
GAZİ ÜNİVERSİTESİ
Gazi
1926

M2M PLATFORMLAR İÇİN SERVİS VE ARAYÜZ GELİŞTİRİLMESİ

Saadin OYUCU

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2015

Saadin OYUCU tarafından hazırlanan "M2M PLATFORMLAR İÇİN SERVİS VE ARAYÜZ GELİŞTİRİLMESİ" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Teknoloji Fakültesi Anabilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Yrd. Doç. Dr. Hüseyin POLAT

Bilgisayar Mühendisliği, Teknoloji Fakültesi, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum ..……

Başkan: Prof. Dr. O. Ayhan ERDEM

Bilgisayar Mühendisliği, Teknoloji Fakültesi, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum ……...

Üye: Yrd. Doç. Dr. Taner TOPAL

Bilgisayar Mühendisliği, Mühendislik Fakültesi, Kırıkkale Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum ……...

Tez Savunma Tarihi: 12/06/2015

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..…….

Prof. Dr. Şeref SAĞIROĞLU

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Saadin OYUCU

12/06/2015

M2M PLATFORMLAR İÇİN SERVİS VE ARAYÜZ GELİŞTİRME

(Yüksek Lisans Tezi)

Saadin OYUCU

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Haziran 2015

ÖZET

Bu çalışmanın amacı gelişen teknoloji sayesinde maliyeti azalan ve sayıları gittikçe artan Makineler Arası Haberleşme (M2M: Machine to Machine) cihazlarını kullanarak geliştirilen M2M sistemleri için standartlara uygun ve ölçeklenebilir bir web tabanlı M2M platformu geliştirmektir. Geliştirilen bu platform yalnızca özel bir alana değil değişik alanlarda hizmet verebilecek şekildedir. Böylelikle çeşitli alanlarda geliştirilen M2M projeleri için hazır bir alt yapı sunularak, proje geliştirme sürelerinin kısaltılması sağlanmaktadır. Bu çalışma kapsamında Avrupa Telekomünikasyon Standartları Enstitüsü (ETSI: European Telecommunications Standards Institute) ve OneM2M standartları kullanılmıştır. Sistem Web 3.0 standartları ve Service Tabanlı Mimari (SOA: Service Oriented Architecture) yaklaşımı kullanılarak duyarlı bir tasarımla geliştirilmiş, metotlara ve verilere erişim web servisler ile servis katmanı üzerinden sağlanmıştır. Kullanılan web servisleri Temsili Durum Transferi (REST: Representional State Transfer, RestFul Web Services) yaklaşımıdır. Veriler ise ilişkisel olmayan veri tabanlarında saklanarak hem maliyet hem de ölçeklenebilirlik ön plana çıkarılmıştır. Çalışmanın sonunda bir M2M platformu geliştirilmiştir. M2M platformu geliştirilirken ilişkisel olmayan veri tabanı ve RestFul web servislerinin kullanılması sayesinde hızlı, ölçeklenebilir, cihaz ve kullanıcıdan bağımsız hizmet verebilecek bir M2M platformu sunulmuştur.

Bilim Kodu : 902.1.014
Anahtar Kelimeler : M2M, RestFul, IoT, NoSQL, SOA
Sayfa Adedi : 85
Danışman : Yrd. Doç. Dr. Hüseyin POLAT

SERVICE AND INTERFACE DEVELOPMENT FOR M2M PLATFORMS

(M. Sc. Thesis)

Saadin OYUCU

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

June 2015

ABSTRACT

The aim of this study is to develop a web based M2M (Machine to Machine) platform which is conforming to the standards and scalable. It is worth mentioning that the costs of M2M systems are decreasing while its use is increasing. This platform is not planned specifically for a certain area but rather it will serve for various fields of use. As a result; an infrastructure will be ready for the M2M projects that will be developed in different fields, thus the project development duration will be shortened. In this study European Telecommunications Standards Institute (ETSI) and OneM2M standards were used. The system was developed through a sensitive design by using Web 3.0 standards and Service Oriented Architecture (SOA) approach, and access to the methods and data was enabled through service layer. Representational State Transfer (RestFul Web Services, REST) are the web services used in this study. The data are stored in unrelated databases, by doing so both cost effectiveness and scalability are highlighted. At the end of the study an M2M platform is developed. By means of unrelated database and RestFul Web Services used in the development phase, fast and scalable M2M platform is developed which will serve independent of device and user.

Science Code : 902.1.014
Key Words : M2M, RestFul, IoT, NoSQL, SOA
Page Number : 85
Supervisor : Assist. Prof. Dr. Hüseyin POLAT

TEŞEKKÜR

Çalışmalarım boyunca yardım ve katkılarıyla beni yönlendiren değerli hocam Yrd. Doç. Dr. Hüseyin POLAT'a, yine kıymetli tecrübelerinden faydalandığım Gazi Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Bölümü öğretim üyelerine, web servisler hakkında yardım aldığım değerli arkadaşım LST Yazılım A.Ş. yazılım direktörü M. Uygar AKGÜL ve manevi destekleriyle beni hiç yalnız bırakmayan eşim Yurdanur OYUCU'ya teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

RESİMLERİN LİSTESİ

Resim **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklamalar
API	Uygulama Programlama Ara yüzü
ASP	Microsoft'un Etkin Sunucu Sayfaları
Bayt	Bellek Ölçü Birimi (1 Bayt = 8 Bit)
BTK	Bilgi Teknolojileri Kurumu
COOS	Bağlı Nesneler İşletim Sistemi
CORBA	Ortak Nesne İstem Aracı Mimarisi
CSS	Basamaklı Stil Şablonları
DCE	Dağıtık İşlem Ortamı
DCOM	Dağıtık Bileşen Nesne Modeli
ETSI	Avrupa Telekomünikasyon Standartları Enstitüsü
GSM	Mobil İletişim İçin Küresel Sistem
HTML	Hiper-Metin İşaretleme Dili
HTTP	Hiper-Metin Transfer Protokolü
ID	Kimlik
IoT	Nesnelerin İnterneti
IP	İnternet Protokolü
IT	Bilgi Teknolojileri
ITU	Uluslararası Telekomünikasyon Birliği
J2EE	Java Enterprise Edition
JSON	Javascript Nesne Gösterimi
JSP	Java Sunucu Sayfaları
KB	Kilobayt
MB	Megabayt
M2M	Makineler Arası Haberleşme
MAC	Ortam Erişim Kontrolü
MEMS	Mikro Elektro Mekanik Sistemleri
MHz	Megahertz
ms	Milisaniye

MVC	Spring MVC (Model-Görünüm-Kontrolcü)
NoSQL	İlişkisel Olmayan Veri Tabanı
OECD	Ekonomik İşbirliği ve Kalkınma Örgütü
PHP	Üstün yazı Ön işlemcisi
PSTN	Genel Aktarmalı Telefon Şebekesi
POS	Satış Noktaları Terminali
REST	Temsili Durum Transferi
RESTFUL	Temsili Durum Transferi Web Servis Yaklaşımı
RMI	Uzak Yordam Çağrımı
RPC	Uzaktan Yordam Çağrısı
SD	Güvenli Sayısal Hafıza Kartı
SOA	Service Tabanlı Mimari
SOAP	Basit Nesne Erişim Protokolü
SQL	Yapısal Sorgu Dili
TCP	İletim Kontrol Protokolü
UDDI	Evrensel Tanımlama, Bulma ve Entegrasyonu
URL	Tekdüzen Kaynak Bulucu
USB	Evrensel Seri Veri Yolu
Wi-Fi	Kablosuz Bağlantı Alanı
WPAN	Kişisel Alan Ağı
WSDL	Web Servisleri Açıklama Dili
XML	Genişletilebilir İşaretleme Dili

1. GİRİŞ

Makineler Arası Haberleşme (M2M: Machine to Machine), farklı cihazların kablolu ya da kablosuz bir şekilde haberleşmesini sağlayan teknolojiyi ifade etmektedir [1]. Günümüzde M2M kavramını özel bir alana sığdırmak doğru değildir. Haberleşme teknolojilerindeki mevcut durum ve ağ alt yapısındaki gelişmelere uygun olarak M2M kavramının önemi gittikçe arttırmıştır. Nitekim Ekonomik İşbirliği ve Kalkınma Örgütü (OECD: Organisation for Economic Co-operation and Development) M2M kavramını kablolu ve kablosuz şebekeler üzerinden sürekli iletişimde olan ancak bilgisayar sınıfına girmeyen cihazlar aracılığıyla genellikle internet ve benzeri yapılar üzerinden gerçekleştirilen iletişim olarak tanımlarken [1], Uluslararası Telekomünikasyon Birliği (ITU: International Telecommunication Union) aynı kavramı iki ya da daha fazla makinenin insan ihtiyacı olmadan ya da minimum insan ihtiyacıyla birbiriyle haberleşmesi olarak tanımlamaktadır [1].

M2M sistemlerin tarihi endüstriyel otomasyon sistemlerine kadar dayanmaktadır. 1970'li yılların başından itibaren ivme kazanan ve seri üretimlerini arttırmaya yönelik çalışmaların yapıldığı otomotiv sektörü M2M kavramının temeli olarak anılmaktadır. Haberleşme altyapılarının gelişmesi, bilgi teknolojisinin önem kazanması ve kullanılan cihazların maliyetlerinin azalması M2M üzerine yapılan çalışmaların artmasında önemli rol oynamıştır. Otomasyon sistemlerinin pahalılığı, ağır olması nedeniyle taşınmasının zor olması ve zaman içerisinde gelişen İnternet Protokolü (IP: Internet Protocol) tabanlı şebekeler ile veri iletiminin hızlı bir şekilde gerçekleştirilmesi M2M sistemlerin internet üzerinden kolayca haberleşmesini sağlamıştır. Böylelikle hem ucuz hem de daha fonksiyonel sistemlerin geliştirilmesinin önü açılmıştır.

Son yıllarda MEMS (Mikro Elektro Mekanik Sistemleri) teknolojisindeki gelişmeler ile küçük boyutta ve oldukça fonksiyonel mikro aygıtların gelişmesi günlük hayatın her alanına algılayıcıların girmesine olanak sağlamıştır. Genel olarak otomotiv sektöründe karşımıza çıkan bu algılayıcıların maliyetinin düşürülmesi ile her alanda kullanılmaya başlanmış ve günümüzde küçük boyutları sayesinde cep telefonlarına kadar girmiştir [2]. Kızılötesi görüntü alabilen kamera sistemlerinin boyutu küçülerek cep telefonlarında yerini

almıştır [3]. Algılayıcılardaki bu gelişmelerin M2M sistemler üzerine olumlu katkısı olmuş ve M2M uygulamalar oldukça yaygınlaşmıştır.

Dünya genelinde sayıları gittikçe artan M2M uygulamalar mimari olarak üç temel adımdan oluşmaktadır. Bunlardan ilki cihaz alanı olarak da bilinen ve sisteme veri sağlayan M2M hizmetinin başlangıç noktasıdır. Bu noktada kullanıcının tetiklemesiyle yâda otomatik olarak sistem için talep edilen veri sağlanmaktadır. Sistem için uygun olarak geliştirilen protokoller sayesinde elektronik veri formatına dönüştürülen girdiler ikinci olarak verilerin iletildiği bir haberleşme ortamına aktarılır. Bu haberleşme ortamı vasıtasıyla veriler bir makineden başka bir makineye yâda değişik sistemlere aktarılmaktadır. Son olarak ise verilerin kullanıcıya yâda alıcı sistemlere sunulduğu alan bulunmaktadır. Bu alanda verinin işlenerek nasıl sunulacağı vb. hizmetler geliştirilmektedir. Hizmetler geliştirilirken ETSI veya OneM2M gibi dünya çapında standart geliştiren kuruluşların temel mimari ve senaryoları göz önünde bulundurularak sistemler geliştirilmektedir.

IP tabanlı şebekelerin yaygınlaşması ve maliyetinin düşürülmesi ile M2M sistemlerinin iletişim alanında IP tabanlı şebekelerin kullanılması yaygınlaşmıştır. Ülkemizde neredeyse her evde bir internet bağlantısı bulunmaktadır. Bu durum son kullanıcıların da M2M uygulamaları geliştirmelerinin önünü açmaktadır. Piyasada düşük maliyeti ile kolayca bulunabilecek M2M cihazları mevcuttur. Kullanıcı bu cihazları ve gerekli algılayıcıları alarak kendi M2M uygulamasını geliştirebilir. Fakat bu noktada M2M için hazır bir platformun olması gerekmekte ya da kullanıcı kendi Bilgi Teknolojileri (IT: Information Technology) alt yapısını oluşturmak zorundadır. Bu durum teknoloji konusunda profesyonel bir bilgi gerektirmektedir. Tez çalışması sonucunda ortaya çıkacak olan M2M platformu sayesinde son kullanıcılarında kolaylıkla M2M uygulaması geliştirmelerinin önü açılmış olacaktır. M2M uygulamaları için gerekli IT alt yapısı geliştirilen M2M platform tarafından karşılanabilecektir. Mevcut M2M platformları geliştirilirken farklı iletişim teknikleri, farklı servis yöntemleri ve farklı veri tabanı sistemleri kullanılmıştır. M2M platformlar üzerine yapılan çalışmalarda ise prototip olarak genellikle akıllı ev sistemleri, sıcaklık, nem veya gaz algılayıcıları üzerinden üretilen veriler iletişime alınmış, saklanmış ve kontrol edilmiştir.

S. Okay Tosunoğlu 2009 yılındaki yüksek lisans tezinde [4], akıllı ev çalışması yaparak ev ortamında bulunan çamaşır makinesi ile ilgili tüm bilgilerin erişebilir hale getirilmesi

üzerinde durulmuştur. Ev sahibi, yönetici ve servis yetkililerinin kullanımı için veri tabanında makine bilgilerinin saklanması çalışılmıştır. Çamaşır makinesinden alınacak veriler RS-232 seri haberleşme üzerinden gerçekleştirilmiştir. Çalışmanın büyük bölümünde veri alma ve gönderme işlemleri seri haberleşme üzerinden yapılmaktadır. Alınan verilerin veri tabanına kaydedilmesi için ise IP şebekeler kullanılmıştır ve seri Ethernet dönüştürücü kartlar çalışmaya dâhil edilmiştir. Bu çalışmada veri tabanı olarak ilişkisel bir veri tabanı kullanılmıştır. Uzak ortamdaki sunucuya veriler aktarıldıktan sonra uzak ortamdaki servis yetkilisine veriler masaüstü uygulaması şeklinde sunulmaktadır. Her bir makinenin akıllı olabilmesi için üreticiler tarafından benzersiz bir numara verilmesi önerilmektedir.

Zhang, Zhang ve Li M2M üzerine yaptıkları platform tasarımı çalışmasında [5], J2EE (Java Enterprise Edition) ve SOA yaklaşımı kullanılarak sistem geliştirmeye çalışmışlardır. Bu çalışmalarında dağıtık sistem M2M platformu önerilmektedir. M2M uygulamalarının yalnızca makinelerle değil çeşitli iş uygulamaları ile de entegre edilmesi gerektiği savunulmuştur. J2EE temel yapı taşlarını ve Spring iskeleti yardımıyla nesne tabanlı bir yaklaşım ile ilişkisel veri tabanı kullanarak M2M platformu geliştirilmeye çalışılmıştır. Servis odaklı mimariye göre geliştirilen sistemde mesajlaşma dili olarak Genişletilebilir İşaretleme Dili (XML: Extensible Markup Language) kullanılmıştır. M2M katmanları ve servisler bir sisteme yâda teknolojiye bağlı kalmayacak şekilde geliştirilmiştir. Böylelikle farklı teknolojiler kullanarak farklı alanlarda M2M uygulamaları geliştirilebilir olduğu savunulmuştur.

Youm ve Kim yatay hizmet entegrasyonu sağlayan M2M platformu mimarisi kullanılarak, prototip olarak ise bir akıllı ev projesi geliştirdikleri çalışmalarında [6], M2M uygulamalarını üç alana ayırmış ve sistemi her alan için ayrı ayrı ele almayı önermişlerdir. M2M hizmet platformu, M2M iletişim alanı ve kullanıcı/yönetici olarak alanlara ayırdıkları çalışmalarında açık kaynak kodlu Bağlı Nesneler İşletim Sistemi (COOS: **C**onnected Objects Operating System) kullanılmıştır. İki tip kullanıcı ara yüzü geliştirilmiştir. Biri web tarayıcı ara yüzü diğeri ise akıllı telefon mobil uygulamasıdır. Sistemde güvenlik çalışmaları yaparak yeni bir cihaz eklendiğinde ilk önce cihaza kısa mesaj yoluyla ulaşıp doğrulanması gerekmektedir. Doğrulanan kullanıcılara kod parçacıkları verilerek M2M cihazlardan gelen verilerin depolanması sağlanmaktadır. Prototip geliştirilirken M2M cihaz alanında bulunan birden fazla cihazı sensör düğümleri

ile haberleştirip ardından ZigBee bağlantısı ile bir gateway cihazı kullanarak COOS mesajlaşma ile verileri bir üst katmana aktarmaktadır. Veri tabanına kaydedilen veriler hazırlanan ara yüzler sayesinde görüntülenebilmekte ve yönetilebilmektedir.

M. Soliman, T. Abiodun ve T. Hamouda Nesnelerin İnterneti (IoT: Internet of Things) kavramına web servisler ve bulut bilişim entegre edilerek gerçekleştirdikleri akıllı ev çalışmasında [7], bu alanda yapılan çalışmalardan farklı olarak bulut bilişimi sistemlerine entegre etmişlerdir. Bu çalışma kapsamında Javascript Nesne Gösterimi (JSON: Javascript Object Notation) veri mesajlaşma yöntemini ve bulut bilişim üzerindeki veri tabanını kullanmışlardır. İletişimde ise ZigBee kullanılmıştır. Arduino kart üzerinden alınan sıcaklık ve nem değerleri bulut üzerindeki servisler yardımıyla yine bulut üzerindeki veri tabanına kaydedilmektedir. Bu yaklaşımda her istek veya her iş bulut üzerinden yapılmaya çalışılmıştır.

Akın Aslan 2014 yılındaki yüksek lisans tezinde [8], akıllı ev ve otomasyon sistemlerini incelemiş ve insan hayatını kolaylaştıracak farklı uygulamalar ve otomasyonlar üzerinde durmuştur. Akıllı ev ve ev otomasyon sistemleri ayrı ayrı ele alınmış tarihçesinden, sunduğu faydalara kadar değinilmiştir. Ayrıca dünyada ve ülkemizde örnek akıllı bina uygulamalarına yer verilmiştir. Elektronik araçlar ve kullanım yerleri hakkında bilgilerin de sunulduğu bu çalışmada akıllı ev kavramlarının her yönden avantajlı olduğu ve modern mimarilere entegre edilerek yaygın kullanılması gerektiği savunulmuştur.

Omar Talal Algoiare, 2014 yılındaki yüksek lisans tez çalışmasında Mobil İletişim İçin Küresel Sistem (GSM: Global System for Mobile Communications) şebekesi kullanarak bir akıllı ev uygulaması geliştirmiştir[9]. Bu çalışmada cihaz olarak Arduino mega akıllı kartını, Arduino GSM modülünü ve çeşitli algılayıcıları kullanarak bir akıllı ev tasarlamıştır. Arduino kart üzerinden alınan algılayıcı bilgileri Arduino üzerinde bulunan Güvenli Sayısal Hafıza Kartı (SD: Secure Digital Memory Card) içerindeki metin dosyalarına kaydedilmekte ve istenilen durumlarda cep telefonuna kısa mesaj gönderilmektedir. Ayrıca çalışmada Visual Basic sürüm 6 ile geliştirilen bir ekran tasarımda veriler kullanıcıya sunulmaktadır. Bu çalışmada GSM operatörü ile geliştirilen bir sistemin esnekliği ve erişebilirliği vurgulanmıştır.

S. Lee, J. Jo, Y. Kim, Nevada Güneş Enerjisi-Su-Çevre projesinde çevresel izlenimlerin kaydedildiği büyük boyutlara ulaşan veriler için bir çalışma yapmış öncelikle geçerli algılayıcı sisteminin alt yapısını Arduino ile değiştirmiştir [10]. Algılayıcılardan alınan veriler belirli normalleştirme süreçlerinden sonra ilişkisel veri tabanına kaydedilmektedir. Bu çalışmada algılayıcılar için sensör ağlar kullanılmış ardından verilerin başka bir sunucu üzerinde tutulması ve analizini kolaylaştırmak adına ResTFul web servisleri geliştirilmiştir. Verilerin analizi için ise Google Chart kullanılmıştır. Tekdüzen Kaynak Bulucuya (URL: Uniform Resource Locator) dayalı olarak gerçekleştirilen sistemlerin güvensizliğine değinilen çalışmada algılayıcı verilerinin taşınması için RESTFUL web servislerinin diğer servis türlerine göre daha etkili olduğu savunulmuştur.

Literatürde yer alan çalışmalara bakıldığında çeşitli alanlarda M2M uygulamaları geliştirildiği ve bu uygulamaların büyük bir çoğunluğunu akıllı ev uygulamaları olduğu görülmektedir. Bu tez çalışmasında ise gelişen teknoloji alt yapısı ve teknoloji araçlarını kullanılarak ölçeklenebilirliği, ulaşılabilirliği, kullanılabilirliği ve performansı yüksek, farklı uygulamalar geliştirmeye uygun katmanlı bir mimariye sahip M2M platformu geliştirilmesi hedeflenmiştir. Platform geliştirilirken, günümüz teknolojilerine uygun ve diğer platform örneklerinden farklı olarak ilişkisel olmayan veri tabanları, RestFul web servis yaklaşımı, JSON mesajlaşma yöntemi, ETSI ve OneM2M standartları kullanılarak cihazdan ve algılayıcılardan bağımsız bir yaklaşım ile sıcaklık ve nem bilgileri saklanarak istenildiğinde kullanıcıya platformdan bağımsız bir şekilde sunulmaktadır. Böylelikle M2M sistemlerde gerekli olan bilgi teknolojileri için farklı bir sistem örneği geliştirilmiş ve prototip olarak sunulmuştur.

