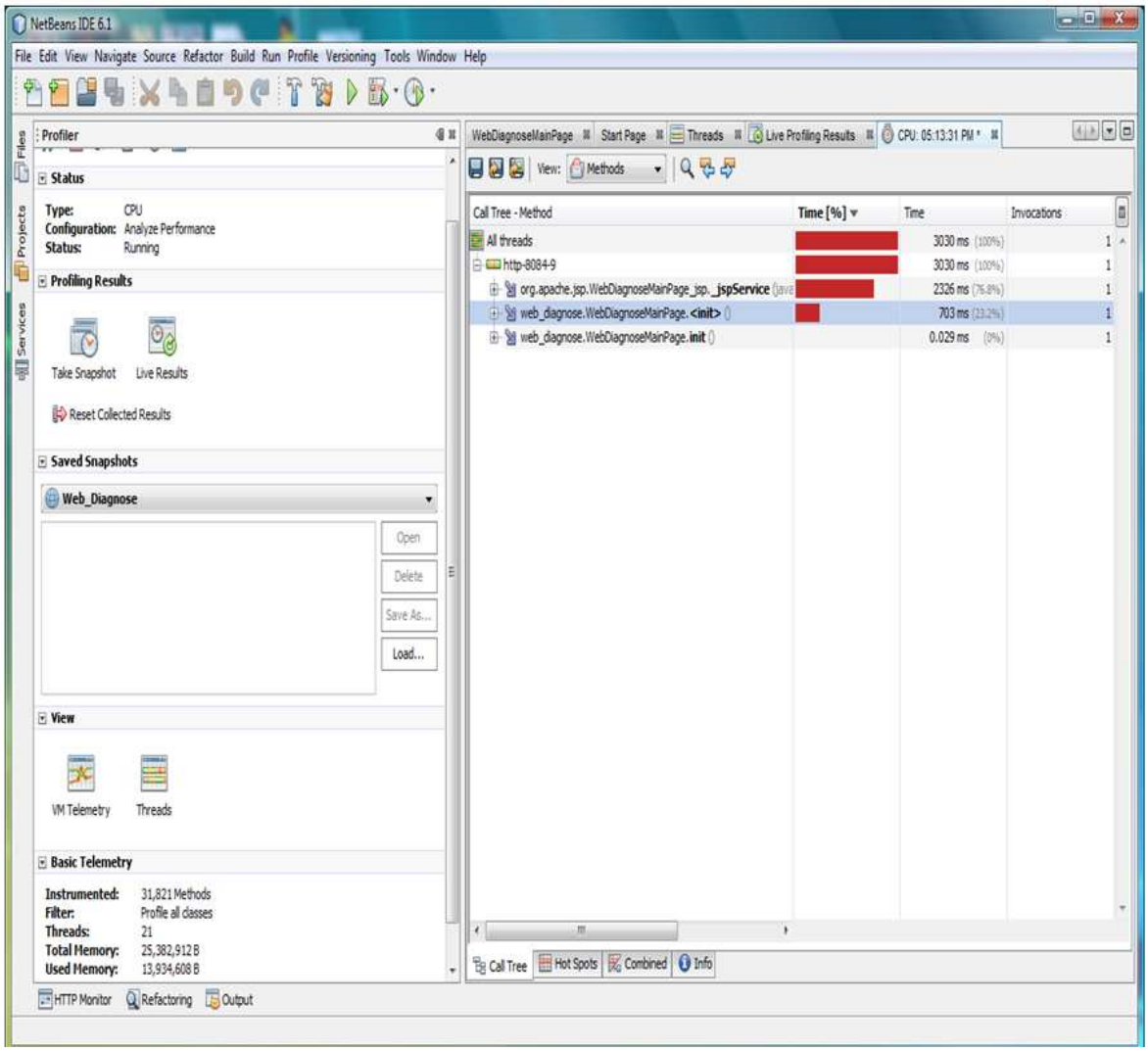
Web uygulamasının teknik ve yük performans analizleri yapılırken daha önce bahsedilen Netbeans plug-in'i olan Netbeans Profiler aracından yararlanılmıştır. Bu araç aynı şekilde yine grafikler ve gerçek zamanlı sonuç raporları verebilmektedir.

Java web uygulamalarındaki performans testleri web sayfası görüntülenirken kullanıcının bilgisayarındaki Java Sanal Makinesi için geçerlidir. Şekil 6.2'de görüldüğü gibi, Total Memory Current, Used Memory Current, Used Memory Maximum, Time At Cursor, Total Memory At Cursor, Used Memory At Cursor gibi parametreler mevcuttur. Bu parametreler sırasıyla, anlık toplam kullanılan hafıza, anlık kullanılan hafıza, maksimum kullanılan hafıza, Mouse imlecindeki toplam hafıza, Mouse imlecinin gösterdiği yerdeki toplam kullanılan hafızadır.