Tezin birinci bölümünde M2M hakkında kısa bilgilere yer verilerek literatürde yer alan çalışmalara değinilmiştir. Ayrıca tez sonucunda ortaya çıkacak olan M2M platformuna da yine bu kısımda yer verilmiştir. Tezin ikinci bölümünde M2M teknolojisi ele alınmış olup genel kullanım alanları, M2M standartları ve M2M mimarisi detaylı bir şekilde açıklanmıştır. İkinci bölümün son kısmında ise dünyada ve Türkiye'de M2M pazarı ve uygulama alanları ayrı ayrı ele alınıp grafik ve rakamlarla desteklenmiştir. Üçüncü bölümde M2M platformu geliştirilirken literatürde ve uygulamada farklı yaklaşımlar elde etmek amacıyla kullanılacak mesajlaşma biçiminin seçilmesi, kullanılacak veri tabanının seçilmesi, kullanılacak web servislerin belirlenmesi, kullanılan M2M cihazların belirlenmesi, algılayıcıların belirlenmesi aynı zamanda da ara yüz hazırlanırken

kullanılacak materyal ve metotlar açıklanmıştır. Dördüncü bölümde, üçüncü bölümde belirtilen materyal ve metotlar üzerine sistemin geliştirilmesi adım adım ele alınarak açıklanmış ve şekillerle desteklenmiştir. Ayrıca her adımda test işlemleri yapılmaya çalışılmış ve geliştirilen platformun kullanım senaryosu açıklanmıştır. Son olarak beşinci bölümde sonuç ve önerilere yer verilmiştir.

2. M2M TEKNOLOJİSİ

İletişim sistemleri, internet ve algılayıcı teknolojilerindeki gelişmelere paralel olarak M2M kavramı önemini gittikçe arttırmaktadır. M2M fikri düşük maliyetli, ölçeklenebilir ve güvenilir teknolojiler ile birbirine bağlı aynı zamanda da uzaktan kontrol edilebilen sistemler sunmaktadır. M2M teknolojisi ile cihazlar arasında iletişim sağlanarak, verimliliği arttırmak ve maliyeti düşürmek amacıyla birçok uygulama geliştirilmiştir. Bu uygulamaların bazıları kamu güvenliği, ulaşım, sağlık ve enerji yönetimi gibi alanlarda geliştirilen projelerdir.

M2M sistemlerin gelişmesi, yeni proje ve teknolojilere de öncülük etmektedir. Bunların IoT kavramı ve akıllı şebekeler (smart grid) gelmektedir. M2M, akıllı şebekeler ve IoT temelde birbirlerine oldukça benzemektedir. M2M makinalar arası iletişimi ifade ederken IoT, haberleşme kavramı içerisine gerçek hayattaki nesneleri katarak yelpazeyi biraz daha genişletmiştir. Akıllı şebekeler ise enerjinin çift yönlü iletimini ele alarak daha esnek, ölçeklenebilir, izlenebilir ve sorunsuz bir hizmet sunmayı ana hedefleri içerisine almıştır.

Algılayıcı teknolojilerinin ilerlemesi ile birlikte sistemler nem, sıcaklık, basınç, görüntü, hız vb. değerleri kolaylıkla ölçebilmektedirler. Teknoloji oldukça hızlı ilerlerken endüstri ve diğer alanlarda M2M sistemlerini kullanmak kaçınılmaz hale gelmiştir [11]. M2M haberleşmesinin gerçekleştirilebildiği bütün hizmet alanlarında uygulanılabilen M2M projeleri genel olarak aşağıdaki hizmet gruplarında kullanılmaktadır.

Üretim: Bu grupta kullanılan M2M sistemleri üretim aşamasında destek vermek için geliştirilen sistemlerdir. Bu sistemler tarım, hayvancılık, perakende ve benzeri sektörlerde uzaktan yönetim, bilgi alımı ve kontrol gibi ihtiyaçları karşılamak için geliştirilen sistemlerdir. Ayrıca yaygın kullanım alanlarından biride taşıt ve makine üretim sanayisidir. En güzel örneklerinden biri Ford firmasının Oakville, Ontario, Kanada'da yer alan Oakville Üretim Tesisidir. Söz konusu tesiste yer alan 440 adet robot M2M uygulamaları ile çalışanlara üretim sürecinde yardımcı olmaktadır [12].

Hizmet: Bu grupta kullanılan M2M sistemler iş süreçlerinin her aşamasında hizmet sunan ve alan kişi veya kişilere daha kaliteli ve hızlı erişimi sağlamayı hedeflemektedir. Son

yıllarda hayatımızda sıklıkla kullandığımız mobil ödeme Satış Noktaları Terminali (POS: Point of Sales Terminal) cihazları sayesinde sunulan hizmetin kalitesi artmış ve hizmet almak isteyenler için işlem kolaylığı sağlanmıştır. İngiltere menşei olan Verifone firması mobil ödeme alanında POS cihazlarının gelişmesinde öncü firmalardan biridir. POS çözümlerinde kablosuz M2M yaklaşımını kullanmaktadır. Geliştirdiği birden fazla M2M sistem sayesinde hem hizmet sunan kişilerin maliyetini azaltmış hem de hızlı ve kaliteli hizmet anlayışını arttırmıştır [13].

İzleme: Bu grupta bulunan M2M sistemler insanların izleme gereksinimi duyduğu tüm alanlara uygulanabilir. Kamera sistemleri, meteorolojik izlemeler, akıllı sayaçlar (gaz, su, elektrik), parkmetreler ve sağlık hizmetleri bu gruptaki M2M uygulamalarından bazılarıdır. Örneklerinden bir tanesi Quebec Kanada'da hizmet veren Hydro Quebec isimli su dağıtım firmasıdır. Söz konusu firma yerel elektronik haberleşme hizmeti sağlayıcılarından biri olan Rogers Telecom ile birlikte geliştirdiği ortak bir platform aracılığıyla 3,8 milyon akıllı su sayacının verisini 600 adet akıllı toplama birimine entegre ederek merkezi bir sistem oluşturmuş ve su sayaçlarındaki kaçakların büyük ölçüde önüne geçmiştir [12].

Akıllı Enerji Sistemleri: Bu grupta bulunan M2M sistemleri enerji tarifesi, elektrik, su ve doğalgaz sayaçları ile enerji sektöründe kullanılan enerji analizörü gibi algılayıcı ve cihazlara entegre edilerek M2M sistemlere aktarılarak kullanılır. Bu sayede elektrik, su ve doğalgaz üretimi, dağıtımı veya iletimi yapan kuruluşlar, sayaç ve algılayıcıların uzaktan okuma ve açma/kapama işlemlerini gerçekleştirebilir. Ayrıca sayaçlara ait endeks, tarife, aktif/reaktif gibi parametreler bu sayede uzaktan izlenebilir depolanabilir ve analiz edilebilir. Günümüzde birçok firma akıllı ev otomasyonları sunmaktadır.

Reklam: Bu grupta kullanılan M2M sistemler gelişen ve sürekli büyüyen ticari faaliyetlerin içerisinde farkındalığı ve görülebilirliği arttıran reklam sektörüne hizmet etmesi için geliştirilen uygulamalardır. Bu uygulamalardan bazıları; elektronik reklam panoları ve mobil reklam uygulamaları olarak göze çarpmaktadır. Anlamsal ilişkilerin geliştiği günümüz teknolojisinde reklamcılık anlayışı da farklı bir boyut kazanmış ve doğrudan hedef kitleye ulaşmak oldukça kolaylaşmıştır. M2M sistemlerin bu konudaki katkıları göz ardı edilemeyecek kadar fazladır. Amerikan mobil cihaz firması olan Sprint geliştirdiği akıllı reklamcılık projeleriyle reklam panolarında otomatik güncelleme sürecini kolaylaştırmış ve hızlandırmış, günün belli saatlerine göre açılan ve kapanan reklam

panosu tasarımıyla fazla enerji tüketiminin önüne geçme yolunda da önemli bir adım atmıştır. Ayrıca mobil reklam ağını geliştirerek hedef kitle belirlemesinde oldukça başarılı projeler üretmiş ve hizmete sunmuştur [14].

Lojistik: Bu grupta yer alan M2M uygulamaları, kullanan kurum veya kuruluşlara tasarruf sağlamakta ve hizmet kalitesini arttırmaktadır. Taşınan malzemelerin yalnızca araçlarda değil depolarda da kontrolünün sağlanması adına bu uygulamalar kullanılmaktadır. Araç filolarına ait yol bilgileri, seyir hızı, kat edilen mesafe, yakıt tüketimi, rota bilgileri ve benzeri bilgilerin anlık olarak yönetici veya yetkili kişilere sunularak lojistik yönetimi kolaylaştırılmıştır [12].

2.1. M2M Standartları

Makine haberleşmesi son yıllarda dünya genelinde yoğun ilgi görmüştür. Bu ilgi her ülkenin kendine ait bir M2M politikası izlemesine neden olmuştur. Bu politikaların bir gereği de oluşturulan sistemlerin birer standarda dayanmasıdır. Bu nedenle her ülke M2M sistemleri için kendi standardizasyon çalışmalarını yapmıştır. Bu karmaşıklığın önüne geçmek için bazı kuruluşlar dünya çapında standartlar üretmektedirler. Böylece dünya çapında ortak bir M2M standardının oluşturulması amaçlanmaktadır.

M2M üzerine 2012 yılına kadar dünya genelinde standartlar geliştiren ETSI akıllı ölçüm durumları, akıllı ev sistemleri ve ev enerji yönetim sistemleri üzerine birçok çalışma yapmış ve standartlar üretmiştir. 2010 yılında ETSI, ETSI TR 102 691 numaraları teknik raporunda “Akıllı Ölçüm Kullanım Durumları” nı yayınlamış ve 2012 yılında ETSI TR 102 935 V2.1.1 numaralı teknik raporunda “Akıllı Şebeke Ağlarında M2M Mimarisi Uygulanması ve Akıllı Şebeklerin M2M Platformlara Etkisi” üzerine çalışmalar yapmıştır [15]. M2M üzerine yaptığı çalışmaları Akıllı Şebekelere de uyarlayan ETSI M2M standartları üreten kuruluşların başını çekmektedir.

ETSI’nin “Smart M2M” Teknik Komitesi akıllı cihazlar arasındaki haberleşmeye yönelik standartlar geliştirilmesi için bir çalışma başlatmıştır. Bu çalışmadaki standartlar, ETSI’nin M2M mimarisine dayalı, ortak bir veri modelini ve haberleşme protokollerinin tanımlanmasını içermektedir. Standartların uygulandığı sistemler farklı hizmet platformları

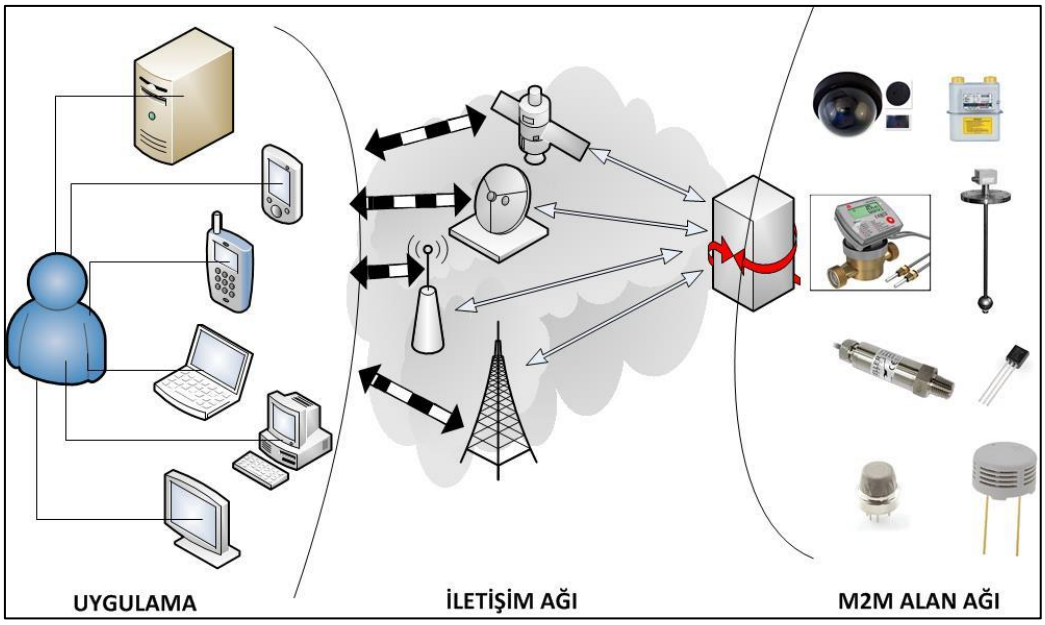
ile haberleşme sağlayabiliyor nitelikte olacaktır. ETSI organizasyonunun 27-28 Mayıs 2014 tarihinde Brüksel'de gerçekleştirdiği çalıştayda ETSI'nin standart çalışmalarındaki amaçları, Avrupa Komisyonu çalışmalarının kapsamı, endüstrinin ve ilgili paydaşların konuya olası katkıları ele alınmış ve değerlendirilmiştir [16].

M2M üzerine standart geliştiren diğer bir organizasyon olan OneM2M organizasyonu, M2M alanında çeşitli alanlarda hizmet vermektedir. M2M özelliklerinin uygulanması ve standartlar çerçevesinde geliştirilmesi adına çalışmalar yapmaktadır. Bu bağlamda dünyanın ilk küresel çaptaki M2M standardını yayınlamıştır [17]. Çalışmalarını öncelikle M2M mimarileri üzerine yapan OneM2M sonraki çalışmalarında farklı alanlarda geliştirilen M2M uygulamalar için gereken gereksinimleri yayınlamıştır.

Makine haberleşmesi üzerine geliştirilen uygulamaların çokluğu ve karmaşıklığı bununda yanında güvenlik, yöntem ve mimari bakımından farklılıklar standartlaşmayı önemli kılmıştır. ETSI ve OneM2M bu standartların geliştirilmesinin öncülerindendir. Bu tez çalışmasında OneM2M standartları ve sistem mimarisi kullanılmıştır.

2.2. M2M Mimarisi

Hâlihazırda teknolojik ilerlemeye paralel olarak M2M üzerine geliştirilmiş birçok uygulama bulunmaktadır. Gerek uygulamaların çokluğu gerekse de farklı mimarileri bir arada kullanma isteği M2M üzerine yapılan çalışmaların bir standart haline getirilmesini kaçınılmaz kılmıştır. Standart haline getirilmesi gereken bir konu da M2M mimarisidir. Küresel çapta standartlar üreten OneM2M'e göre M2M mimarisi ana yapı olarak üç temel katmandan oluşmaktadır [18]. Birinci katmanda M2M sistemi için veri sağlayan algılayıcılar bulunmaktadır. Bu algılayıcılar M2M sistemi için gerekli olan verileri aktarmaktadırlar. Örneğin ısı, nem, gaz algılayıcıları, kamera, akıllı telefonlar vb. cihazlar bu katman için verilecek örneklerden birkaçıdır. Veriler sisteme aktarılırken ikinci katman olan İletişim Ağı yardımıyla istenilen alana aktarım gerçekleşir. Bu alan insan etkileşimin olduğu sistemler ya da doğrudan haberleşme yeteneğine sahip makineler olabilmektedir. İletişim Ağı üzerinden veriler anlamlı birer bilgi olarak son kullanıcıya sunulmalıdır. Bu noktada ise üçüncü katman olan Uygulama Katmanı devreye girmektedir. Burada kullanıcılar verileri incelemekte, analiz etmekte ve gerektiğinde M2M sistemine komut verebilmektedirler. Şekil 2.1.'de yukarıda sayılan katmanlar şematik olarak gösterilmiştir.



Şekil 2.1. Temel M2M Mimarisi

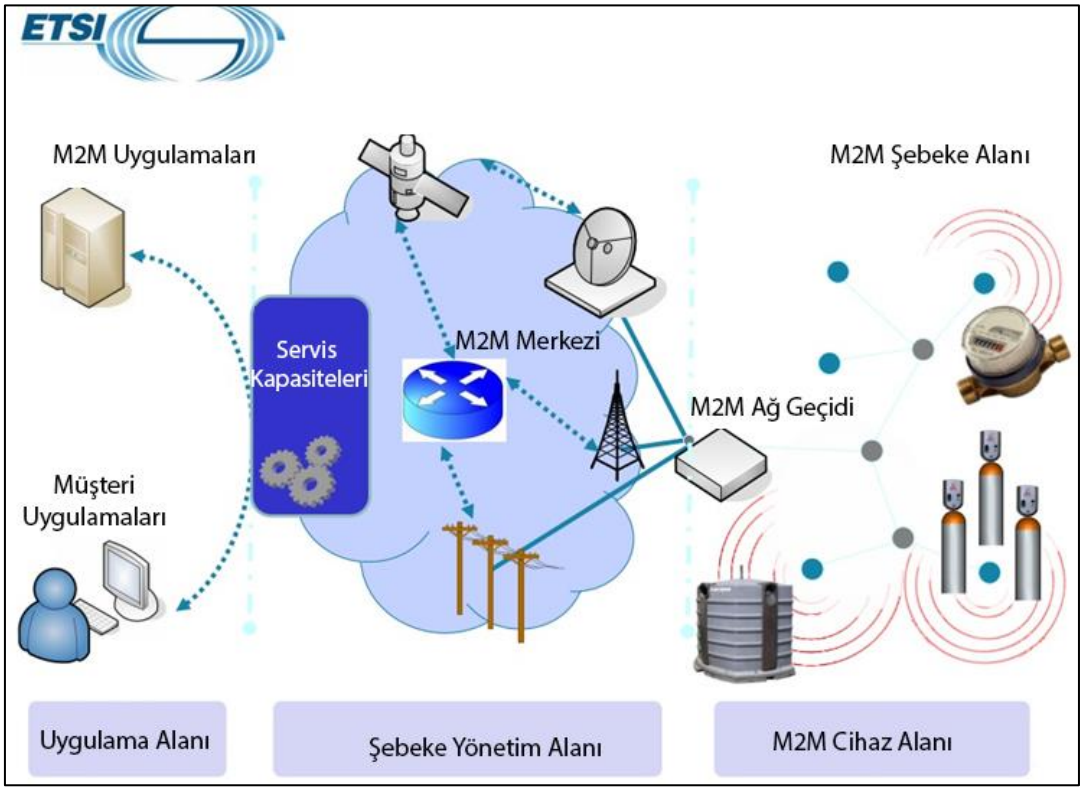
Şekil 2.1.'de görüldüğü gibi her adımda farklı teknolojiler kullanılabilmektedir. Örneğin tarım sektöründe kullanılmak üzere geliştirilen bir M2M sisteminde buğday depolarının sıcaklık değerinin sürekli izlenebilmesi istenilebilir. Bu durumda M2M alan ağı için sıcaklık değerini ölçen bir elektronik cihaz, iletişim ağı için ise ihtiyaca uygun bir altyapı ve uygulama adımı için ise son kullanıcının kontrol sağlayacağı ekranların tasarlanması gerekmektedir. Her iş alanı için farklı adımlardan oluşan bir M2M sistemi aynı zamanda her adımda da farklı teknoloji seçeneklerinden oluşabilir. Örneğin iletişim ağı adımında birbirinden farklı teknolojiler kullanılabilir, kablolu veya kablosuz altyapı kullanılabilir. Uygulama adımında ise kullanıcı her yerden sisteme ulaşmak ve kontrol etmek isteyebilir. Bu noktada ise kullanıcı tablet, akıllı telefon veya masaüstü bilgisayardan erişim yapmak isteyebilir. Bu yüzden sistem oluşturulurken bu üç husus göz önüne alınmalı ve kullanıcının isteği asla göz ardı edilmemelidir.

Bir web platformu üzerinden çift yönlü ses ve görüntü aktarımı insandan insana iletişimin temelini oluştururken, gelişen teknoloji sayesinde makine-insan ve makine-makine etkileşimi de giderek yaygınlaşmaktadır. Geleneksel olarak basit iletişim protokolleri ile gerçekleştirilen iletişimler artık IP tabanlı şebekeler ile yüksek performansta gerçekleştirilebilmektedir. Çizelge 2.1.'de sabit ve sabit olmayan M2M cihazlarının bağlantı türüne göre seçilebilecek şebeke yapıları gösterilmektedir.

Çizelge 2.1. Şebeke yapıları [12]

	Sabit Cihazın Bağlantısı	Sabit Olmayan Cihazın Bağlantısı
Dağınık	PSTN Sabit Geniş bant 2G/3G/4G Enerji Hattı İletişimi	2G/3G/4G Uydu
Merkezi	Kişisel Kablosuz Ağ Alanı (WPAN) Kablolu Şebekeler Bina İçi Enerji Kabloları Wi-Fi	Wi-Fi Kişisel Kablosuz Ağ Alanı (WPAN)

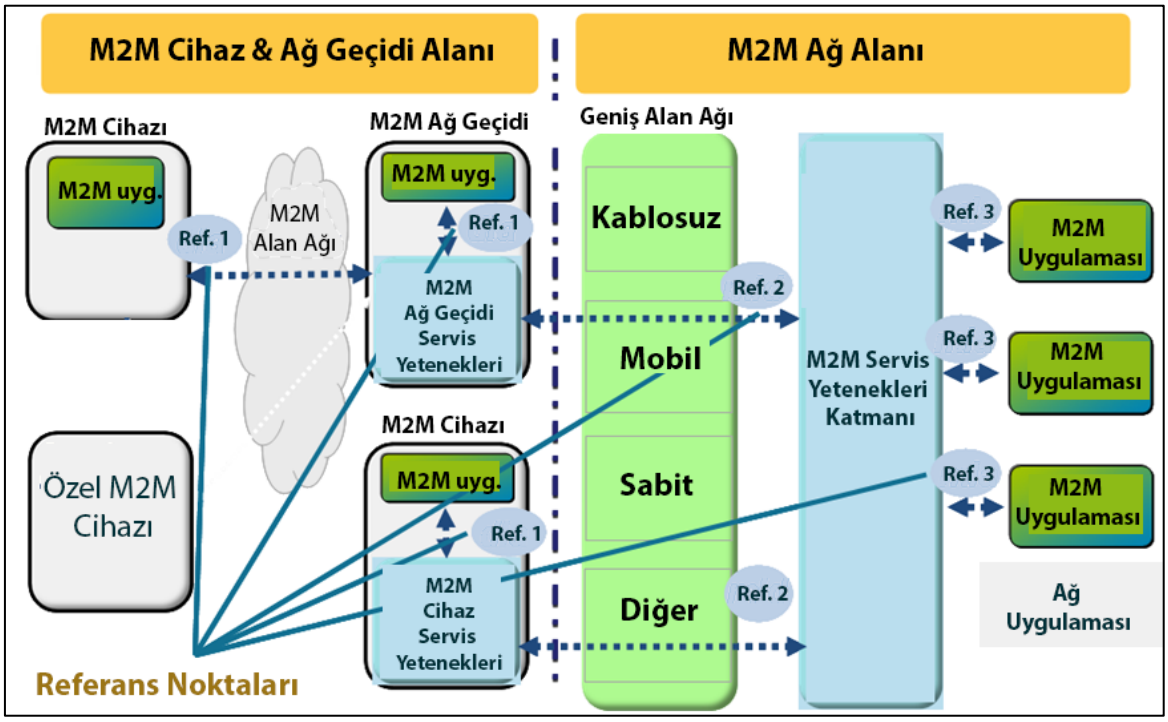
M2M üzerine standartlar üreten kuruluşlardan biri olan ETSI M2M mimarisini şekil 2.2.'de gösterildiği gibi alanlara ayırmaktadır.



Şekil 2.2. ETSI temel M2M mimarisi [12]

Şekil 2.2.'de gösterildiği gibi ETSI'ye göre M2M mimari tasarımı, M2M Cihaz Alanı, Şebeke Yönetim Alanı ve Uygulama Alanı olmak üzere üç temel kısımdan oluşmaktadır.

Tek bir M2M cihazı doğrudan Şebeke Yönetim Alanına ağ geçidi üzerinden bağlı olabileceği gibi birden çok M2M cihazı bir araya gelip kendi aralarında bir ağ oluşturup daha sonra yine bir ağ geçidi üzerinden M2M ağ alanına bağlanabilirler. M2M servis yetenekleri tek bir cihaz üzerinde barındırılabileceği gibi aynı zamanda da farklı cihazlara dağıtılıp tek bir noktadan kontrol edilebilir. Fakat ETSI ve OneM2M standartları bize servis katmanının ayrı bir yerde toplanması gerektiğini söylemektedir. Servis katmanındaki fonksiyonlar cihazlar veya ağ geçidi üzerindeki M2M uygulamaları tarafından da kullanılabilir. Kendi aralarında yerel ağ oluşturan cihazlara örnek olarak Zigbee, Bluetooth gibi teknolojileri barındıran cihazlar verilebilir. Bu cihaz gruplarının geniş alan ağlarına erişimi ise IP şebekelerle rahatlıkla yapılabilmektedir. Şekil 2.3.'te ise OneM2M'in M2M mimarilerini incelediği dokümanında yer alan ve diğer mimarilere göre daha fonksiyonel olan bir mimari yapısı görülmektedir.



Şekil 2.3. ETSI fonksiyonel M2M mimarisi [18]

OneM2M'in M2M mimarilerini incelediği dokümanında ETSI'nin fonksiyonel M2M mimarisi şekil 2.3.'teki gibi gösterilmektedir. M2M Cihaz & Ağ geçidi Alanı ve M2M Ağ Alanı olmak üzere temelde ikiye ayrılan mimari her alan içerisinde farklı alt alanlara ayrılmaktadır. Özellikle basit mimariden farklı olarak M2M servis yetenekleri katmanının devreye alınması ile ölçeklenebilirlik bir kat daha arttırılmış, farklı cihazlar ve farklı

uygulamalar geliştirilmesini kolaylaştırmıştır. Şekil 2.3.'te gösterilen referans noktaları M2M uygulamalarının birleşim noktaları olarak düşünülebilir ve bu noktalarda yapılabilecek değişiklikler diğer noktaları etkilemeyecek fakat işlevselliğini ve erişilebilirliğini kullanılan teknolojiye göre daha avantajlı veya dezavantajlı hale getirecektir. Örneğin referans noktası 1 M2M alan ağı için bluetooth veya IP tabanlı bir şebeke kullanılabilir. Bluetooth kısa mesafede iletişim sağlarken IP tabanlı şebekelerde daha uzun mesafeler ve daha hızlı veri iletişimi sağlanabilir. Bu referans noktasında seçilebilecek teknoloji diğer referans noktalarındaki iletişimi etkilemeyecek fakat sistem bir bütün olarak ele alındığında doğru seçimler kullanıcıya hız, sağlamlık, süreklilik ve esneklik olarak geri dönecektir. M2M uygulamalarından ve M2M cihazlarından gelen istekler M2M servis yetenekleri katmanında yorumlanır ve yapılması gereken iş gerçekleştirilir. Arttırılması istenilen veya yeni eklenecek olan fonksiyonlar servis katmanına eklenerek sistem rahatlıkla genişletilebilir. Bu kullanım şekli OneM2M'in de desteklediği kullanım şekli olup ölçeklenebilirliği oldukça yüksek bir mimaridir.

2.3. Dünyada ve Türkiye'de M2M Pazarı ve Uygulama Alanları

Bu bölümde dünyada ve ülkemizde M2M pazarı ve uygulama alanları açıklanmaya çalışılmıştır.

2.3.1. Dünyada M2M Pazarı ve Uygulama Alanları

Teknolojik ilerlemeler ve teknoloji araçlarının kullanılmasının yaygınlaşması ile M2M her alanda karşımıza çıkmaktadır. Özellikle endüstri alanının vazgeçilmezi olan M2M uygulamalarını OneM2M çizelge 2.2.'de görüleceği gibi farklı endüstri alanlarında farklı kullanım alanlarına göre gruplamıştır.

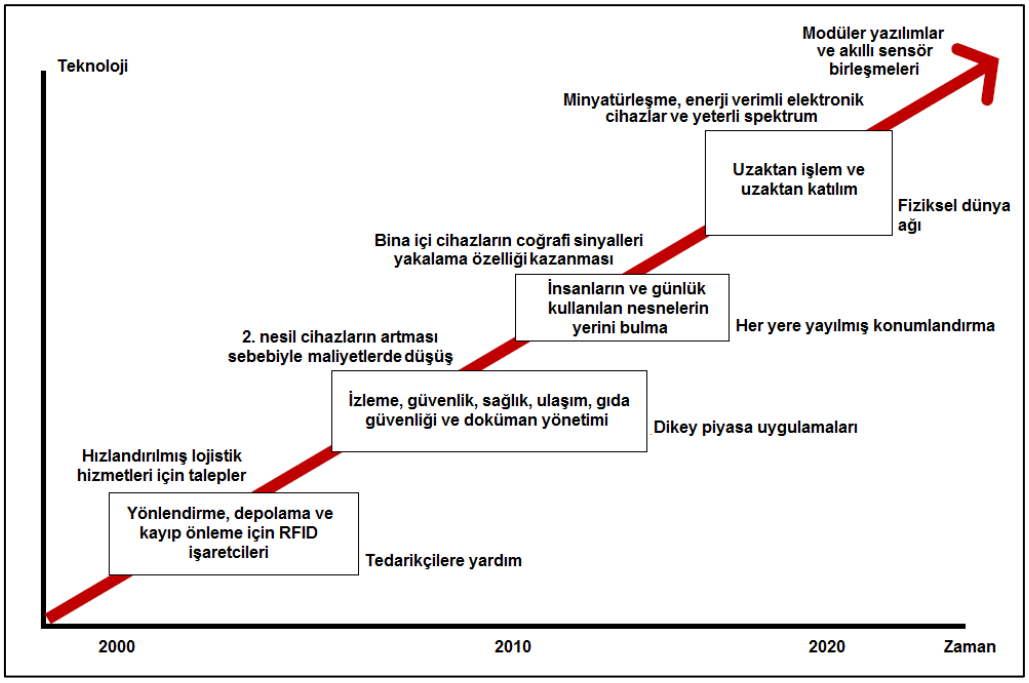
Çizelge 2.2. M2M kullanım alanları [19]

Endüstri Alanı	OneM2M Kullanım Alanları				
Enerji	Akıllı Şebekeler, Enerji otomasyon uygulamaları	M2M için analizler	Akıllı ölçümleme ve okuma	Hidro Elektrik santralleri izlemeleri	Petrol ve Gaz boru hattı kontrolü
Kurum ve Kuruluşlar	Akıllı Binalar				
Sağlık Hizmetleri	M2M Sağlık hizmetleri uygulamaları	Sağlıklı yaşam hizmetleri	Güvenli ve uzaktan hasta takibi ve bakımı		
Kamu Hizmetleri	Sokak aydınlatma otomasyonu	Cihazlar, Sanal cihazlar ve nesnelerin kontrolü	Araç/Bisiklet paylaşım servisi uygulaması	Akıllı Park sistemleri	
Konut	Ev Enerji yönetimi	Ev Enerji yönetim sistemi	Gerçek zamanlı ses ve video iletişimi	Olay tetikleme ve görev yürütme	Anlam-sal ev kontrolü
Taşımacılık	Araç teşhis ve bakım raporu	Uzaktan araç yönetim uygulamaları	Trafik kazası üzerine komşu uyarma	Dijital takograf ile filo yönetim hizmetleri	
Diğer	Uydu ağlarının kullanılması ile M2M erişim ağlarının genişletilmesi	Ağ operatörleri üzerinden veri trafiğinin yönetilmesi	Mobil ağlar ile bağlantı parametrelerini n optimize edilmesi	M2M sistem verilerinin toplanması	

Türkiye Cumhuriyeti'nde telekomünikasyon sektörünü düzenleme ve denetleme yetkisine sahip olan Bilgi Teknolojileri ve İletişim Kurumu (BTK) tarafından M2M kapsamında hazırlanan 2013 yılındaki raporda Ericsson firmasının 2020 yılı M2M cihaz sayısı tahminlerine yer verilmektedir. Rapora göre 2020 yılında 50 milyar mobil cihazın internete bağlı olabileceği belirtilmektedir [12]. Ayrıca Microsoft'a göre 2020 yılında nesnelerin internetinin parçası olması ön görülen nesne sayısı 30 milyar civarındadır [20].

M2M uygulamaları mobil ve sabit ağ şebekelerinin gelişmesi ile doğru orantılı olarak artmaktadır. Teknolojinin ucuza mal edilebilmesi ise M2M uygulamalarının gelişmesine katkıda bulunan diğer bir unsur olarak görülmektedir. ETSI 2020 yılındaki M2M uygulamaları hedefi için mobil şebeke yapısının yetersiz kalacağını belirtmektedir [15]. Bu nedenle farklı organizasyonlar mobil ve uydu kullanan ağ şebekelerini geliştirmek için çeşitli çalışmalar yapmaktadırlar.

M2M uygulamalarının temeli olarak kabul edilen otomotiv sektörü başta olmak üzere zaman içerisinde M2M uygulamaları kullanım alanlarına göre farklılıklar göstermektedir. Gerek bağlantı gerekse algılayıcı teknolojisindeki ilerlemelere göre kendini geliştiren M2M uygulamaları 2000'li yılların başından başlayarak 2020 yılına kadar uygulamalarda hangi unsurları içereceği şekil 2.4.'te gösterilmektedir.



Şekil 2.4. M2M'in gelişimine yönelik beklentiler [12]

M2M uygulamalarının yaygınlaşması ile IP tabanlı şebekelerin bant genişliği ihtiyacı katlanarak artmaktadır. Bu artışı karşılayabilmek için hem sabit hem de mobil şebekeler kendini yenilemekte ve yeni teknolojiler sunmaktadırlar. Gelişen teknoloji ile sadece insandan insana iletişim değil makineden makineye ve makineden insana iletişimde önemli ölçüde hayatımızda yerini almaktadır. Dünya genelinde haberleşme yönleri ile ilgili yapılan araştırmada insandan insana, insandan sunucuya/makinaya ve makinadan makinaya iletişimin örnekleri çizelge 2.3.'te gösterilmiştir.

Çizelge 2.3. Farklı yönlerde kullanılan iletişimler ve örnekleri [21]

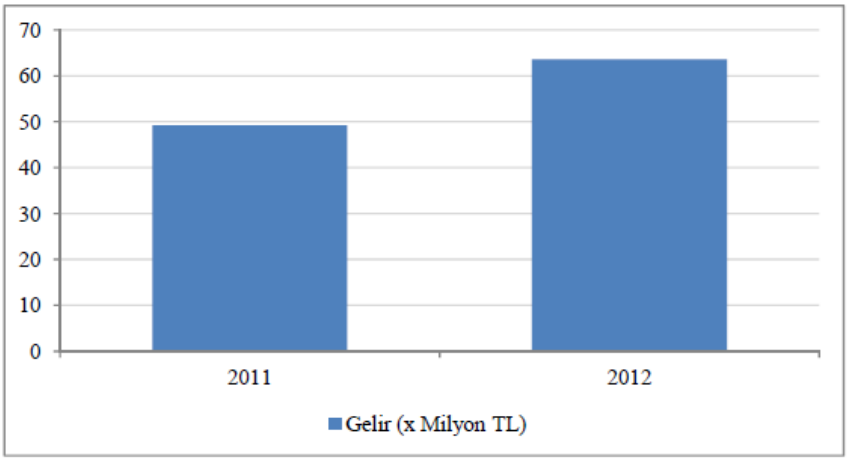
İnsandan insana (P2P)	İnsandan sunucu/makinaya (P2S / P2M)	Makinadan makinaya (M2M)
Telefon (ses, görütü)	Elektronik / uzaktan eğitim	Telemetri (telemetri)
Mesajlaşma	Elektronik ticaret / bankacılık	Telematik (telematic)
Dosya paylaşımı	Teletıp	Telesağlık (e-health)
Görüntü, müzik paylaşımı	Elektronik kamu hizmetleri	Akıllı yollar/araçlar
Karşılıklı oyun	Elektronik iş sistemleri	Akıllı evler, binalar
	Arama motorları	İş gören makinalar/robotlar

Deloitte Türkiye tarafından yapılan araştırmada Telekom hizmetleri içinde en büyük payı %47 ile geleneksel ses tabanlı telefon hizmetleri oluştururken 2010 yılında bu oran eşitlenerek yaklaşık %50'sini ses dışı hizmetler yani işletim sistemleri, haberleşme ekipmanları, haberleşme yazılımları ve bunlara benzer her türlü olgu artık Telekom hizmetleri içerisindeki yerini sürekli arttırarak büyümektedir [21].

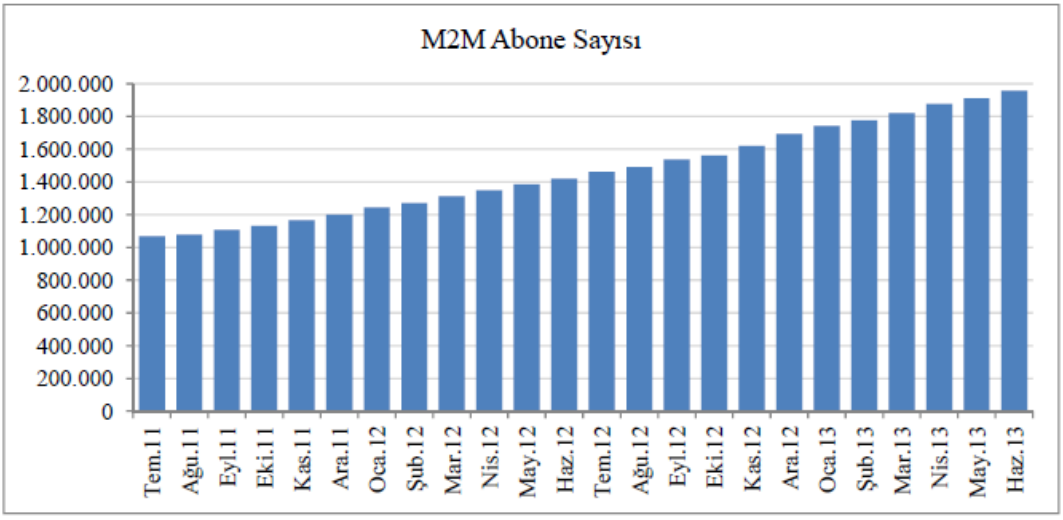
Elektronik haberleşmesinin yaygınlaşması ve dünya genelindeki IP tabanlı şebekelerin artık canlı olmayan nesneler üzerine haberleşme tekniklerini geliştirmesi ile M2M uygulamalarının önemi artmıştır. İnsan hayatını kolaylaştıran projelerin başında gelen M2M uygulamaları için hedef 2020 olarak ön görülse de günümüzde her şeyin başına bir akıllı kelimesi eklenmesi ile M2M uygulamaları için zemin hazırlanmaktadır. Değerlere ve tablolara bakıldığında M2M'in önü kesilemez ilerleyişi dünya genelinde hızını sürekli arttırarak devam edecektir. Bu hızı düşüren en önemli unsur ise insan hayatını kolaylaştıran M2M uygulamalarının yine insanların özel hayatına müdahale olarak görülmesidir.

2.3.2. Ülkemizde M2M Pazarı ve Uygulama Alanları

Ülkemizde M2M uygulamalarının en yaygın kullanım alanı araç takip sistemleri olarak karşımıza çıksa da, makineler arası iletişim çok kapsamlı bir alana hitap etmektedir. Ülkemizde M2M üzerine uygulama geliştiren firmalar veya kuruluşların başında GSM operatörleri gelmektedir. Bunun nedeni ise artık Telekom alt yapılarının geleneksel ses iletişimi payının azalması makine veya nesnelerin iletişimin giderek artmasıdır. Ağ şebeke yapılarını bu güdü ile geliştiren GSM operatörleri hazırladıkları çeşitli M2M uygulamaları ile kullanıcının karşısına çıkmaktadırlar. M2M üzerine çalışmalarını yoğunlaştıran GSM operatörleri çalışmalarının meyvelerini almışlardır. Şekil 2.5.'de ülkemizde M2M üzerine çalışma yapan GSM operatörlerinin gelirlerine yer verilmiştir.



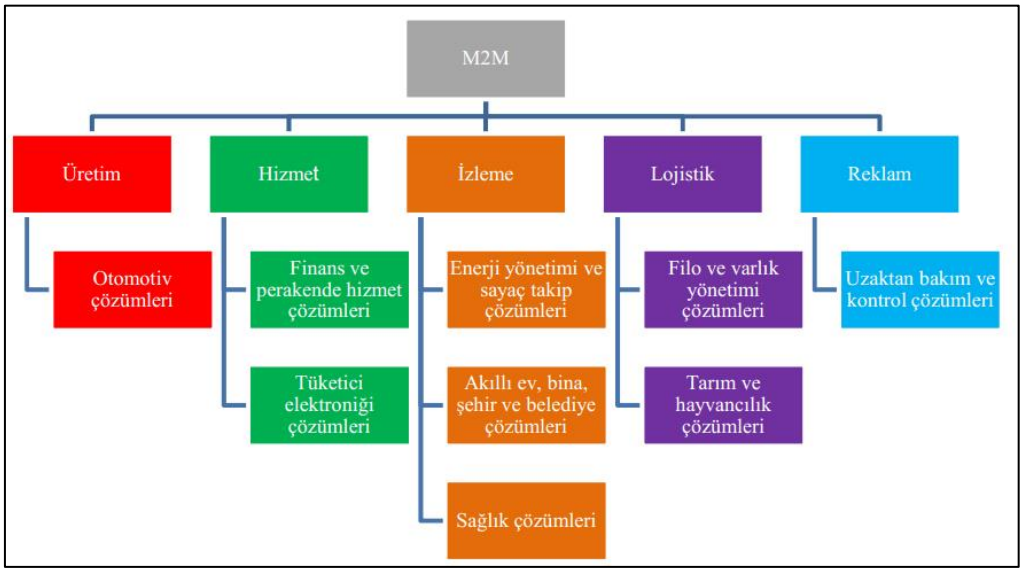
Şekil 2. 5. Türkiye'de mobil işletmecilerin M2M hizmet gelirleri (x Milyon TL) [12]



Şekil 2. 6. M2M hizmet aboneleri sayısı (x Milyon TL) [12]

2011 yılı Temmuz ayı itibariyle, mobil işletmecilerin M2M abonelerinin toplam sayısı şekil 2.6.'da gösterilmiştir. Ayrıca şekil 2.6.'da görüldüğü gibi M2M hizmet aboneleri sayısı giderek artmaktadır. Bu artışa bağlı olarak M2M alanına yatırım yapan mobil işletmecilerin M2M hizmet gelirleri de şekil 2.5.'te görüldüğü gibi giderek artmaktadır. Görüldüğü gibi hem yurtiçi hem yurtdışındaki M2M pazarının büyüklüğü ve kazanç getirisinin fazla olması M2M üzerine birçok proje geliştirilmesini tetiklemiştir.

Ülkemizde herhangi bir şekilde haberleşme alt yapısını kullanan son kullanıcıların yanı sıra kamu, kurum ve kuruluşların istekleri doğrultusunda farklı alanlarda M2M projeleri geliştirilmiştir. Bir "Soğuk Zincir Takip ve Stok Yönetim Sistemi" projesi ile aşı ve anti serumların Türkiye Halk Sağlığı Kurumu'na bağlı 10.000 bölgesel depoya nakli sağlanmıştır [22]. Bu Aşı Takip Sisteminde aşıların bulunduğu ortamların sıcaklık ölçümleri belirli aralıklarla yapılarak hem yetkililere sunulmakta hem de veriler düzenli olarak kaydedilmektedir. Bu ve buna benzer birçok proje geliştirilmiş ve uygulamaya alınmıştır. Özellikle uzaktan araç takibi, sağlık bilgileri kontrolü ve mobil ödeme cihazlarında kullanılan M2M hizmetleri gelişen teknoloji ve alt yapı sayesinde her alanda kullanılmaya başlamıştır. Ülkemizde dünya genelindeki gelişmelere paralel olarak farklı alanlarda uygulamalar geliştirilmektedir. Altyapıdaki kablolu ve kablosuz esnekliğe göre uygulamalar geliştirilmiştir. Esnek ağ sunma potansiyeline sahip mobil işletmeciler ülkemizde M2M çözüm üreticilerinin başını çekmektedir. Ülkemizde sıklıkla görülen ve destek verilen M2M uygulama alanları şekil 2.7.'te gösterilmektedir.



Şekil 2. 7. Türkiye'de M2M uygulama alanları [12]

Deloitte “2015 Teknoloji Medya Telekomünikasyon Öngörüleri” raporunda M2M ve IoT ile ilgili teknoloji alanları üzerinde durmakta, 2015 yılı itibariyle IoT akımı kapsamında birbiriyle haberleşebilen cihazların değerinin 10 milyar dolar ve cihazlarla gerçekleştirilen işlem değerinin 70 milyar dolar civarında olacağını tahmin etmektedir [24]. Rakamlarla ifade edildiği gibi M2M ve IoT pazarı son günlerde giderek artmakta ve oldukça büyük değerlere ulaşacağı tahmin edilmektedir. Ülkemizin de bu büyümeye ayak uydurması ve bu pastadan pay alması adına geliştirdiğimiz sistem sayesinde makine haberleşmesinde kullanılabilecek bir platform tasarımı yapılmış ve prototipi sunulmuştur.

3. PLATFORM GELİŞTİRİRKEN KULLANILAN TEKNOLOJİLER

Bu bölümde, tez çalışması içerisinde geliştirilen M2M platformunda kullanılan mimari, veri tabanı, haberleşme yapısı, yazılım teknolojileri, M2M cihazı ve ara yüz teknolojileri detaylı bir şekilde açıklanmış ve neden bu teknolojilerin kullanıldığı belirtilmiştir.

3.1. XML ve JSON Kullanımı

Bilgi kaynakları ve bilgiye dönüştürülmemiş veriler her sisteme ve her kullanıcıya standart olarak sunulmak istenmektedir. Bu nedenle dünya genelinde bu amaç doğrultusunda tasarlanmış olan XML veri yapısı bu işi yapmaktadır. Fakat yapısal, yarı yapısal ve yapısal olmayan verilerin çoğalması ve büyük verilerin devreye girmesi XML'in yerini JSON'a bırakmasına neden olmuştur. Popülaritesi gittikçe artan JSON özellikle kendi için özel tasarlanan web servislerle sorunsuz çalışmaktadır.