Şekil 6.2. Web arayüzünün hafızanın heap bölgesindeki durumu.

Şekil 6.3'te görülen değerler tüm thread'leri (işlem parçacıklarını) listelemektedir. Web uygulamasında çalışan elemanlar JSP sayfalarıdır ve bu JSP sayfalarının çalışabilmesi için JSP servisi gereklidir. Tüm JSP sayfaları çalışırken bu servis aktif durumda olmaktadır. Bundan dolayı grafikte JSP servislerinin kullanım zamanları gösterilmektedir. Bu grafik alınırken "WebDiagnoseMainPage" adı verilen uygulamanın açılış ve kapanış anındaki zamanları gösterilmekte ve bunlar Java sanal makinesinde çalışan tüm uygulamaların harcadığı zaman ile birbirine oranlanarak yüzdelik değerler halinde de gösterilmektedir. Aynı şekilde milisaniye birim cinsinden gösterilmektedir.



Şekil 6.3. CPU'daki zamana bağlı işlem durumunu gösteren grafik.

Bundan ayrı olarak web uygulamasının herhangi bir isteğe cevap vermesindeki toplam zaman hesaplanmalıdır. Bir web projesinin performans testinde birçok faktör incelenmelidir. İnternet hızına göre değişim gösterilebilir veya kullanılan web tarayıcı yine hız ve performans kriterlerini etkileyebilir. Buradaki teşhis uygulaması sunucu tarafında çalıştığı için, sunucu tarafının gösterdiği performans sonuçları asıl olarak bakılması gereken çalışma alanıdır. İç hastalıkların teşhisinde yararlanılan Delphi uygulaması sunucu tarafında çalışmaktadır ve öncelikle bu uygulamanın tepki zamanı ve web sunucusunun tepki zamanları toplanarak toplam geri dönüş süresi hesaplanmaktadır. Sunucu tarafındaki performansın ölçümünde web tarayıcısının sayfayı yüklerken harcadığı zaman da aynı şekilde dikkate alınır.

Başlangıç olarak web sunucusunda çalışan Delphi uygulamasının tepki süresi 5 ayrı deneme için ölçülmüş ve ortalama değeri hesaplanmıştır. Sistemin eğitilmesi ile ilgili olan performans/zaman ölçümleri bir grafiğe dökülmüştür. Sistemin eğitimi, 1000 Iterasyon, 0.1 Öğrenme Oranı, 0.05 Momentum, ve 19 Nöron sayısı ile gerçekleştirilmiş ve elde edilen zaman/performans ilişkisi Tablo 6.2'deki grafik aracılığıyla gösterilmektedir.

Tablo 6.2. Web uygulamasının performans verileri.

Eğitim	**Zaman**
1	16.3 sn
2	17.8 sn
3	17 sn
4	17.4 sn
5	16.9 sn
6	16.9 sn
7	17.5 sn
8	17.1 sn
9	16.8 sn
10	16.7 sn
Ortalama	17.04 sn

Tablo 6.2'deki verilere göre sistem ilk çalıştırmada eğitim işlemini toplam 16.3 saniyede tamamlarken, ikinci denemede 17.8 saniyede, üçüncü denemede 17.8 saniyede ve bu şekilde devam edip son onuncu denemede ise 16.7 saniyede tamamlamıştır. Zaman dilimleri arasında ani değişimlerinin olması, ağ trafiğinde meydana gelen değişimlerden veya istemci makinesinin ağa olan erişiminde meydana gelen problemlerden kaynaklanabilir.

Tüm bu denemeler sonucunda elde edilen ortalama değer 17.04 saniyedir. Yani ağın eğitilerek istemciye test imkanı sunması 17.04 saniye içerisinde gerçekleşmektedir. Sistemin çalıştırıldığı web tarayıcı olarak Google Chrome kullanılmış ve 1 Mbit hızında bir Internet bağlantısı kullanılmıştır.

7. SONUÇLAR

Bu tezin başlangıcında belirlenen 3 ana hedef konusunda çalışılmış ve birçok sonuca ulaşılmıştır. İlk hedef olarak belirlenen Kurumsal Uygulama yapıları ve arayüzleri tüm yönleriyle incelenmiş, benzer projeler göz önünde bulundurularak ikinci hedef olan ve yine bu tezde bahsi geçen Hastaneler için hastalık teşhis imkanı sağlayan prototip uygulamalar geliştirilmiştir. Projenin sonucunda ortaya çıkan bu prototip uygulamalar çalıştırılarak test edilmiş ve doğru bir şekilde birçok iç hastalığının teşhisini sunmaya imkan vermiştir.