M2M platforma farklı kullanıcılar aynı anda birden farklı istekte veya işlemde bulunabilirler, bu gibi durumlarda verilerin daha düşük boyutlu ve daha hızlı bir şekilde sunulması gerekmektedir. Çizelge 3.1.'de görüldüğü gibi örnek XML yapısında veri tanımlamalarda başlık etiketleri kullanılırken JSON yapısında bu işlem noktalama işaretleri ile basitçe yapılabilmektedir. Etiketleri oluştururken veya farklı işlemler yaparken XML dönüştürücüler kullanmak gerekir. Fakat JSON yapısı, JSON yapısına uygun veri tabanı ve uygun web servislerin seçilmesi ile bu dönüşüm gereksinimi ortadan kaldırılarak sistemlerde karmaşıklığı azaltıp, hızı arttırmak mümkündür.

Çizelge 3.1. XML ve JSON veri formatı

Örnek XML Yapısı
<?xml version="1.0" encoding="UTF-8"?> <cihaz> <_class>com.saadinoyucu.model.Cihaz</_class> <cihazID>2e6e0c8b-12c7-4353-be5c-7a2486c52db8</cihazID> <cihaz_adi>Okul DHT11</cihaz_adi> <kullaniciID>55227aed88e5f382b2006c59</kullaniciID> <nem>35.00</nem> <sicaklik>21.00</sicaklik> </cihaz>
Örnek JSON Yapısı

{
"_class" : "com.saadinoyucu.model.Cihaz",
"kullaniciID" : "55227aed88e5f382b2006c59",
"cihazID" : "2e6e0c8b-12c7-4353-be5c-7a2486c52db8",
"cihaz_adi" : "Okul DHT11",
"sicaklik" : "21.00",
"nem" : "35.00"
}

JSON yapısı sadece karmaşıklığı gidermekle kalmayıp veri boyutunu da düşürmektedir. Böylelikle şebekeden taşınacak olan veriler XML'e göre daha küçük boyutlarda JSON veri formatı yapısında taşınmaktadır.

Çizelge 3.2. XML ve JSON veri formatlarının boyut karşılaştırması

Adet	XML Dosya Boyutu	JSON Dosya Boyutu
1	314 Bayt	220 Bayt
10	2.780 Bayt	2.173 Bayt
50	13.748 Bayt	10.856 Bayt
100	28.458 Bayt	21.756 Bayt

Bu tez çalışmasında gerek sadelik gerekse de basit ve anlaşılabilir yapısı bakımından JSON veri formatı kullanılmıştır. Çizelge 3.2. incelendiğinde JSON veri formatının şebeke üzerinden taşınırken XML'e göre daha az boyutlu olduğu açıkça görülmektedir. Bu nedenle ağ trafiğini daha verimli kullanmak adına geliştirilen M2M platformunda JSON veri formatı tercih edilmiştir.

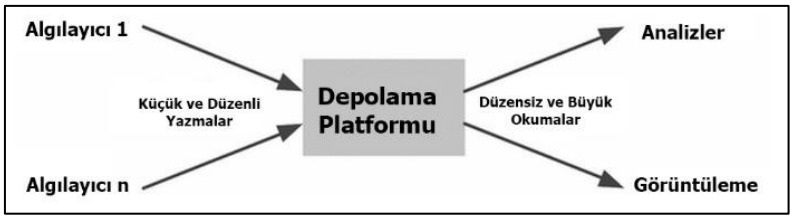
3.2. Veri Tabanının Seçilmesi

M2M sistemlerine veri sağlayan en önemli bileşenler algılayıcılardır. M2M sistemlerde algılayıcılardan anlık olarak veri alınmakta ve daha sonra bir M2M platformuna gönderilerek bu veriler izlenebilmekte, saklanabilmekte ve istatiksel analizlere tabi tutulabilmektedir. Saklanan veriler oldukça büyük boyutlara ulaşabilmektedir ve bu büyüklükteki verileri yazmakta, okumakta ve analiz etmekte bazı problemler ortaya çıkabilmektedir.

Geleneksel olarak verileri depolamak için ilişkisel veri tabanı modeli kullanılmaktadır. Bu tip veri tabanları sorgu dili olarak bilinen Yapısal Sorgu Dili (Structured Query Language-SQL) veri tabanı olarak da bilinirler [24]. Günümüzde SQL Veri Tabanlarının büyük öneme sahip olması ve büyük projelerde kullanılması arka planda desteğinin çok iyi olmasındandır. Özellikle Oracle, IBM ve Microsoft gibi dev teknoloji firmalarının desteklemesi ile bu veri tabanları piyasada oldukça fazla yer tutmuştur. Fakat günümüzde Bulut Bilişim ve dağınık web uygulamalarının yaygınlaşması, kullanılabilirliği ve ölçeklenebilirliği yüksek veri tabanlarına ihtiyacın artmasına sebep olmuştur. Böylelikle ilişkisel olmayan veri tabanı İlişkisel Olmayan Veri Tabanı (NoSQL: Not Only SQL) kavramı ortaya çıkmıştır. Özellikle artan veri depolama ihtiyacına bir çözüm ve veri tabanı performansını arttırmak isteyen Amazon ve Google gibi şirketler ilişkisel olmayan veri tabanlarını kullanmaya başlamıştır. Amazon şirketinin Dynamo teknolojisi ve Google'ın Bigtable uygulaması günümüzdeki NoSQL veri tabanları için kaynak olmuştur [25].

3.2.1. Algılayıcı Verileri

M2M sistemde algılayıcıdan gelen verilerin boyutu ve türü kullanılan algılayıcıya göre değişmektedir. Örneğin basınç ölçümü yapan bir algılayıcıdan gelen veri ile sıcaklık ölçümü yapan bir algılayıcıdan gelen veri birbirinden farklıdır. M2M uygulama hangi özel alanda gerçekleştiriliyor ise o alana uygun algılayıcılar kullanmak gerekir. Veriler uygun olarak alınır ve veri tabanlarına kaydedilir.



Şekil 3.1. Veri tabanında algılayıcı verilerini yazma ve okuma işlemi [26]

M2M sistemlerinde veriler düzenli olarak veri tabanına kaydedilirken yazma işleminde farklı veri tabanlarında farklı dalgalanmalar gözlenmektedir. Fakat okuma işlemi sırasında birden fazla kullanıcı birden farklı platformda verileri analiz etmek için veri tabanından istekte bulunabilir. Bu işlem sırasında şekil 3.1.'de görüldüğü gibi veri tabanına aşırı yüklenme olabilmektedir. Bundan dolayı çoklu yazma ve çoklu okuma işlemlerinde üst düzey bir performans gösteren veri tabanının seçilmesi gerekmektedir.

3.2.2. SQL Veri Tabanları

SQL veri tabanlarında veriler önceden tasarlanmış şemalar içerisinde tutulur. Şemayı oluşturan unsurlar satır ve sütunlardır. SQL veri tabanlarının temel özellikleri aşağıda sıralanmıştır.

- İlişkisel veri tabanı, önceden tanımlanmış ve kategorize edilmiş tablolar içerisine veri yerleştirme biçimidir.
- Her tablo sütunları bir veya daha fazla veri kategorisi içerir.
- Her satır, sütunlara göre belirlenen kategoriler içinde eşsiz bir veri örneğini içerir.
- Kullanıcı veri tabanı tablosunun yapısını bilmeden veri tabanındaki veriye erişebilirsiniz.

SQL veri tabanı bazı kısıtlamalara sahiptir. Bunlardan ölçeklenebilirlik ve karmaşıklık aşağıda açıklanmaya çalışılmıştır.

Ölçeklenebilirlik: İlişkisel veri tabanında ölçeklenebilirlik çok güçlü ve pahalı sunucular ile gerçekleştirilebilir. Tek bir yerden depo edilmeli prensibine dayanan bu veri tabanı birden fazla yerdeki kaynakların birleştirilmesi oldukça zordur.

Karmaşıklık: SQL sunucu verilerini tablolar içerisinde saklar bu durum farklı boyutta verilerin girilmesinde zorluklar ortaya çıkarır.

3.2.3. NoSQL

Son günlerde sıklıkla duyulan NoSQL veri tabanı kavramı, yıllardır bilişim dünyasında kullanılan ilişkisel veri tabanı sistemlerine alternatif olarak çıkmıştır. İnternet ortamında gün geçtikçe büyüyen verileri depolayabilmek ve yüksek erişilebilirliğin yanında yatay ölçeklenebilen sistemlere verilen genel bir isim olarak günümüzde anılmaktadır.

Google'ın BigTable ve Amazon'un Dynamo teknolojisinde kullandığı ilişkisel olmayan veri tabanındaki başarısı ve aynı zamanda NoSQL veri tabanlarının ölçeklenebilirliği, hızı, erişim kolaylığı, maliyeti vb. gibi kavramlarda sağladığı üstünlük NoSQL veri tabanlarının popülaritesini arttırmıştır [24].

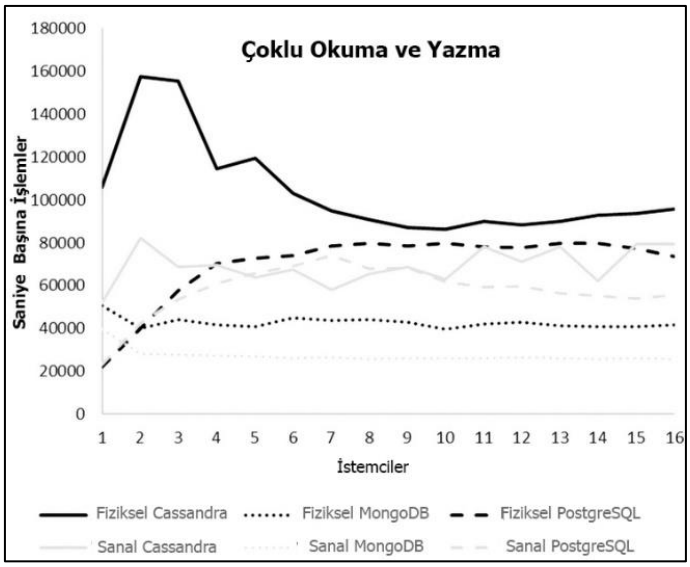
Günümüzde farklı NoSQL veri tabanları bulunmaktadır. Farklı NoSQL veri tabanlarının bazılarında ise raporlama ve SQL standartlarının desteklenmemesi gibi kısıtlar vardır. NoSQL veri tabanlarının avantajları ise şunlardır; veri okuma ve yazma hızı, toplu veri işlemlerini desteklemesi, genişletilmesinin kolay olması ve düşük maliyetinin olmasıdır [27].

NoSQL veri tabanlarının belge odaklı olması ile farklı teknolojiler bir arada kullanılabilmektedir. Bir NoSQL veri tabanı XML veya JSON teknolojisini destekleyebilir. Veriler belirtilen bu teknoloji serileri halinde bulunmaktadır. NoSQL içerisinde kolonlardan bağımsız saklanan her bir veri parçası basit anahtar değerleri ile veri tabanı sistemine gönderilir. Benzersiz bir ağ yapısı ile yerleştirilen verileri okurken yüksek yoğunluk altında bile üst düzey performans alınarak düşük gecikme ile işlem yapılabilmektedir. Veriler NoSQL veri tabanlarında tutulurken nesne tabanlı mimariye benzer şekilde tutulurlar. Bu benzerlik yapısal olmayan ve farklılık gösteren verilerin saklanmasını kolaylaştırmaktadır.

Son yıllarda karşımıza çıkan NoSQL kavramı bir sistemde farklı yapıdaki verileri tutmaya olanak sağlamaktadır. Örneğin bir sıcaklık algılayıcısından gelen veri tipleri ile basınç algılayıcısından gelen veri tipi birbirinden farklı olabilir. SQL veri tabanında bu farklılık

için ayrı ayrı tablo tasarımı yapmak gerekirken belge tabanlı bir NoSQL veri tabanı tablolardan bağımsız olduğu için veriler farklı tip ve boyutlarda tutulabilir.

M2M sistemlerde veriler bir veya birden fazla algılayıcıdan alınır. Bu veriler veri tabanına yazım sırasında belirli aralıklarla veri tabanı yoğunluk yaşar fakat veriler veri tabanından okuma esnasında birden fazla kullanıcı veri tabanından farklı isteklerde bulunabilir. Bu durumda veri tabanının cevap verme süresi geç olursa, sistemde kilitlenmelere neden olmaktadır. Bu durumu engelleyebilmek için performans bakımından üst düzey bir belge tabanlı NoSQL veri tabanı tercih nedeni olmalıdır. Bunun için ise MongoDB kendini kanıtlamış bir sistemdir ve Türkiye Halk Sağlığı Kurumu "Soğuk Zincir Takip ve Stok Yönetim Sistemi" projesinde de kullanılmaktadır [22]. Sanal ve fiziksel veri tabanları üzerine yapılan bir çalışmada birden fazla okuma ve yazma işlemi için sistemin performansı şekil 3.2.'de gösterilmiştir.



Şekil 3. 2. Veri tabanı karşılaştırılması [26]

Şekil 3.2.'de görüldüğü çalışma fiziksel ve sanal olarak en çok kullanılan ücretsiz NoSQL veri tabanları olan Cassandra ve MongoDB üzerine ve İlişkisel veri tabanı olan PostgreSQL üzerine bir çalışma yapılmıştır. Bu çalışmada veri tabanları aynı anda birden fazla okuma ve yazma işlemine tabi tutulmuştur. Performansların değerlendirildiği bu çalışmada NoSQL veri tabanlarının çoklu işlemlerde başarılı olduğu tespit edilmiştir. Bu çalışma M2M sistemlerindeki çoklu işlemler için veri tabanı seçimine ışık tutmaktadır.

NoSQL veri tabanı yazılımı olarak MongoDB gibi birden fazla ücretsiz yazılımda mevcuttur. MongoDB belge veri modeli, zengin sorgu desteği, yatay ölçeklenebilirlik, yüksek kullanılabilirlik, esneklik ve dinamik şema gibi birçok özelliği geliştiricilere sunmaktadır. Veri saklama işleminde JSON yapısını kullanmaya izin veren NoSQL veri tabanı ile RestFul web servislerinin kullanıldığı sistemler, daha dinamik bir yapı ve platformdan bağımsız uygulamalar geliştirilmesine olanak sağlamaktadır. Bu nedenlerden dolayı bu tez çalışmasında veri tabanı olarak NoSQL veri tabanı olan MongoDB kullanmak uygulamalarda başarılı sonuçlar almayı kolaylaştırmış ve veri dönüşümü yapmadan sistemin işlemesini sağlamış hızı arttırmıştır.

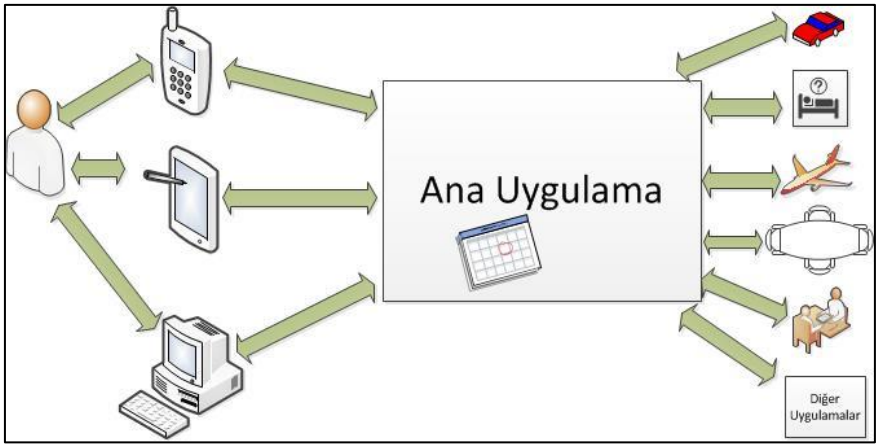
3.3. Web Servisler

Daha önceleri yerel alan ağlarında dosya paylaşımları ile gerçekleştirilen kaynak paylaşımı uygulamaları, internet hızının artması ve şebeke yapılarının güçlenmesi ile yerini farklı teknolojilere bırakmıştır. Verilere ve uygulamalara her yerden erişebilme isteği web servis teknolojilerinin çıkış noktasıdır. Web sayfalarının çoğalması, iş süreçlerinin değişmesi ve erişebilirliğin arttırılması isteği birçok uygulamanın ve farklı mimarilerin geliştirilmesine neden olmuştur. Ortak uygulama üzerinde işbirliğinin arttırılması ve farklı kişi, cihaz ve uygulamaların aynı sistemlere erişiminin sağlanması ise web servis kavramının gelişmesinde rol oynayan diğer bir unsurudur.

Merkezi bir noktada bulunan ve dağıtılmış kullanıcılara yetki seviyelerine göre yâda isteklere göre yanıt verebilen dağıtık uygulamalar ilk olarak Uzaktan Yordam Çağrısı (RPC: Remote Procedure Call) ile başlamıştır. Daha sonra ise Dağıtık İşlem Ortamı (DCE: Distributed Computing Environment) ile RPC'ler standart haline getirilmiştir. DCE'den sonra Ortak Nesne İstem Aracı Mimarisi (CORBA: Common Object Request Broker Architecture) geliştirilmiş ve dağıtık süreç teknolojisini standartlaştırmıştır. CORBA'dan sonra ise Microsoft firmasının Dağıtık Bileşen Nesne Modeli (DCOM: Distributed Component Object Model) uzak nesne protokolü ortaya çıkmıştır. Bu sıralarda en başarılı dağıtık mimariler web ve e-posta olarak gösterilmektedir. CORBA ve DCOM'dan sonra ise Java, Uzak Yordam Çağrımı (RMI: Remote Method Invocation) protokolü ile dağıtık sistemlerde büyük bir başarı yakalamıştır [28].

IBM, Oracle, Microsoft, HP ve daha birçok firma web tabanlı uygulama geliştirmeler adına yoğun çalışmalar yapmışlardır. Web servisleri, yazılım ve uygulama geliştirme araçlarını, web tabanlı yazılım geliştiren firmalara ve programcılara sunmaktadırlar. Web ortamındaki gelişmelere bakıldığında ilk olarak belgelerden oluşan web kavramı ortaya çıkmış ve Hiper-Metin Transfer Protokolü (HTTP: Hyper-Text Transfer Protocol) üzerinden statik belgeler paylaşılmaya çalışılmıştır. Böylelikle herkes basit bir Hiper-Metin İşaretleme Dili (HTML: Hypertext Markup Language) ile kendi web belgelerini oluşturmuş ve kolayca yayınlamıştır. HTML'in statik bir yapıda olması ve artık web kullanıcılarının da işin içine katılması isteği dinamik web sayfa ve belgelerinin oluşturulmasına neden olmuştur. Bu noktada Microsoft'un Etkin Sunucu Sayfaları (ASP: Active Server Pages), Java'nın Java Sunucu Sayfaları (JSP: Java Server Pages) ve Üstün yazı Ön işlemcisi (PHP: Hypertext Preprocessor) kodları ile oluşturulan web sayfaları HTML ile oluşturulan statik web sayfalarının yerini almıştır [29].

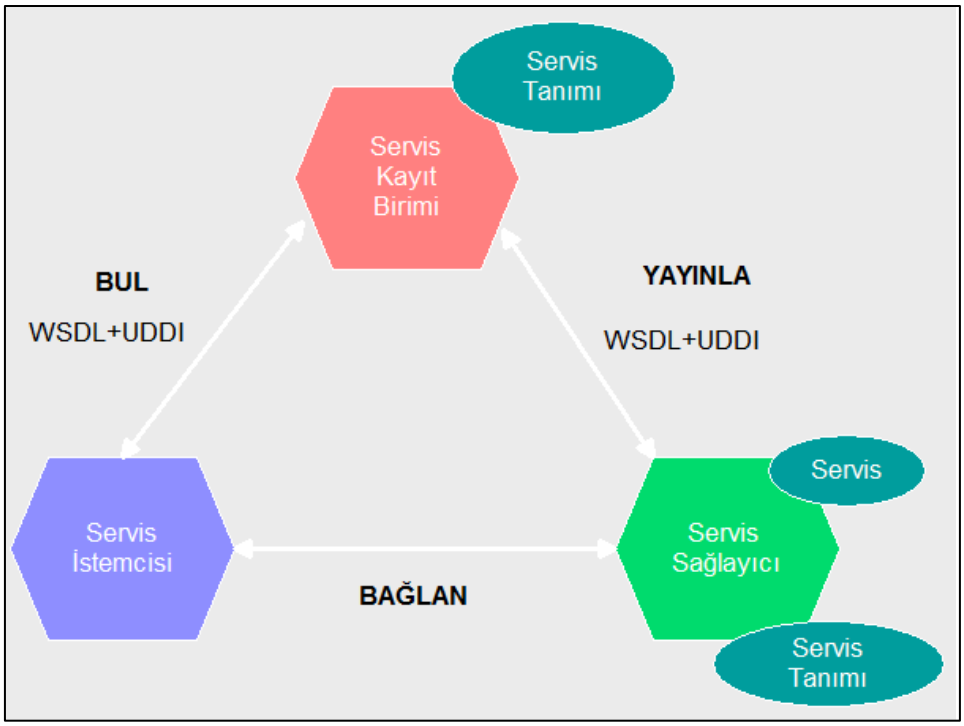
Web servisler ile geliştirilen uygulamalar farklı uygulamaları birbiriyle haberleştirebilmekte ve dağıtık mimari sayesinde birden fazla uygulamayı tek çatı altında toplayarak kullanıcıya sunabilmektedir. Şekil 3.3.'te gösterilen bir yönetici asistan uygulamasında gerekli veriler web servisler ile kolaylıkla alınabilir, dağıtılabilir ve kullanıcıya sunulabilir yapıda olduğu açıkça görülmektedir. Bu sistemde tek uygulama üzerinden asistan veya yönetici istediği tüm hizmetlere aynı ortamdan ulaşıp, plan ve programının yanında gerekli rezerve işlemlerini de yine geliştirilen sistem üzerinden gerçekleştirebilmektedir.



Şekil 3. 3. Yönetici asistan uygulaması

SOA çözüm geliştirmek için temel unsur olarak servisleri kullanan bir yazılım mimarisidir. Servisler ise kısaca farklı uygulama ve cihazlara istek verebilen yazılım parçaları olarak adlandırılabilir. SOA yaklaşımı yeni teknolojilerin geliştirilmesine öncülük etmiş dağıtık sistemlerin ve akıllı cihazlar arasındaki bağlantıların yapılmasını kolaylaştırmıştır. İnternet tabanlı uygulamaların yaygınlaşması SOA yaklaşımının sürekli gelişmesini sağlamıştır.

Web servisleri SOA yaklaşımının mevcut uygulamalarından biridir ve bu teknolojilerin kullanımının sağlanması için üç unsurun birbiriyle bağlantı kurması gerekmektedir. Bu unsurlar servis sağlayıcı, servis istemcisi ve servis kayıt birimidir. Bu yapı üzerine birçok standart geliştirilmiştir fakat temel olarak Basit Nesne Erişim Protokolü (SOAP: Simple Object Access Protocol), Web Servisleri Açıklama Dili (WSDL: Web Services Description Language) ve Evrensel Tanımlama, Bulma ve Entegrasyonu (UDDI: Universal Description, Discovery and Integration) standartları tercih edilmektedir. Web servisler HTTP ve XML gibi birlikte çalışabilen kavramların kullanılmasıyla daha da güçlü hale gelmiştir. Servis sağlayıcı istemcilerin sunucuda bulunan servislerle haberleşmesini sağlamaktadır. Servis sağlayıcı ise kendisine gelen istekleri nasıl cevaplaması gerektiğini kayıt birimine kaydederek servislerin nasıl çağırılması gerektiğini belirtir. Servisleri sağlayıcıdan çağıran ve kullanan uygulamalara ise servis istemcisi adı verilir [30].



Şekil 3. 4. Web servis çağırma aşaması [31]

Şekil 3.4.'te web servis mimarisi görülmektedir. Mimariye göre servis istemcisi servisleri çağırır, servis sağlayıcısı ise istemcinin isteklerini cevaplamaktadır. Servis kayıtçısında servis sağlayıcı tarafından yayınlanan servis tanımları ilan edilmekte ve yayınlanmaktadır. Servis sağlayıcı servis kayıtçısında servisleri tarayarak bu servisler hakkında bilgiler elde etmekte ve ilgili görevleri yerine getirmektedir [32].

Son yıllarda popülerleşen ve kullanım alanları gittikçe artan diğer bir servis mimarisi ise RestFul yaklaşımıdır. Roy Fielding'in 2000 yılında yayınlanan doktora tezi [33] sonucu ortaya çıkan bu kavram, tek başına yeterli olmayıp bir dizi kıstaslar ile sistem tasarımlarında kullanılan mimari tarzıdır. Fielding REST için 6 adet kısıtlama belirlemiştir [33]. Bu kısıtlar aşağıdaki gibidir;

- Client-Server, (Sunucu-istemci) istemci ve sunucunun yapısının birbirinden ayrı olması anlamına gelmektedir. İstemcinin veri kaynağı hakkında hiç bir bilgi bilmemesi ve sunucunun doğru istekler geldiği sürece yanıt vermesidir. Amaç, platformdan bağımsız çalışmayı ve kapsamı arttırmaktır.

- Stateless, (Durum bilgisi) sunucuda istemci ile ilgili bir bilgi ya da oturum bilgileri tutulmaz. İstemci tarafından yapılan her istek sunucu için gerekli bilgiyi taşır. Böylece her türlü durum istemci tarafında saklanmış olur. Sunucu hiçbir durumu saklamak zorunda kalmaz.

- Cache, (Önbellekleme) önbellek ağ performansını artırır. Veri, sunucu tarafından "cacheable" olarak işaretlenirse istemci bu veriyi bellekleyebilir. Bu durum ağ performansını artırır ve kullanıcıya daha hızlı cevap verilmesini sağlar.

- Uniform Interface, (Tek biçimlilik) sunucu ve istemci için ortak olan bu kısıtlama diğer 5 kısıtlamaya göre daha önemli sayılmaktadır. Tek biçimli ara yüz, iletişim yöntemini basitleştirmektedir. Ayrıca ortak bir ara yüz olması sayesinde her parçanın birbirinden bağımsız bir şekilde evrimleşmesine olanak sağlamaktadır.

- Layered System (Katmanlı Sistem), istemci her durumda direkt olarak son sunucuya bağlanmayabilir, arada başka sunucular bulunabilir. Sistemde her katman tek bir katmanı bilmektedir. Arada bulunan sunucular yük dağılımı yaparak kapsamı

arttırabilmektedir ve istemcileri belirli güvenlik kurallarına zorlayabilmektedir. Bu durumun çok kullanımı istemci ve sunucu arasında gecikmelere yol açmaktadır.

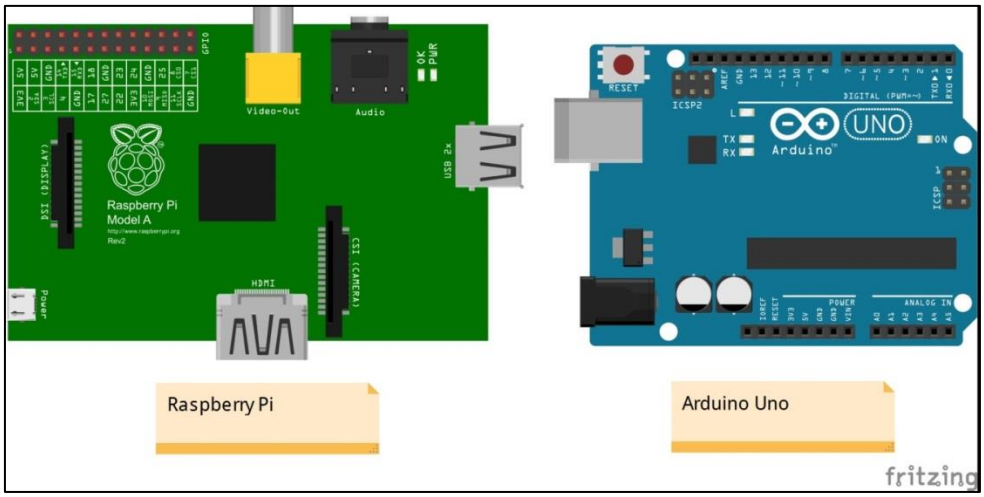
- Code-On-Demand, bu kısıt isteğe bırakılmıştır. Sunucu belli durumlarda istemci tarafına çalıştırılabilir kod parçacıkları ve uygulamalar gönderebilir.

REST mimarisi temelde SOAP'a benzemektedir fakat istemci-sunucu arasında basit istek-yanıt prensibine dayandığından SOAP'a göre kolay anlaşılabilir bir yapı içerisindedir. Bu basit istek-yanıt yapısı akıllı cihazlar için geliştirilen Uygulama Programlama Arayüzü (API: Application Programming Interface)'lerde işleri oldukça kolaylaştırmaktadır. RestFul yapısı herhangi bir teknolojiye bağlı değildir. XML veya JSON mesajlaşma yapısını destekleyebilmektedir. RestFul mimarisinde HTTP'nin dört temel operasyonu kullanılır. Bunlar GET/POST/PUT ve DELETE'dir. GET fiili kaynak listesi keşfi yâda kaynağın anlık durumunu hiçbir yan etki yâda değişiklik yaratmadan alabilmeyi sağlarken, PUT ve POST fiilleri kaynak üzerinden bir değişiklik yapmak için kullanılmaktadır [34].

Bu tez çalışmasında mimari olarak SOA ve web servis yaklaşımı olarak RestFul web servisleri seçilmiştir. Bunun nedeni ise RestFul yapısının desteklediği JSON yapısının XML'e göre avantajlı olması, JSON kullanımının basit ve boyutunun küçük olması, Roy Fielding'in 6 kıstasından biri olan katmanlı mimari, önbellekleme, sunucu-istemci yapısı ve durum bilgisinde oturum bilgilerinin tutulmamasıdır. M2M uygulamalarında bu kıstaslar önemli derecede performans ve kullanım kolaylığı sağlamaktadır. Ayrıca temelde JSON mesajlaşma yapısının seçilmesi, JSON destekleyen web servislerin kullanılması ve kayıtların JSON olarak saklanabileceği NoSQL veri tabanı sistemlerinin kullanılması tez çalışmasında veri dönüşümsüz olarak işlem yapıldığı ve kaydedildiği için önemli derecede hız sağlamaktadır.

3.4. M2M Cihaz ve Algılayıcıları

Bu tez çalışmasında M2M cihaz ve algılayıcıları bakımından temelde iki farklı ürün üzerinde durulmuştur. Bunlardan ilki Arduino akıllı kartları ikincisi ise Raspberry Pi akıllı kartıdır. Her iki kart, üzerinde barındırdığı işlemci, denetleyici, kullanıcıya sunulan pin sayısı ve farklı özellikleri ile birbirinden ayrı üstünlükler sağlamaktadır. Resim 3.1.'de Raspberry Pi ve Arduino Uno kartları gösterilmektedir.



Resim 3. 1. Raspberry Pi Model A ve Arduino Uno üst görünümü

Raspbery Pi ARM tabanlı bir mikroişlemciyi kartlarında sunmaktadır. Saat hızı olarak ise 1 Ghz'e kadar destek veren Raspberry Pi 512 MB RAM kapasitesine sahiptir. Üzerinde barındırdığı ses, grafik, Evrensel Seri Veri Yolu (USB: Universal Serial Bus), Ethernet ve hafıza kartı portu sayesinde küçük bir bilgisayarın yaptığı işleri kolaylıkla yapabilmektedir. Raspberry Pi kullanabilmek için Linux bilgisine sahip olmak gerekmektedir. Bu ve vb. özelliklerinden dolayı daha çok işlemci tabanlı ve işletim sisteminin özelliklerinden yararlanmak istenilen projelerde Raspberry Pi kullanılmaktadır. Bu tez çalışmasında gerek kullanım kolaylığı gerekse de maliyet açısından üstünlük sağlayan Arduino kart kullanılmıştır.

Arduino kart, 8 bit işlem yapabilen mikro denetleyiciye sahip piyasada farklı modelleri olan bir akıllı kart sistemidir. Arduino 8-16 MHz hızına ve 2-8 KB RAM kapasitesine sahiptir. Raspberry Pi'nin aksine üzerinde USB portundan başka etkileşim yapılabilecek Ethernet veya grafik portu barındırmamaktadır. Arduino farklı işletim sistemlerinde çalışabilen bir kod yazma uygulaması ile kolayca programlanabilmektedir. İnternet ortamında Arduino ile geliştirilmiş birçok proje yer almaktadır. Bu durum Arduino ile işlem yapanların işlerini kolaylaştırarak örnekler sunmaktadır. Arduino maliyet bakımından da Rassberry Pi'ye göre daha ucuzdur. Arduino ile her projeye uygun kalkan (Shield) adı verilen eklenti katmanları bulunabilir. Örneğin kablosuz haberleşme için Wi-Fi eklentisi, kablolu bağlantı için Ethernet eklentisi kolaylıkla kullanılabilir ve fonksiyonelliği arttırılabilir. Ayrıca piyasada Arduino üzerinde kullanılacak algılayıcılar

için hazır yapılar bulunmakta ve bu yapılar proje geliştirenlerin işlerini kolaylaştırmaktadır.

Algılayıcı teknolojisi gün geçtikçe gelişmekte ve boyut olarak da küçülmektedir. Arduino ve Raspberry Pi üzerinde kullanılabilecek sıcaklıktan spesifik gaz algılayıcılarına kadar birçok algılayıcı bulunaktadır. Fakat bu tez çerçevesinde örnek teşkil etmek açısından LM35 sıcaklık ve DHT11 sıcaklık-nem algılayıcıları kullanılmıştır. Arduino modellerinden Arduino Uno modeli 14 sayısal ve 6 analog pin sunup 32 KB hızlı bir belleğe sahip olması açısından tez çalışmasında kullanılmıştır. Platformla bağlantıyı sağlamak adına ise Arduino Ethernet kalkanı kullanılmıştır.

3.5. Ara yüz Standart ve Teknolojileri

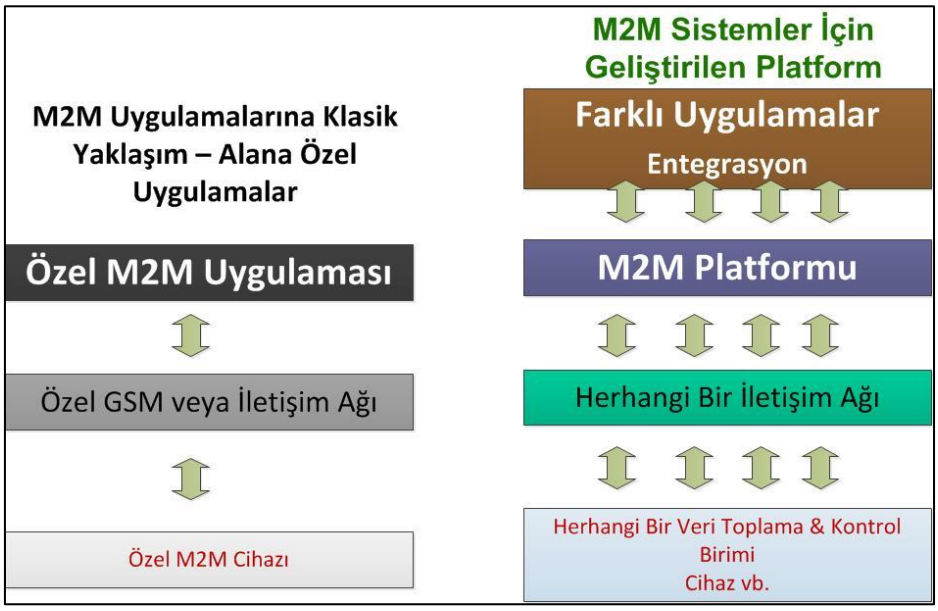
Kullanıcıyla etkileşim içerinde olan bütün projelerde verilerin nasıl gösterileceği, kullanıcıdan verilerin nasıl ve hangi şekillerle alınacağı ve kullanım kolaylığının nasıl sağlanacağı oldukça önemlidir. Bu nedenle bu tez çalışmasında son uygulama geliştirme teknikleri, yöntemleri ve tasarım standartları kullanılmıştır. Özellikle HTML sürüm 5'in Basamaklı Stil Şablonları (CSS: Cascading Style Sheets) sürüm 3 ve JQUERY kütüphaneleriyle yakaladığı mükemmel uyum prototipi geliştirilen M2M platformunda kullanılmıştır. Ayrıca platform ara yüzleri duyarlı bir tasarımla kullanıcıya sunulmuştur. Böylelikle mobil sistemler, akıllı telefonlar, tabletler, dizüstü bilgisayarlar ve masaüstü bilgisayarlara otomatik olarak uyum sağlayan bir ara yüz geliştirilmiştir. Bu konuya platform prototipi geliştirilirken detaylı olarak değinilecektir.

4. M2M PLATFORMUNUN GELİŞTİRİLMESİ

Bu bölümde M2M sistemler için hazırlanan platformun geliştirilme adımları detaylı bir şekilde anlatılmıştır. Mevcut uygulamalardan farkları ve üstünlükleri her adım için ele alınmış ve açıklanmaya çalışılmıştır. Bu bağlamda ilk olarak platformda kullanılan M2M mimarisi son olarak ise ara yüz çalışmaları açıklanmıştır.

4.1. M2M Platformu Mimarisi

M2M uygulamalar genel olarak alana özel gerçekleştirilen projelerle göze çarpmaktadır. Alana özel yapılan projeleri bir standarda dayandırmak veya herkesin rahatça kullanabileceği bir yapıya getirmek oldukça zordur. Bu nedenle bu tez çalışmasında uluslararası standart geliştiren ve M2M sistemlerin gelişmesine sürekli katkıda bulunan kuruluşların basit mimarisinden yola çıkılarak platform için gerekli mimari yapı tanımlanmıştır. Şekil 4.1.'de klasik M2M mimari yaklaşımı ve önerilen mimari yaklaşımı görülmektedir.



Şekil 4.1. M2M sistem mimarileri

Klasik yaklaşımda müşteri veya kullanıcı geliştiriciden uygulama isteğinde bulunur, gereksinimler çıkartılır ve bu gereksinimlere göre özel M2M cihazı, özel GSM veya iletişim ağı ve özel M2M uygulaması geliştirilerek kullanıma sunulur. Bu yaklaşım kişiden

kişiye ve kullanım yerine göre değişiklik göstermekte ve her değişen unsur için tüm mimari yeniden tasarlanmak zorundadır. Önerilen mimaride ise herhangi bir veri toplama, kontrol birimi veya cihaz kullanılabilir, iletişim için ise temelde IP tabanlı iletişim yapabilen herhangi bir ağ kullanılabilir, M2M platformu farklı isteklere, farklı cihazlara, farklı kullanıcılara, farklı cihazlarda (tablet, pc, akıllı telefon vb.) aynı anda hizmet verebilecek şekilde tasarlanmıştır. Ayrıca bir adım ilerisi düşünülerek geliştirilen bu platformun başka uygulamalar, kamu projeleri ve çeşitli spesifik alanlarda kullanılması adına çalışmalar yapılmış ve sistem tasarımı bu doğrultuda geliştirilmiştir.

Standartları bu tez çalışmasında kullanabilmek adına OneM2M'in M2M sistem mimarilerini incelediği dokümanında yer alan ve ETSI'nin de desteklediği M2M servis yetenekleri katmanının devreye alındığı yapı M2M platform mimarisinde kullanılmıştır. Bu yapıya göre katmanlı bir mimariye sahip olacak şekilde geliştirilmesi gereken platformun servis katmanı ile güçlendirilmesi gerekmektedir. Farklı kullanıcılardan ve cihazlardan gelecek istekleri karşılayan servis katmanı kullanıcı ile bir ara yüz yardımı ile etkileşime geçecek arka planda ise veri tabanı sistemi sürekli servis katmanı ile iletişim içerisinde olacaktır. Böylelikle Roy Fielding'in savunduğu ve kullanıcının verinin kaynağı hakkında bilgi sahibi olmaması kavramı gerçekleştirilmiş olunacaktır [33]. Mimari yapı belirlendikten sonra yapılacak ilk iş uygun veri tabanı sisteminin kurulması ve hazır hale getirtilmesidir.

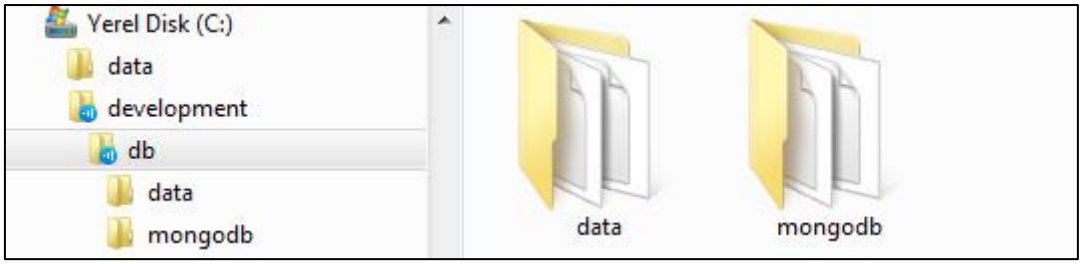
4.2.M2M Platformu Veri Tabanı ve Kurulması

M2M cihaz bilgilerini, cihazlardan alınan verileri, kullanıcıların kişisel bilgilerini ve oturum bilgilerini saklayabilmek için veri tabanı sistemi kullanmak şarttır. Fakat hangi veri tabanının kullanılması gerektiği sistem geliştiricilere bırakılmıştır. Günümüzde ilişkisel ve ilişkisel olmayan iki farklı veri tabanı yapısı mevcuttur. Bu tez çalışmasında ilişkisel olmayan belge tabanlı bir veri tabanı sistemi kullanılmıştır. Bu yapının neden seçildiği ise "Veri Tabanının Seçilmesi" başlığı altında detaylı bir şekilde anlatılmıştır. Bu bölümde ise veri tabanı sisteminin kurulması ve hazır hale getirilmesi işlemleri açıklanmaktadır.

Geliştirilen platformda kullanmak üzere NoSQL belge odaklı veri tabanı kullanılmasına karar verildikten sonra farklı veri tabanı firmalarının sunduğu yapılar incelenmiştir ve ücretsiz bir NoSQL veri tabanı olan MongoDB'nin kullanılması uygun görülmüştür.

MongoDB belge odaklı olması, üzerinde geliştirilmiş birçok uygulama olması, geliştiricilerin sürekli çevrimiçi eğitimlerle desteklenmesi adına kullanımı kolay ve performansı yüksek bir veri tabanı sistemidir. Ayrıca MongoDB belge tabanlı olarak JSON yapısını desteklemektedir. Böylelikle veri dönüşümü yapılmadan tasarlanmak istenilen platformun JSON destekleyen bir veri tabanı kullanarak sağlam bir temel üzerine inşa edilmesi sağlanmıştır.

MongoDB Linux, Windows ve OS X işletim sistemlerinde sorunsuz olarak çalışmaktadır. Tez çalışması kapsamında geliştirilen M2M platformu Windows 7 tabanlı bir işletim sisteminde gerçekleştirildiğinden kurulumlar ve çalışmalar Windows uyumlu MongoDB üzerinde yapılmış ve hazır hale getirilmiştir. Sistem bileşenlerini tek klasörde toplamak ve karmaşıklığı gidermek adına tüm çalışmalar "C" dizininde "development" adlı klasörde gerçekleştirilmiştir. Veri tabanı işlemleri ise "db" klasörü içerisinde yapılmıştır. Öncelikle web sayfasından indirilen MongoDB "db" klasörüne kurulmuş ardından gerekli ayarlamalar yapılmıştır. MongoDB belge tabanlı bir sistem olduğundan temelde bir ara yüz ile değil kodlarla çalışmakta ve başlatılmaktadır. Resim 4.1.'de gösterildiği gibi dizin yapısı oluşturulmuş ve MongoDB sistem dosyaları "mongodb" klasöründe, veri tabanı verileri ise "data" klasöründe saklanmıştır.