Yazılımlar geliştirildikten ve test edildikten sonraki aşamada ise, üçüncü ana hedef olarak geliştirilen programların performans analizleri yapılmıştır. Teşhis programlarının mobil sürümlerinin ve PC ortamındaki çalışma performansları test edilmiştir. Uygulamaların mobil cihaz ve kişisel bilgisayarlarda çalışırken harcadıkları süreler karşılaştırılarak yapay sinir ağlarının mobil cihazlarda çalıştırılıp çalıştırılamayacağı hakkında fikirler elde edilmiştir.

Bu çalışmada geliştirilen programlar, herhangi bir hastanede tedavi gören ve çeşitli hastalıklarla ilgili araştırmalara tabi tutulan hastaların, evlerinden veya hastane dışından sisteme erişerek kendi kendilerine kolayca teşhis yapabilme imkanı sağlamaktadır. Ayrıca yine bu çalışma sonucunda, benzer programların kullanılmaları durumunda hastanelerin ara eleman yükünde de azalma gözleneceği sonucuna varılmıştır. Çünkü basit bir teşhis işlemi için personel çalıştırmaya gerek kalmayacaktır. Dolayısıyla insan gücünden ve zamandan da kazanç sağlanacaktır.

Ayrıca yine tüm teşhis sonuçlarının belirli bir veri havuzunda depolanmasıyla her yeni teşhis isteğinde sistem daha da fazla eğitilmiş olacağı için çok daha keskin ve net teşhis sonuçlarına ulaşılabilecektir.

8. KAYNAKLAR

1. A. Ersan, B. Karlık, "Real Time Computer Aided Diagnosis of Internal Illness", *UkrObraz'2006*, Kiev, Ukraine, 28-30 August, 2006.
2. O. Karan, C. Bayraktar, H. Gümüşkaya, B. Karlık, "Diagnosing Illnesses using Neural Networks and Pervasive Healthcare Computing", *1st International Symposium on Computing in Science and Engineering (ISCSE)*, Kuşadası, Aydın, June, 3-5, 2010.
3. M. M. Nelson, W. T. Illingworth, *Practical Guide to Neural Nets*, Addison Wesley, 1994.
4. U. Varshney, "Pervasive Healthcare and Wireless Health Monitoring", *Mobile Networks and Applications*, Vol. 12, Issue 2-3, 2007.
5. V. Tresp, T. Briegel, J. Moody, "Neural Network Models for the Blood Glucose Metabolism of a Diabetic", *IEEE Transactions on Neural Networks*, pp. 1204–1213, 1999.
6. M. Pradhan, R.K. Sahu, "Predict the Onset of Diabates Disease Using Artificial Neural Network (ANN)" *International Journal of Computer Science & Emerging Technologies*, Vol. 2, Issue 2, pp. 303-311, 2011.
7. P. Venkatesan, M. L. Suresh, "Breast Cancer Survival Prediction Using Artificial Neural Network, *International Journal of Computer Science and Network Security*, Vol. 9, No. 5, pp. 169-174, 2009.
8. R. Johnson, "J2EE Development Frameworks", *IEEE Computer*, Vol 38, Issue 1, pp. 107-110, 2005.
9. O. Hamed, N. Kafri, "Prediction Prediction of Web Based Application Architectures Case Study: .Net vs Java EE", *International Journal of Web Applications*, Vol. 1, Issue 3, pp. 146-156, 2010.
10. Java EE Tutorial: http://download.oracle.com/javaee/6/tutorial/doc/
11. Java Oracle Blog: http://www.javaoracleblog.com/java/Technology_Standards_in_Telecom_Domain.jsf
12. Microsoft: http://msdn.microsoft.com
13. Delphi about resmi web sitesi: http://delphi.about.com/
14. Netbeans web sitesi: http://netbeans.org/
15. C# online: http://en.csharp-online.net
16. Open source testing: http://www.opensourcetesting.org/
17. Java Performance Tuning: http://www.javaperformancetuning.com/
18. C# Code Speed Test Utility: http://www.vcskicks.com/code_speed_test.php
19. L. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*, Prentice-Hall, 1994.
20. J. Heaton, *Introduction to Neural Networks for C#"*, St Louis: Heaton Research, Inc, 2008.
21. Y. Chauvin, D. E. Rumelhart, *Back Propagation Theory, Architectures and Applications*, Psychology Press, 1995.

22. R.W. Brause, “Medical Analysis and Diagnosis by Neural Networks”, *Proceedings of the Second International Symposium on Medical Data Analysis*, Lecture Notes in Computer Science, Vol. 2199, pp.1-13, Springer-Verlag, 2001.

23. American Diabetes Associates Ana sayfası: www.diabetes.org

24. T. Gale, *The Gale Encyclopedia of Alternative Medicine*, Farmington Hill: The Thomson Corporation, 2005.