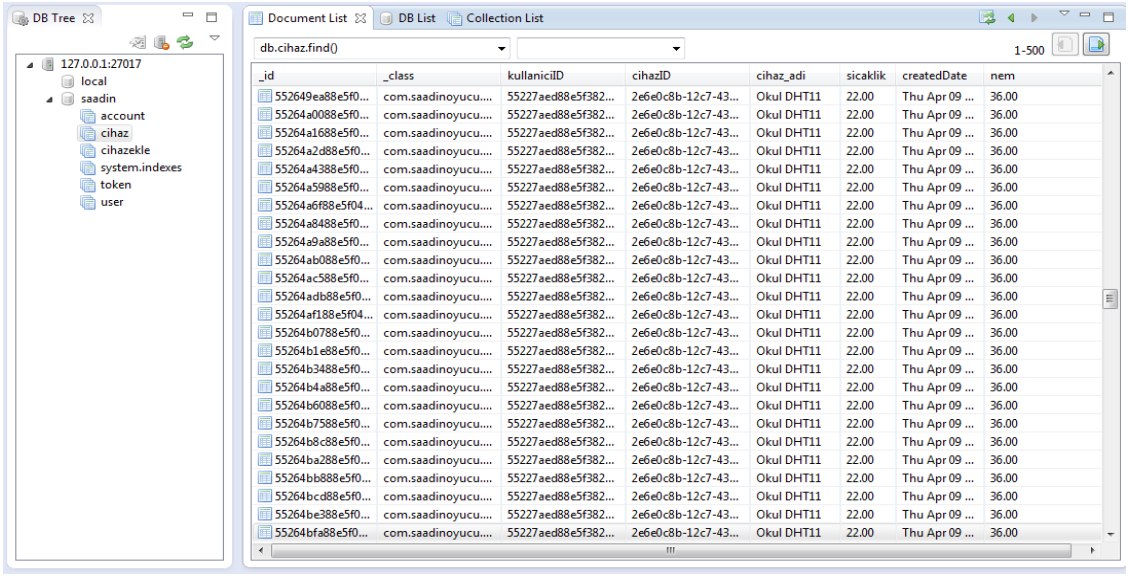
Resim 4.1. Veri tabanı dosya dizini

MongoDB çalıştırmak için yönetici yetkisine sahip Windows kullanıcısı olmak ve komut istemi ekranından kod yardımıyla veri tabanını hazır hale getirmek gerekmektedir. Bu nedenle aşağıdaki uygulama kodu işimizi görecektir.

c:\development\db\mongodb\bin\mongod --dbpath c:\development\db\data --port 27017 --rest

Kod yapısı incelendiğinde öncelikle MongoDB yapısını sistemde çalıştırmak için gerekli olan dosyaların yeri belirtilmiş ve “mongod” çalıştırılarak sistem hazır hale getirilmiştir. Fakat sisteme veri dosyalarının nereye kaydedeceğini de yine kullanıcının belirtmesi gerekmektedir. Bu nedenle yukarıdaki kod parçasının devamında “--dbpath” komutu ile hemen sonrasında belirtilen dizine veri dosyalarının kaydedileceği belirtilmektedir. MongoDB başlatıldığı andan itibaren 27017 İletim Kontrol Protokolü (TCP: Transmission Control Protocol) portunu kendisine rezerve etmektedir fakat bazı bilgisayar özellikleri ve durumlarında bu port değişebilmektedir. Sabit bir portta yayın yapması adına yukarıdaki kod parçasında “—port” komutu ile yayın yapması ve kendisine rezerve etmek istediğimiz port numarası belirtilmektedir. MongoDB web servis mimarilerine göre kullanabileceğimiz farklı teknikler sağlamaktadır. Yukarıdaki kod parçasında “--rest” komutu eklenerek basit HTTP üzerinden gelen isteklere karşı veri tabanı sisteminin hazır olması istenilmektedir.

MongoDB kendisine gelen her bir veriye benzersiz bir kimlik (ID: Identity) ekleyerek bünyesine yerleştirmektedir. Yerleşen bu verileri kullanıcı doğrudan JSON dosyası olarak görememekte arada yardımcı araçlar kullanmak zorundadır. Bunlardan bazıları RoboMongo gibi tek başına çalışan ara yüz araçları bazıları ise uygulama geliştirme yazılımlarına eklenen Eclipse MonjaDB vb. yazılımlardır. Bu yazılımlar SQL veri tabanı gibi verileri satırlar ve sütunlar şeklinde listeleyip kullanıcıya sunmaktadır. Resim 4.2.’de Eclipse MonjaDB eklentisi üzerinden bağlanılan MongoDB veri tabanı verileri listeler halinde gösterilmektedir.



Resim 4.2. Eclipse MonjaDB eklentisi ile MongoDB veri tabanının görüntülenmesi

MongoDB ilişkisel olmayan veri tabanlarının, ilişkisel olan veri tabanlarının kullanıldığı gibi kullanılması mümkün değildir. SQL veri tabanlarında veriler tablolarda tutulurken. NoSQL'de tablo kavramının yerini koleksiyon (collection) kavramı almaktadır. Buna göre SQL'de aynı alan içinde tutmak istediğimiz veriler tablolarda tutulurken, NoSQL'de koleksiyonlarda tutulmaktadır. Resim 4.2.'de sol kısımda gösterilen "saadin" isimli veri tabanının koleksiyonları account, cihaz, cihazekle, token, user ve system.indexes olarak görülmektedir. Resim 4.2.'de sağ kısımda ise "cihaz" koleksiyonunun verileri liste halinde görüntülenmektedir. Önemli bir nokta ise buradaki liste kavramının MonjaDB yazılımı ile yapılıyor olmasıdır. Normal şartlar altında ham veriler MongoDB veri tabanında JSON olarak tutulmaktadır. MongoDB üzerinde JSON olarak tutulan kullanıcı ve cihaz bilgileri çizelge 4.1.'de gösterilmektedir.

Çizelge 4.1. Koleksiyon içerisinde verilerin saklanma biçimi

"account" Collection JSON Görünümü	"cihaz" Collection JSON Görünümü

{
"_id":
ObjectId("55227aed88e5f382b2006c59"),
"_class":
"com.saadinoyucu.model.Account",
"un" : "saadin",
"p" : "123456",
"ep" : "saadin@gazi.edu.tr",
"createdDate":ISODate("2015-04-06T12:24:13.480Z")
}
	{
	"_id":
	ObjectId("55264bfa88e5f0402f9d41e9"),
	"_class":
	"com.saadinoyucu.model.Cihaz",
	"kullaniciID":
	"55227aed88e5f382b2006c59",
	"cihazID" : "2e6e0c8b-12c7-4353-be5c-7a2486c52db8",
	"cihaz_adi" : "Okul DHT11",
	"sicaklik" : "22.00",
	"nem" : "36.00",
	"createdDate":ISODate("2015-04-09T09:52:58.135Z")
	}

Temelde JSON yapısını saklayabilen, REST işlemlerine karşılık verebilen ve aynı zamanda da çoklu yazma ve okuma işlemlerinde üstünlük gösteren yatay ölçeklenebilir bir

NoSQL veri tabanını hazır hale getirilmiştir. Bundan sonraki işlem ise web servis katmanının inşa edilmesidir.

4.3.M2M Platformu Servis Katmanının Hazırlanması

OneM2M mimarisinde belirtildiği gibi servis yetenekleri katmanı oldukça önemlidir. "Web Servisler" konu başlığında web servis kavramı detaylı bir şekilde ele alınmış ve platformda RestFul web servisler kullanılmasına karar verilmiştir. RestFul web servislerin inşa edilebilmesi için birçok yapı mevcuttur. Bunların başında ise Spring hazır yapısı gelmektedir. Spring, Java tabanlı programlama ve uygulamalar geliştirmek için kapsamlı bir yapılandırma modeli sağlamaktadır. Spring Rod Johnson tarafından geliştirilmiş olup ilk olarak 2003 yılında Apache 2.0 lisansı altında yayınlanmıştır [35].

Spring, geliştiricilere sunduğu farklı özelliklerle göze çarpmaktadır. Bunların başında Bağımlılık Enjeksiyonu (Dependency Injection) yazılım geliştirme yöntemi gelmektedir. Bu yapı geliştirilen yazılımın daha anlaşılabilir ve kolay gerçekleştirilmesini sağlamaktadır. Görünüm Odağı (Aspect Oriented) ise yazılım içerisinde servislerin çalışma zamanlarının kontrolü ve doğru zamanda yapacakları işi gerçekleştirmelerini sağlamaktadır. Bu özelliklerinin yanında Spring, geliştiricilere sadece ihtiyacı olan sınıfları ve paketleri kullanıp diğerlerini kullanmamak gibi birçok kullanım esnekliğini de beraberinde sunmaktadır. Spring versiyonlarından 3.0 Rest mimari yapısını desteklemektedir geliştirilen sistemde ise model sürüm 3.2.3 kullanılmıştır. Ayrıca geliştiricilerin işlerini kolaylaştırmak adına bazı ek bilgi notları geliştirilmiştir. Ek bilgi notları sayesinde kullanıcıya göstermek istenilen ya da kaydetmek istenilen bilgiler kolayca tanımlanabilmektedir [36].

Spring, yazılım geliştiricilerin işlerini kolaylaştırmak ve geliştirilen yazılımın taşınabilirliğini arttırmak için temelde iki farklı yönteme başvurmaktadır. Bu yöntemlerin ilki "Maven" projeler ikincisi ise "Gradle " projelerdir [36]. Maven projelerinin geliştirilmesi ve kullanımı kolay olduğundan bu tez çalışmasına Maven projesi olarak başlanılmıştır. Yazılım geliştirme sırasında bu kullanım tarzı geliştiricilere önemli bir kolaylık sağlamaktadır. Proje geliştirirken bazı noktalarda farklı mimarileri sistemimize entegre etmek gerekebilmektedir. Bu gereksinim ise Maven projelerinde bulunan "pom.xml" dosyasından eklenecek özellikler ile kolaylıkla giderilebilmektedir.

Yazılım geliştirme ortamının başarılı bir şekilde hazırlanması gerekmektedir. Windows işletim sistemi üzerinde gerçekleştirilen projenin yazılım gereksinimleri aşağıdaki gibidir;

- Eclipse Kepler
- Java Standart Edition Version 8
- Apache Tomcat 7.0

Eclipse Kepler sürümü web yazılımları ve web servisleri geliştirmek için biçilmiş kaftandır. Üzerinde barındırdığı farklı araç çubukları ve Maven projelerini desteklemesi açısından bu tez çalışmasında Kepler sürümü kullanılmıştır. Java sürümü olarak 8 seçilmesinin nedeni ise Spring sürüm 3.0 ve 4.0 ile sorunsuz çalışıp NoSQL bağlantı yapılarını da desteklemesidir. Apache Tomcat sunucu bakımından kendini kanıtlamış ve kullanımı kolay yerelde çalışabilen bir sunucu yapısıdır. Bu nedenle geliştirilen sistemde Apache Tomcat 7.0 versiyonu kullanılmıştır. Sistem geliştirilirken karmaşıklık yaşamamak adına tüm kurulumlar veri tabanı ile aynı dizin altında toplanmıştır.

4.3.1. Web Servis İskeletinin Oluşturulması

Maven projeleri iskelet oluşturma adına işleri oldukça kolaylaştırmaktadır. Geliştirme süreçlerinin basitleştirilmesi, standartlaştırılması ve kütüphane bağımlılığını ortadan kaldırmak adına Maven projesi olarak işe başlamak geliştiricilere kolaylıklar sağlamaktadır. Ayrıca Maven eclipse veya başka geliştirme ortamlarında sorunsuz çalıştığından geliştiricilere esneklik kazandırmıştır. Maven projesi açıldığında “pom.xml” dosyası karşımıza gelmektedir. Geliştirici bu dosya içerisinde proje konfigürasyonu ve gerekli bağımlılıkları tutar. Çizelge 4.2.’de “pom.xml” dosyası ile oluşturulan iskeletin üzerine eklenen bazı Spring özellikleri ve MongoDB bilgileri gösterilmektedir.

Çizelge 4. 2. Spring ve MongoDB işlemleri için iskelet yapısı

Pom.xml

```
<project xmlns="HTTP://maven.apache.org/POM/4.0.0"
xmlns:xsi="HTTP://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="HTTP://maven.apache.org/POM/4.0.0
HTTP://maven.apache.org/maven-v4_0_0.xsd">
        <modelVersion>4.0.0</modelVersion>
        <groupId>com.m2m-psiag.web</groupId>
        <artifactId>proje</artifactId>
        <packaging>war</packaging>
        <version>0.0.1-SNAPSHOT</version>
        <name>proje Maven Webapp</name>
        <url>HTTP://maven.apache.org</url>
        <properties>
                <spring.version>3.2.3.RELEASE</spring.version>
                <spring.security-version>3.1.4.RELEASE</spring.security-version>
                <mongo.java.driver-version>2.11.1</mongo.java.driver-version>
                <org.dbunit.mongodb-version>0.0.1</org.dbunit.mongodb-version>
        </properties>
        <dependencies>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-webmvc</artifactId>
                        <version>${spring.version}</version>
                </dependency>
                <dependency>
                        <groupId>org.mongodb</groupId>
                        <artifactId>mongo-java-driver</artifactId>
                        <version>${mongo.java.driver-version}</version>
                </dependency>
        </dependencies>
</project>
```

Çizelge 4.2.'de görüldüğü gibi en üst kısımda proje ile ilgili bilgiler tanımlanır. Maven versiyonu belirlenir, ardından ise kullanılan eklentilerin versiyonları belirlenir. "dependency" ile .jar uzantılı dosyalar yerel sunucuya indirilir böylelikle sistem her çalıştırıldığında uzak bilgisayara bağlanma gereksinimi ortadan kalkar. "groupId" yazılım paketinin ismi "artifactId " ise projenin ismini belirtir. Görüldüğü gibi sistem için gerekli bilgiler "pom.xml" dosyasında tanımlanmaktadır. Yeni eklenecek eklentiler, özellikler ve kütüphaneler bu dosya içerisinden sisteme eklenebilecektir. Örneğin Spring yapısının güvenlikle ilgili olan özellikleri kullanılmak istendiğinde çizelge 4.3.'te bulunan "Spring Security" bilgilerini eklemek yeterlidir.

Çizelge 4.3. Spring Security özelliklerinin Pom.xml'e eklenmesi

Pom.xml Spring Security Bilgileri
<dependency> <groupId>org.springframework.security</groupId> <artifactId>spring-security-web</artifactId> <version>${spring.version}</version> </dependency> <dependency> <groupId>org.springframework.security</groupId> <artifactId>spring-security-config</artifactId> <version>${spring.version}</version> </dependency>

Çizelge 4.3.'te görüldüğü gibi oluşturulan iskeletin üzerine eklemek istenilen farklı özellikler kolaylıkla eklenmektedir. Böylelikle platform için gerekli olan test işlemleri, veri tabanı özellikleri ve vb. diğer tüm yetenekler kolaylıkla iskelet üzerine adapte edilebilecektir.

4.3.2. Veri Tabanı Bağlantıları ve Temel Modelin Oluşturulması

M2M Platformunun iskeleti oluşturulduktan sonra yapılması gereken ilk iş veri tabanı konfigürasyonu ve temel modelin oluşturulmasıdır. İskelet oluşturulurken Spring web mvc

özelliği sisteme eklenmiştir. Spring MVC (Model-View-Controller) kısaca Spring temel alınarak web tabanlı projeler oluşturmayı sağlayan bir yapıdır. Spring mvc istek-yanıt prensibine göre çalışmaktadır. Buna göre istemci sunucuya HTTP üzerinden bir istek gönderir gelen istek ön kontrolcü (Dispatcher Servlet) tarafından karşılanır ve isteğe uygun kontrolcüye (controller) İşleyici Eşlemesi (Handler Mapping) ile gönderilir. Kontrolcüde isteğe uygun hazırlanan yanıtlar ön kontrolcü tarafından kullanıcıya sunulur.

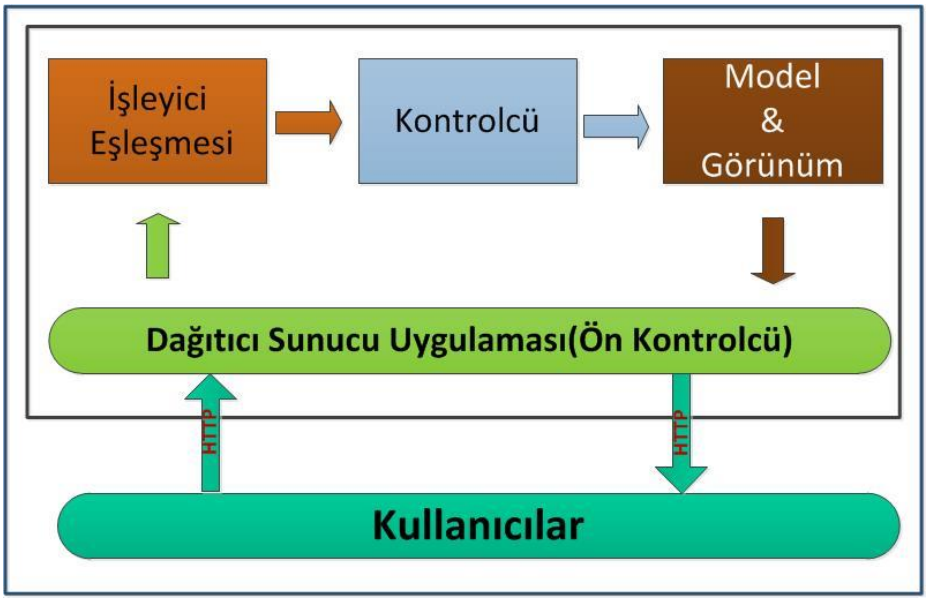
Platformda kullanılan veri tabanına göre gerekli eklemeler iskelet üzerinde mevcuttur. Yapılması gereken tek şey bağlantının gerçekleştirilmesi ve kontrol edilmesidir. Bunun için gerekli bağlantı kod parçası aşağıdaki gibidir.

```
MongoDbFactory mongoDbFactory() throws Exception {
        UserCredentials userCredentials = new UserCredentials("root", "123456");
                return new SimpleMongoDbFactory(new Mongo("127.0.0.1"), "saadin");
        }
```

Yazılım tasarımında her işlem bir paket altında sınıflarla gerçekleştirilmiştir. Böylelikle projeye şeffaflık kazandırılmıştır. Yukarıdaki kod “MongoConfiguration” sınıfı içerisinde yer almaktadır. Kod incelendiğinde veri tabanı yayın adresi, veri tabanı adı, kullanıcı adı ve şifresi kolaylıkla görülmektedir. Veri tabanı bağlantısı sağlandıktan sonraki işlem ise çok fazla tekrarı yapılan ve her veri tabanı işlemlerinde kullanılacak olan temel modellerin oluşturulmasıdır. Örneğin sisteme her bir kayıt eklendiğinde bu kayıtla birlikte bir de kayıt tarihi eklenmektedir. Bu ve buna benzer her defasında yapılan işlemler temel modelde tanımlanır. Bu çalışmada temel modele örnek teşkil etmesi açısından sadece oluşturulma tarihi eklenmiştir. “Model” paketinde “BaseModel” adında oluşturulan sınıf içerisine sıklıkla kullanılan nesneler yerleştirilmiştir. Oluşturulma tarihi için kullanılan nesnenin adı “createdDate” olarak “Date” tipinde belirtilmiştir. Böylelikle her model için ayrı ayrı oluşturulma tarihi nesnesi oluşturmak yerine temel model de bu işlem tanımlanmıştır. Böylelikle esneklik ve sadelik kazandırılmıştır.

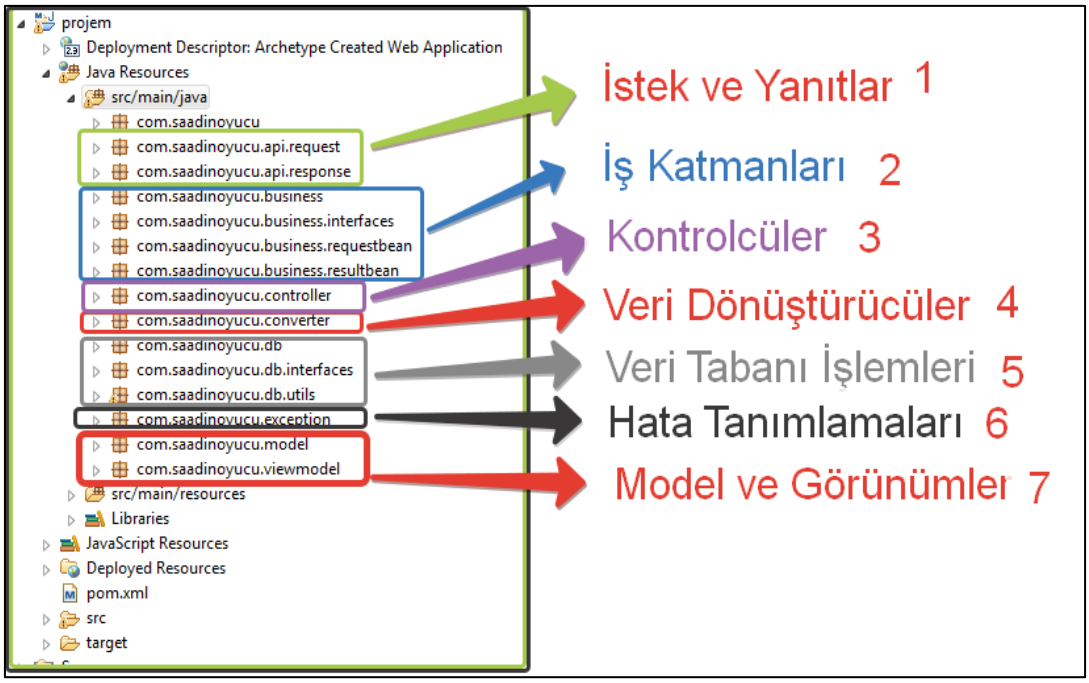
4.3.3. Servislerin Tanımlanması

RestFul mimarisinde istek veya yanıtlar belirli bir düzen ve hiyerarşi içerisinde gerçekleşir. Bu düzen sistemlerde karmaşıklığı azaltırken aynı zamanda da performansı arttırmaktadır. Servis geliştirmelerde kullanılan Spring mvc RestFul mimarisi şekil 4.2.'de gösterilmiştir.



Şekil 4. 2. Spring MVC ResfFul web servis mimarisi [37]

Şekil 4.2.'deki mimari, HTTP üzerinden gelen istekleri ön kontrolcüde karşılar ve istekleri gereken yerlere dağıtır. Dağıtım işleminde istekler işleyici eşlemesi yapılarak gerekli kontrolcülere aktarılır. Eğer kontrolcülerde bir hata yoksa isteklere yanıtlar oluşturulması için modeller ve görünümler düzenlenir. Düzenlenen yanıtlar tekrar dağıtıcı sunucu uygulaması üzerinden kullanıcıya sunulur. Her istek ve her yanıt HTTP üzerinden yapılır. Tez çalışmasında geliştirilen web servisler bu yapıya göre düzenlenmiştir. Şekil 4.3.'te kullanılan mimarinin yazılama dönüşüm aşaması gösterilmektedir.



Şekil 4. 3. RestFul web servis mimarisinin yazılıma dönüştürülmesi

İstek yanıt prensibine göre çalışan Rest mimarisi şekil 4.3.'te görüldüğü gibi yazılıma aktarılmaya çalışılmıştır. Şekle göre HTTP üzerinden alınan istekler Spring'in yazılımcıya sunduğu ön kontrolcüde yorumlanıp işleyici eşleşmesi yapılır böylelikle kontrolcülere doğru istekler gönderilir. İstek ve yanıtların tanımlanması şekil 4.3.'te görülen 1 numaralı yazılım paketlerinde yapılmıştır. Kontrolcülere gelen istekler kontrol edilir ve isteğe göre yanıtlar hazırlanır. Kontrolcülerin tanımlanması ise şekilde 3 numarada gösterilen kontrolcüler paketinde yapılmıştır. Kontrolcüler yanıtları hazırlarken veri tabanına yazma veya okuma işlemi için isteklerde bulunabilir, bu durumda karmaşıklık yaşamamak adına iş katmanları geliştirilmiş ve yazılım içerisinde de katmanlı bir yapı oluşturulmuştur. Şekil 4.3.'te 2 numarada gösterilen iş katmanlarında kontrolcüler tarafından istenilen veya yerine getirtilmesi gereken görevler gerçekleştirilir. Veri tabanı işlemleri 5 numarada tanımlanan paketlerde gerçekleştirilmiştir. Veri tabanı arabirimleri ve bağlantıları bu paketlerde tanımlanmıştır. Model ve görünümler yanıtların nasıl hazırlanacağı ve sunulacağı ile ilgilidir. Şekil 4.3.'te model ve görünümlerin yanı sıra veri dönüştürme görevlerinin yapıldığı işlemler 4 ve 7 numaralı paketlerde verilmiştir. Bunun nedeni ise bazı durumlarda verilerin belirli formatlara dönüştürülüp kullanıcıya gönderilmesidir. Örneğin sistemde veri tabanı liste isteklerinde veri dönüştürme işlemi yapılıp kullanıcıya dizi halinde yanıtlar hazırlanır. 6 numaralı paket ise hata tanımlamalarının yapıldığı paket olup, bir hata ile

karşılaşıldığında kullanıcıya sunulacak hata mesajları ve durum mesajları yer almaktadır. Spring'e göre standart durum mesajları aşasıdaki gibidir [38];

- 1XX - Bilgilendirme
- 2XX - Başarı
- 3XX - Yönlendirme
- 4XX - İstemci hatası
- 5XX - Sunucu hatası

Sistem geliştirilirken istekler GET veya POST komutları ile alınabilmektedir. Fakat güvenirlik ve süreklilik açısından GET kullanmak sistemde bazı tıkanıklıklara neden olmaktadır. Bunun nedeni ise istemcinin URL üzerinden sürekli veri göndermesi bir zaman sonra sunucu tarafından güvenlik duvarına takılması ve virüs olarak görülmesidir. GET küçük çaplı ve fazla veri akışının olmadığı yerlerde rahat kullanım sağlamaktadır. Geliştirilen M2M platformuna istemcilerden sürekli veri akışı olması nedeniyle bütün işlemlerde POST kullanılmıştır. POST edilen her bir istek "200 OK" durum bilgisini döndürdüğünde sistemin başarılı bir şekilde çalıştığı anlaşılmaktadır. Sistemde her bir istek ve yanıt JSON tipinde yapılmaktadır. Böylelikle veri dönüşümüne gerek kalmadan sistem tarafından veri tabanına kaydedilmektedir.

4.3.4. Servislerin Güvenliği

URL üzerinden yapılan işlemler diğer yazılım sistemlerine göre daha güvensizdir. Bunun nedeni ise kolayca görünür olmalarıdır. Özellikle GET metodu URL üzerinden açık bir şekilde işlemleri gerçekleştirmesi güvenlik işlemlerini oldukça zorlaştırmaktadır. Fakat sistemde GET metodunun hiçbir şekilde kullanılmaması güvenlik bakımından zafiyetleri bir miktarda olsa ortadan kaldırmaktadır.

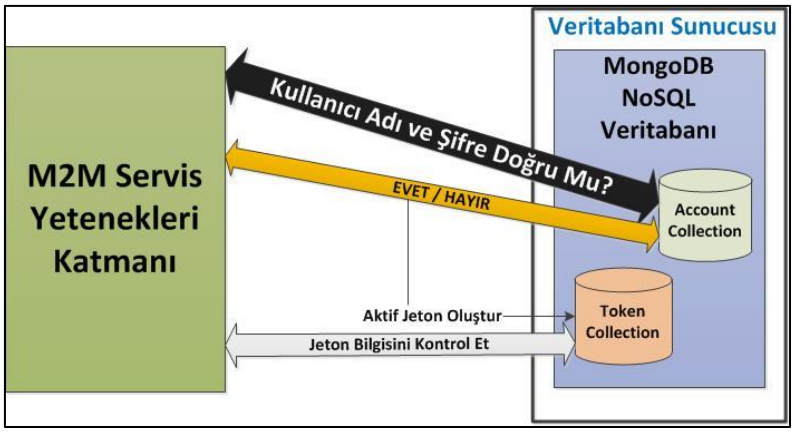
Rest mimarisinde işler genel olarak istemci tarafında yapılmaya çalışılır böylelikle sunucu fazla yorulmamış olur. Rest mimarisini kullanan en büyük örnek Twitter platformudur. Twitter'in API'leri incelendiğinde ve OneM2M'in güvenlikle ilgili çalışmalarına bakıldığında normal yazılım yaklaşımlarındaki gibi oturum açma ve oturum bilgilerini sunucuda tutmak yerine jetonlar (token) üretilerek kontrol sağlanmaya çalışılmıştır. Bu

yaklaşım sunucuyu yormazken aynı zamanda da yeni bir güvenlik anlayışı kazandırmaktadır.

Bu tez çalışmasında geliştirilen M2M platformunda oturum ve kullanıcı kontrolleri jetonlarla yapılmıştır. Bunun için ise sistemde kaydı olan her bir kullanıcı, kullanıcı adı ve şifresi ile sisteme giriş isteği yaptığında web servis katmanı kullanıcıyı veri tabanından kontrol etmekte, kullanıcı kaydı sistemde var ise kullanıcıya yanıt olarak benzersiz ve bir saat süresi olan bir jeton ve kullanıcı ID bilgisini dönmektedir. Kullanıcı herhangi bir işlem yapmak istiyorsa kullanıcı ID ve jeton bilgisini web servis katmanına göndermek zorundadır. Web servis katmanı gelen her istekte kullanıcı ID ve jeton bilgilerini inceler ve onay verirse istek değerlendirilmektedir. Jeton süresi bitmiş ise kullanıcının tekrar oturum açması istenir yanlış jeton veya kullanıcı ID değeri ile istekte bulunulursa kullanıcıya yanlış işlem yaptığı bilgisi gönderilmektedir. Bu durum şekil 4.4.'te gösterilmektedir.

Şekil 4. 4. Jeton bazlı oturum kontrolü

Sistemde bu tarz bir güvenlik çalışmasının yapılması ve başarılı bir şekilde gerçekleştirilmesi HTTP üzerinden gerçekleştirilen işlemlerin güvensizliğini azaltmaktadır. Çünkü herhangi bir yerde oturum bilgileri tutulmamakta, istekler süresi bitmemiş jetonlarla gerçekleştirilebilmektedir. Ayrıca veri tabanı işlemlerinin web servis katmanı ile yapılması jetonlar sayesinde veri tabanı güvenliğini de arttırmaktadır. Çünkü Rest mimarisinde kaynakların bilgisi tutulmamakta ve bu nedenle veri tabanı bilgisi kullanıcıya asla sunulmamaktadır. Şekil 4.5.'te bu durum gösterilmektedir.



Şekil 4. 5. Jetonların üretilmesi ve kontrolü

Jetonlar 36-48 karakter arasında, içerisinde harf ve rakamlar olacak şekilde üretilmektedir. Jetonları çözmek kötü niyetli kullanıcıların işlerini oldukça zorlaştırmaktadır. Ek-1'de sunulan yazılım kodları içerisinde jetonların üretilmesi ve kontrol edilmesi için geliştirilen kod parçacığı yer almaktadır.

4.3.5. Servislerin Test Edilmesi

Web servisler geliştirilirken uygun test ortamını oluşturmak oldukça önemlidir. Google Chrome internet tarayıcısı kullanıcılara üzerinde barındırdığı eklentilerle RestFul web servislerini test etme şansı vermektedir. Bu eklentiler Postman ve Advanced Rest Client'dir. Sistem testi için iki farklı eklentide kullanılmıştır. Advanced Rest Client ile yapılan sistem testi ve web servislerin çalışma biçimleri açısından örnek teşkil eden kullanıcı girişi işlemi çizelge 4.4.'te gösterilmektedir. Diğer servisler ise ek-2'de sunulan servis dokümanında yer almaktadır.

Çizelge 4.4. Kullanıcı girişi servis işlemi

Servis Adresi	HTTP://localhost:8080/saadin/service/account/login

İstek Formatı	{
	"username" : "deger",
	"password" : "deger"
	}

Örnek İstek	{
	"username" : "saadin",
	"password" : "123456"
	}

Cevap Formatı	{
	exceptionCode: 0
	hasException: false
	token: "deger"
	userId: "deger"
	validatormessage: null
	}

Örnek Cevap	{
	exceptionCode: 0
	hasException: false
	token: "3e500d06-6c53-444a-8e81-18abc99fb34f"
	userId: "5507d0b8510dccbfbb9b9fc0"
	validatormessage: null
	}

Servis Açıklama	Cevap olarak gelen hata detayları false ve null ise sorun yok demektir. “token” sistemde aktif kalma süresinin de belirtildiği değişken olup aynı zamanda da platform üzerinden başka işlem yapılırken her seferinde kontrol edilmektedir. “userId” ise kullanıcının ID bilgisidir.

Çizelge 4.4.'te gösterilen servis adresi isteğin yapılacağı adrestir. İstek ve cevap formatları görüldüğü gibi JSON formatında olup, kullanıcı istekte bulunduğu anda kendisine özel bir jeton üretilmektedir. Böylelikle oturum bilgileri tutulmadan servis işlemleri yerine getirilebilmektedir. Test işlemlerinde çıkabilecek hatalar ise durum kontrolleri ile kolaylıkla anlaşılabilmektedir. Surum kontrolleri servisler tanımlanırken belirlenmiş ve kullanıma sunulmuştur.

4.4. M2M Cihaz API'lerinin Hazırlanması

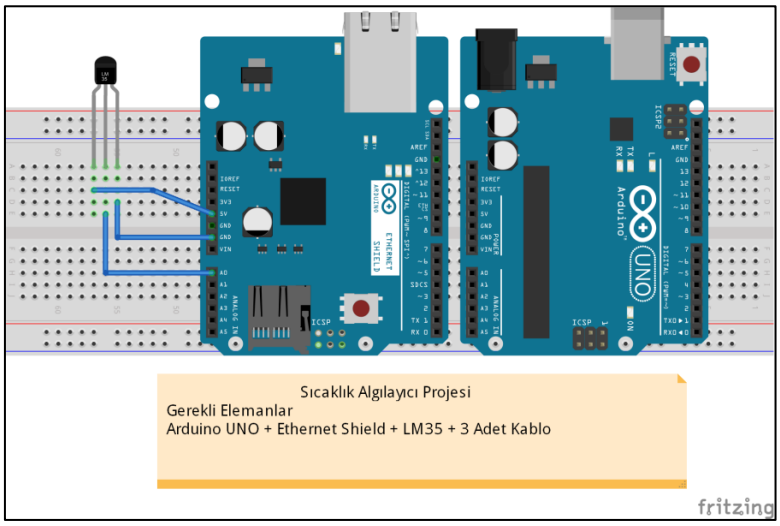
M2M sistemlere veri sağlayan en önemli bileşenler algılayıcılardır. Algılayıcıların doğru ve net bir şekilde çalıştırılması ve düzenli olarak ölçümlerin veri tabanına yazılması gerekmektedir. Algılayıcılar basit yapılara sahiptirler fakat algılayıcıları çalıştırabileceğimiz iletişim işlemlerini de yerine getirebildiğimiz akıllı kartları kullanmak gerekmektedir. Bu tez çalışmasında Raspberry Pi ve Arduino kartları incelenmiş ve Arduino kart kullanılmasına karar verilmiştir.

Kart seçimini yaptıktan sonraki ilk işlem servislerle ve veri tabanıyla iletişimin nasıl olacağını belirlemektir. Bu aşamada düşük maliyetinden dolayı kablolu bir IP şebeke kullanmak işlerimizi oldukça kolaylaştırmaktadır. Bu gün dünya geneline bakıldığında gelişmekte olan ve gelişmiş ülkelerde her evde bir internet bağlantısı mevcuttur. Bu nedenle geliştirilen platformda gerekli olan bağlantı IP tabanlı bir şebeke üzerinden yapılmıştır.

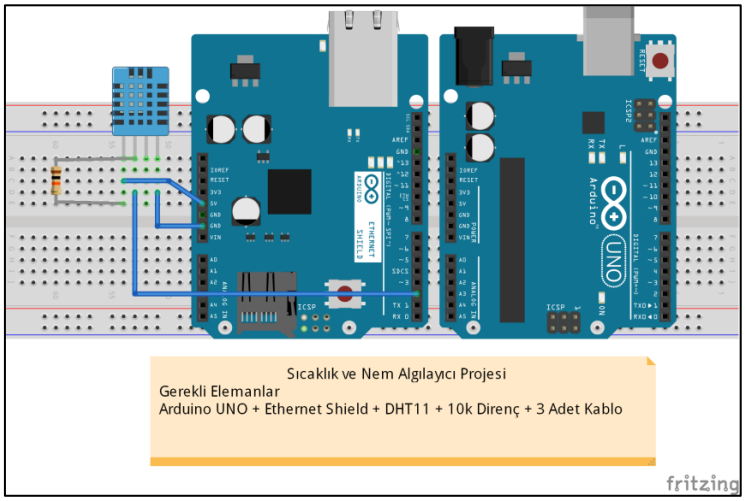
IP şebeke kullanılacak şekilde tasarlanmak istenilen sistem için iki farklı öneri ön plana çıkmaktadır. Bunlardan birincisi akıllı kart üzerinde bir sunucu yayını yapmak ve sunucu üzerinden web servis katmanına erişmek, ikincisi ise akıllı kart üzerinde bir istemci oluşturmak ve istemci yardımı ile web servis katmanına erişmektir. Birinci yöntem kullanım yerine göre bazı avantajlar sunmaktadır. Fakat en önemli eksiği akıllı kart üzerinden sunucu yayını yapıldığında gönderilen istek ve sorguların güvenlik duvarına takılması aynı zamanda da sunucu kart için modem veya yönlendirici üzerinden belirli bir portun akıllı karta yönlendirilmesi gereksinimidir. Bu kullanım tarzı hem karmaşık hem de IP dağıtımı bakımından ekonomik değildir. Çünkü bir ev ortamı düşünüldüğünde içeride IP alabilen cihaz sayısı fazla olabilir fakat dışarıya açılan IP yalnızca bir tanedir ve ev

ortamındaki bütün cihazlar internete bu IP ile bağlanmaktadır. Sunucu yayını yapabilmek için ise benzersiz bir IP kullanmak gerekir.

İkinci yöntem olan akıllı kart üzerinden istemci yayını yapmak, IP bakımından ekonomik ve kullanımı kolay bir yaklaşımdır. Bu nedenle bu tez çalışmasında akıllı kartlar üzerinde istemci yapısını oluşturan bir API geliştirilmiştir. Arduino Uno kullanılarak gerçekleştirilen istemci yapısı Arduino Ethernet Shield ile IP şebekeye aktarılmıştır. Örnek teşkil etmesi açısından ise LM35 sıcaklık ve DHT11 sıcaklık-nem algılayıcı kullanılmıştır. Arduino kullanımı ve bağlantı yapısı LM35 algılayıcısı için şekil 4.6.'te DHT11 algılayıcısı için ise şekil 4.7.'de gösterilmiştir.



Şekil 4.6. LM35 algılayıcı projesi



Şekil 4. 7. DHT11 algılayıcı projesi

Şekil 4.6. ve 4.7.'de görüldüğü gibi Arduino kart üzerinde işlem yapmak oldukça basittir. Örnek olarak kullanılan algılayıcıların kart üzerine montajı ise kolaylıkla yapılabilmektedir. Fiziksel olarak bağlantıları yapılan Arduino sisteminin doğru ve isteğe uygun çalışabilmesi için gerekli kod bloğunun oluşturulması gerekmektedir. Bu noktada ise daha önce hazırlanan ve platforma kayıt olan kullanıcının hizmetine sunulan API'ler devreye girmektedir. Kullanıcı platformdan gerekli API'leri indirir ve Arduino karta yükler. Yükleme işlemi sırasında kullanıcın değiştirmesi ve değiştirmemesi gereken yerlere dikkat etmesi gerekmektedir. Aşağıda API'ler içerisinde kullanıcının değiştirmesi veya değiştirmemesi gereken yerler açıklanmıştır.

- Değiştirilmesi gereken yerler
  - Kullanıcı ID: Platform tarafından her kullanıcı için benzersiz üretilen ve kullanıcıya sunulan ID değeridir.
  - Cihaz ID: Kullanıcının eklediği her cihaz için benzersiz bir şekilde üretilen Cihaz ID değeridir.
  - Cihaz Adı: Platforma cihaz ekleme işlemi yapılırken kullanıcının belirlediği ve cihazını tanımladığı Cihaz Adı değeridir.
  - IPAddress: IP adresi Arduino'nun IP adresidir. Kullanıcı bu değeri kendi kullandığı IP sınıfına göre değiştirebilir.
  - Pin Tanımlamaları: Algılayıcı bağlantılarının yapıldığı Arduino üzerindeki pinleri tanımlamak için kullanılır. Kullanıcının isteğine göre değiştirilebilir.

- Değiştirilmemesi Gereken yerler
  - Mac: Bu değer internete bağlanan her cihaz için olması gereken bir değerdir ve Arduino cihazlar için Mac değeri sabittir.
  - Server: Bu değer web servis katmanının yayın yaptığı adrestir. İstemci bu adrese istekte bulunur.

API'ler üzerinde gerekli değişikler yapıldıktan Arduino artık platformla haberleşmekte ve görevini yerine getirmektedir. API yazılımı tasarlanırken verilerin JSON değerinde gönderilmesi için çalışmalar yapılmış ve başarılı olunmuştur. Bu tez çalışması kapsamında geliştirilen API'lerde istekler ve veri göndermeler JSON tipinde yapılmıştır. JSON tipinde yapılan istekler veya gönderilen veriler JSON olarak servis katmanına kabul edilip JSON dokümanı olarak veri tabanında saklanmaktadır. Böylelikle veriler herhangi bir yerde

dönüşüme gerek duymadan saklanmaya çalışılmıştır. Bu yapı ise M2M sistemlerde hızı arttırmıştır. API'ler ile ilgili kod yapısı ek-3'te sunulmuştur.

4.5. Ara yüzlerin Hazırlanması

Önceki bölümlerde yapılan çalışmalarda düzenli ve standart mimari yapılarına dayanan bir sistem tasarımı yapılmış ve geliştirilerek hazır hale getirilmiştir. Hazırlanan sistem kullanıcı ile etkileşime geçilebilecek herhangi bir ara yüze ihtiyaç duymaktadır. IP şebeke kullanabilen cihazların sayısının artması artık masaüstü uygulamalar yerine web tabanlı uygulamaların yaygınlaşmasına neden olmuştur. Bu nedenle bu tez çalışmasında kullanıcı ile etkileşimde bulunabilecek web tabanlı bir uygulama geliştirilmiştir. Bu bölümde kullanıcı ile etkileşime geçebilecek, kullanılabilirliği yüksek, her yerden ve her sistemden rahatça ulaşılabilecek bir ara yüzün tasarlanması, kodlanması ve sisteme entegre edilmesi işlemleri ele alınmıştır.

Daha önce ara yüzleri HTML5, CSS3 ve JQUERY ile geliştirmeye karar verdikten sonraki ilk işlem ara yüz iskeletinin oluşturulmasıdır. HTML 5 ile gelen bir kullanım şekli olan her işlem ayrı ayrı başlık tanımlama yöntemi gerçekleştirilen M2M platformda başarılı şekilde kullanılmıştır. Her başlık CSS ile kontrol edilmiş ve görsel olarak özellikler eklenmiştir. Platformlarda grafiklerin hareketli bir şekilde kullanıcıya sunulması için ise JQUERY kütüphaneleri kullanılmıştır. Ara yüzler tanımlanırken standartlara uyulması ve kullanılabilirlik açısından renk uyumuna dikkat edilmesi oldukça önemlidir ve bu tez çalışmasında renk uyumu göz önünde bulundurularak çalışmalar yapılmıştır.

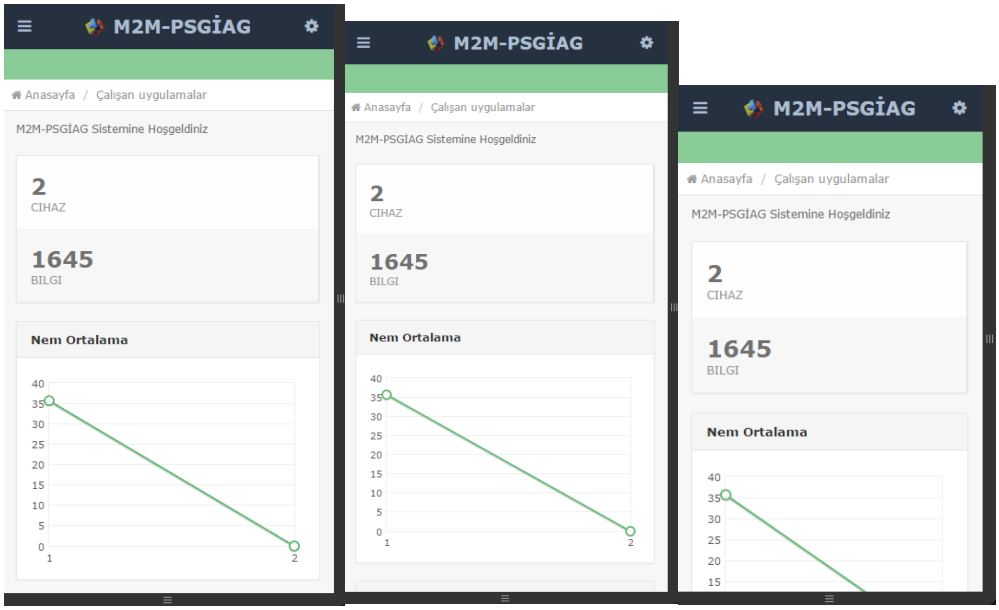
Kullanıcı ile etkileşime geçebilecek cihazların başında masaüstü bilgisayarlar, dizüstü bilgisayarlar, tabletler ve akıllı telefonlar gelmektedir. Bu cihazların ekran çözünürlükleri ve ekran boyutları birbirlerinden farklıdır. Bu nedenle geliştirilen ara yüzlerin her sistemde düzgün çalışabilecek bir yapıda olması gerekmektedir. Bu gereklilik göz önünde bulundurularak geliştirilen sistem her cihazda sorunsuz çalışabilecek bir yapıda geliştirilmiştir. Böylelikle akıllı telefonlar veya tabletler için her işletim sistemine uygun mobil yazılımlar yapmak yerine internet tarayıcı üzerinden düzgün çalışabilen duyarlı bir tasarıma sahip platform ara yüzü kullanıcıya sunulmuş ve işlemlerin yapılması kolaylaştırılmıştır.

Kullanıcı işlemleri, cihaz ekleme, raporlama vb. her işlem için kullanıcının web servis katmanına istekte bulunması gerekmektedir. Bu istekleri yapabilmesi için ise Rest istemci oluşturulması gerekmektedir. Bu gereksinim ise PHP tabanlı bir web sayfası üzerinden karşılanmıştır. Web servis katmanındaki yetenekleri yerine getirebilen istemci yapısının en önemli yönlendirme fonksiyonlarının yer aldığı kod yapısı ek-4 sunulmuştur.

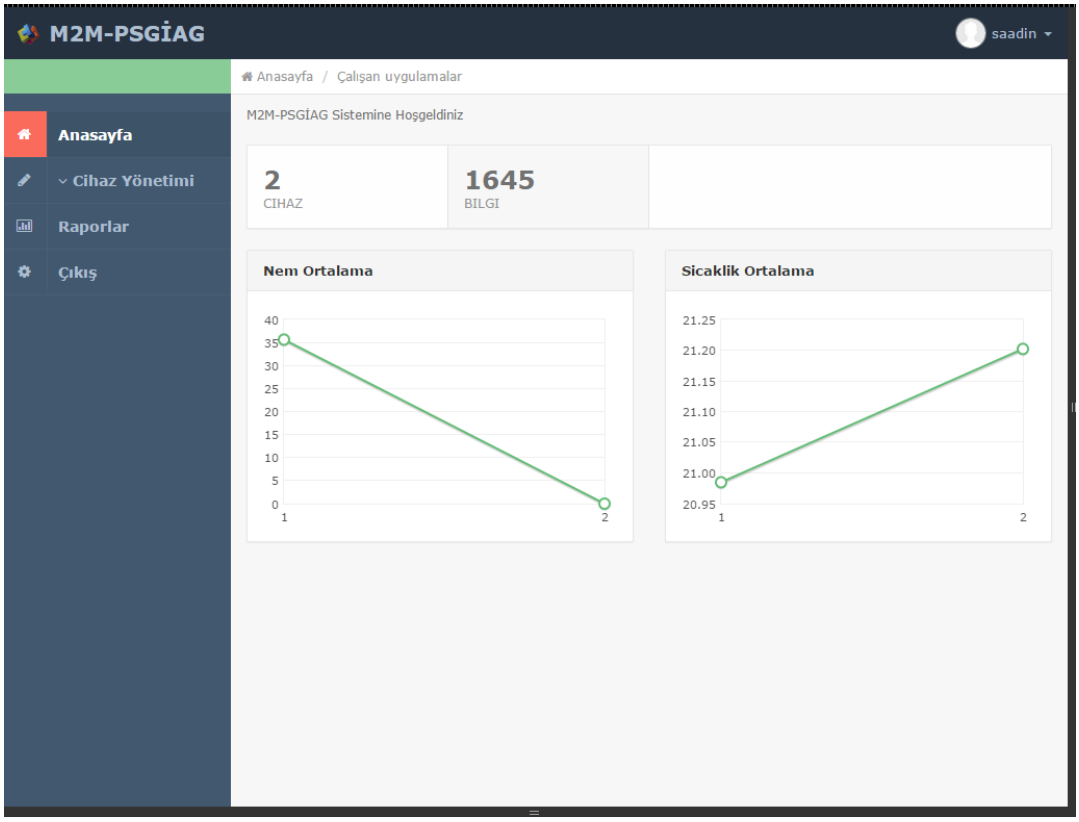
4.5.1. Ara yüzlerin Test Edilmesi

Kullanıcıya sunulacak ara yüzlerin kullanılabilirliğini test etmek açısından farklı yöntemler bulunmaktadır. Bunların başında uzman değerlendirmesi gelmektedir. Geliştirilen web sayfaları uzman değerlendirmesinde renk uyumu, sol tarafta yer alan ve simgelerle desteklenen menü yapısı, sol üst köşedeki logo ve yerleşim yeri, sağ taraftaki çıkış menüsü ve kullanıcıya bilgi vermekte en önemli bileşen olan ekmek kırıntısı yapısının web sayfamızda mevcut olduğu görülmüştür. Bu özellikleri ile diğer platform örneklerine üstünlük sağlamıştır. Bu değerlendirmeler web sayfa tasarımı konusunda deneyimli olan iki uzman tarafından yapılmıştır.

Diğer bir kullanılabilirlik kriteri ise web sayfalarının duyarlı bir tasarımla çözünürlüğe ve ekrana göre özel görünümlere sahip olmasıdır. Bu durumu test etmek için Google Chrome 12 ile gelen web araçları kullanılmıştır. Örnek olarak Iphone 6, Samsung S4 ve HTC One akıllı telefonlarındaki görünümü test edilmiş ve elde edilen görünümler resim 4.3.'te, Ipad 3'teki görünümü ise resim 4.4.'te gösterilmiştir.



Resim 4. 3. Akıllı telefonlarda platformun görünüm şekli



Resim 4.4. Ipad 3'te platformun görünüm şekli

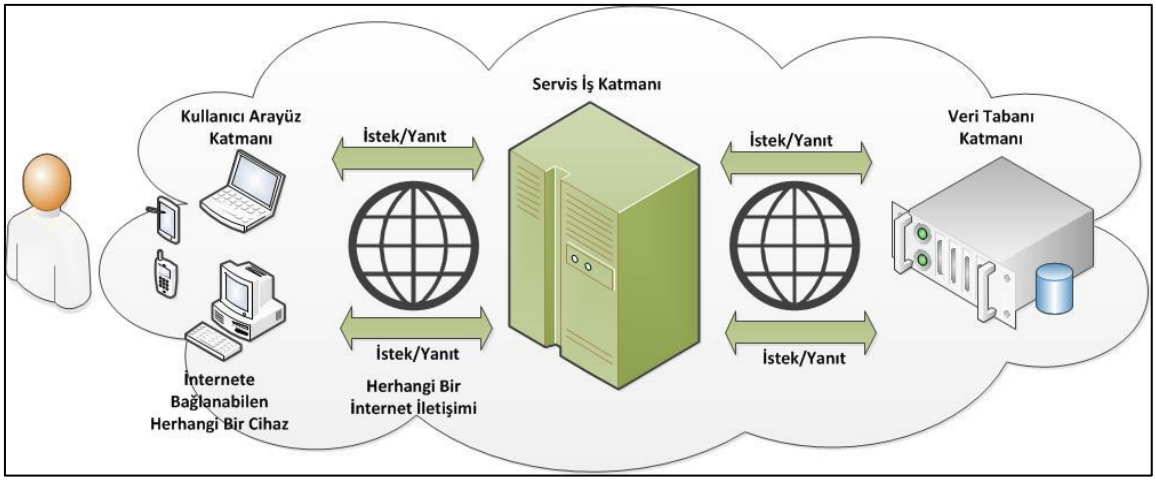
Resim 4.3. ve 4.4.'te de görüldüğü gibi geliştirilen M2M platform farklı cihazlarda farklı görünüme sahip bir şekilde geliştirilmiştir. Bu geliştirme tarzı ile kullanılabilirlik bir kat daha arttırılmıştır. Ayrıca farklı internet tarayıcılarda test edildiğinde platformun sorunsuz

bir şekilde çalıştığı görülmüştür. Böylelikle her internet tarayıcısı ve ekran boyutu için farklı uygulama geliştirme gereksinimi ortadan kalkmış ve hazırlanan platformun genel anlamda kullanılabilirliği arttırılmıştır.

Web platformlarının, uygulamalarının ve sayfalarının birçok kullanılabilirlik testi mevcuttur. Kullanılabilirlik testleri başlı başına bir konu olarak ele alınıp çalışılabilir. Fakat platform geliştirilirken bu gereksinimler de göz önünde bulundurularak sistem geliştirilmeye çalışılmıştır. Böylelikle daha esnek ve sağlam bir yapıya sahip M2M platformu kullanıcıya sunulmuştur.

4.6. M2M Platformunun Kullanım Senaryosu

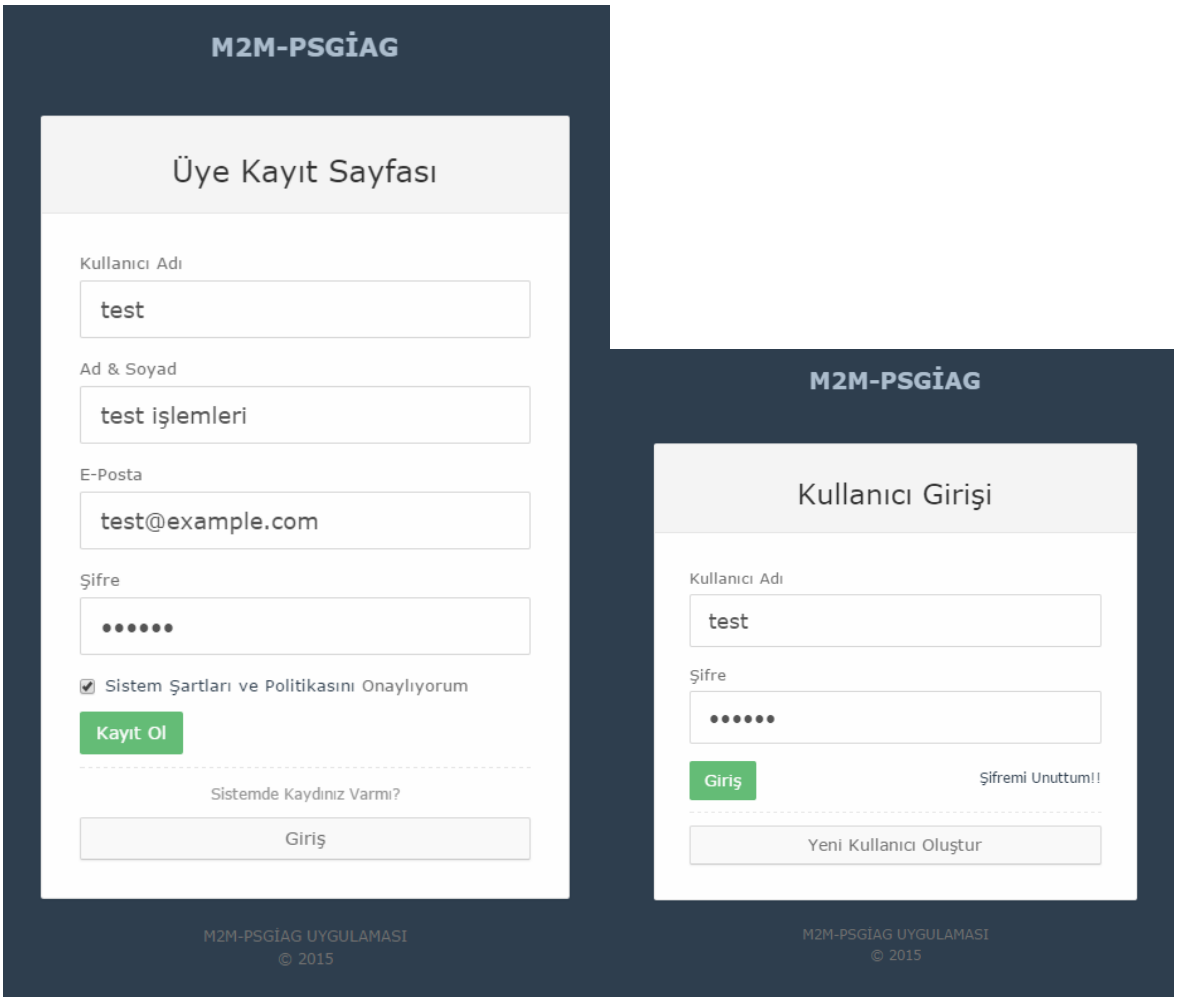
M2M platformu başarılı bir şekilde geliştirildikten ve bazı testler yapıldıktan sonra kullanıcıya sunum için hazır hale getirilmiştir. Geliştirilen platformun kullanım senaryosu şekil 4.8.'de gösterilmiştir.



Şekil 4.8. M2M platformu kullanım senaryosu

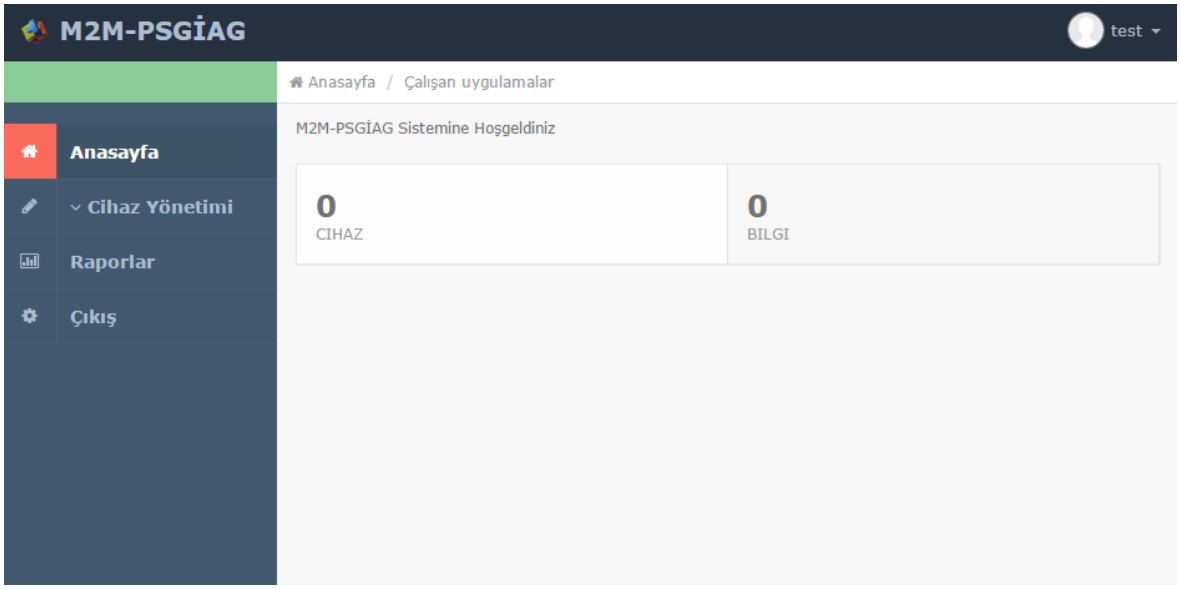
Şekil 4.8.'de görüldüğü gibi kullanıcı veya kullanıcılar herhangi bir cihazdan istekleri yapabilecek ve yanıtları görüntüleyebilecek şekilde web sayfalarına erişim sağlamaktadır. Ardından servis iş katmanı kendisine gelen isteklere uygun yanıtları hazırlar ve bu yanıtları kullanıcının daha net anlayacağı şekilde sunar veya servis katmanı veri tabanına yazma işlemlerinde bulunur. Şekil 4.8.'de görüldüğü gibi katmanlı bir mimariye sahip platformun ölçeklenebilir olması için çaba sarf edilmiştir.

M2M platformuna giriş yapmak isteyen her kullanıcının öncelikle kayıt olması gerekmektedir. Kayıt olan kullanıcıların oturum açma sayfasına yönlendirilip giriş yapması sağlanmaktadır. Bu aşama ile ilgili ekran görüntüleri resim 4.5.'de gösterilmektedir.



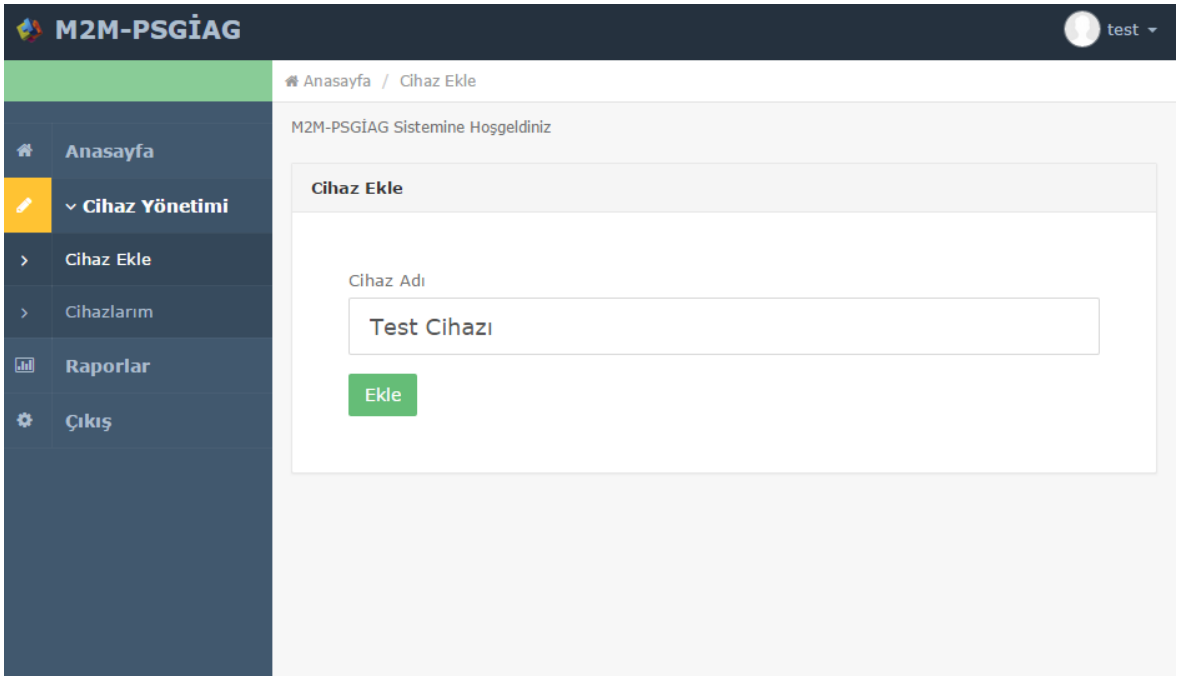
Resim 4.5. M2M platformu kullanıcı kayıt ve kullanıcı giriş ekranı

Kullanıcı giriş yaptıktan sonra kullanıcı karşılama ekranı görüntülenir. Bu kısımda giriş yapan kullanıcının kaç cihazı sisteme eklediği ve bu cihazlara ait kaç bilginin bulunduğu kullanıcıya sunulmaktadır. Ayrıca sistemde cihazlara ait bilgisi bulunan kullanıcılara, cihazlardan gelen veriler grafiksel olarak gösterilmektedir. Kullanıcı karşılama ekranı, ilk kez giriş yapan bir kullanıcı için resim 4.6.'daki görünüme sahiptir.



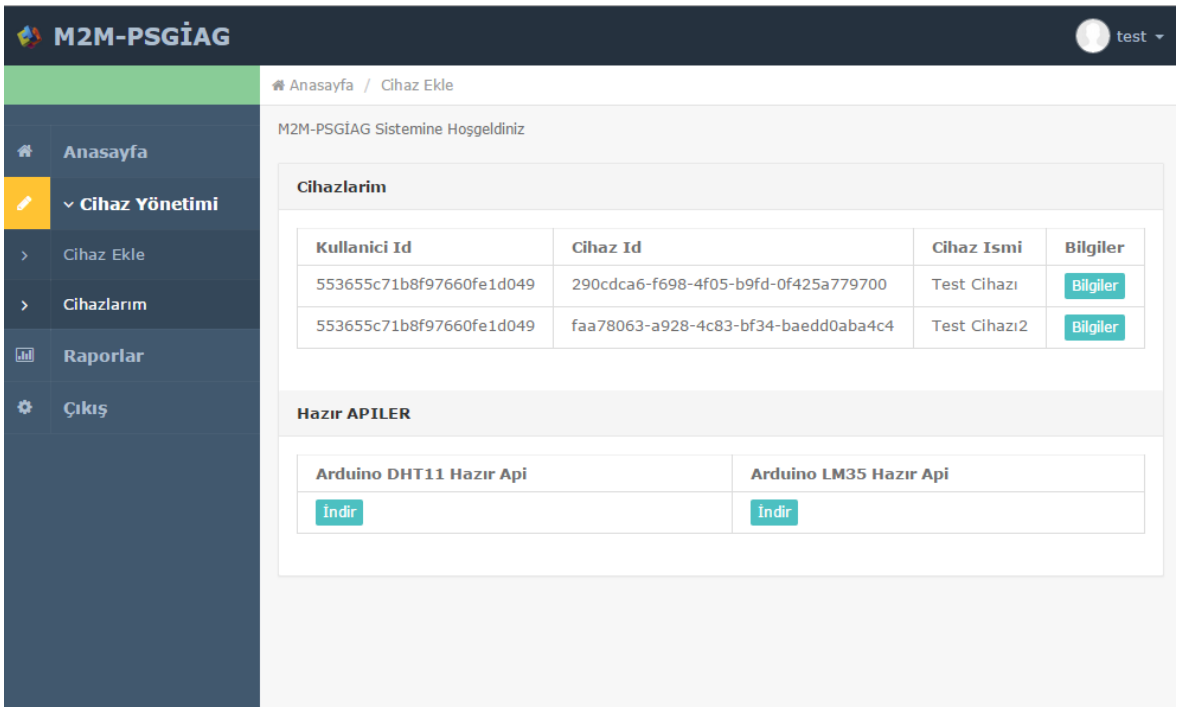
Resim 4.6. Kullanıcı karşılama ekranı

Sisteme ilk defa giriş yapan kullanıcı artık cihazlarını eklemeye başlayabilecektir. Bunun için ise menüde bulunan cihaz yönetimi altındaki cihaz ekle menüsüne girmesi yeterlidir. Kullanıcının karşısına çıkan ekranda bir cihaz adı girilmesi istenmektedir. Kullanıcı cihaz adını girer ve ekle butonuna tıklar. Böylelikle kullanıcının ilk cihazı eklenmiş olur. Bir kullanıcı birden fazla cihaz ekleyebilmektedir. Cihaz ekleme işlemine ait ekran görüntüsü resim 4.7.'de gösterilmiştir.



Resim 4.7. Cihaz ekle ekranı

“Cihazınız başarıyla eklendi” bildirimini alan kullanıcı cihaz yönetimi menüsünden cihazlarım menüsüne gelerek ekli olan cihazlarını ve cihazlara ait bilgileri ayrı ayrı görüntüleyebilir. Bu ekranda ayrıca kullanıcıların algılayıcı cihazlarını programlaması için hazırlanan API’ler yer almaktadır. Örnek olarak geliştirilen platformda Arduino Uno ile kullanılmak üzere LM35 ve DHT11 algılayıcıları için ayrı ayrı API geliştirilip kullanıcıya sunulmuştur. Resim 4.8.’de cihazlarım ekranı gösterilmektedir.



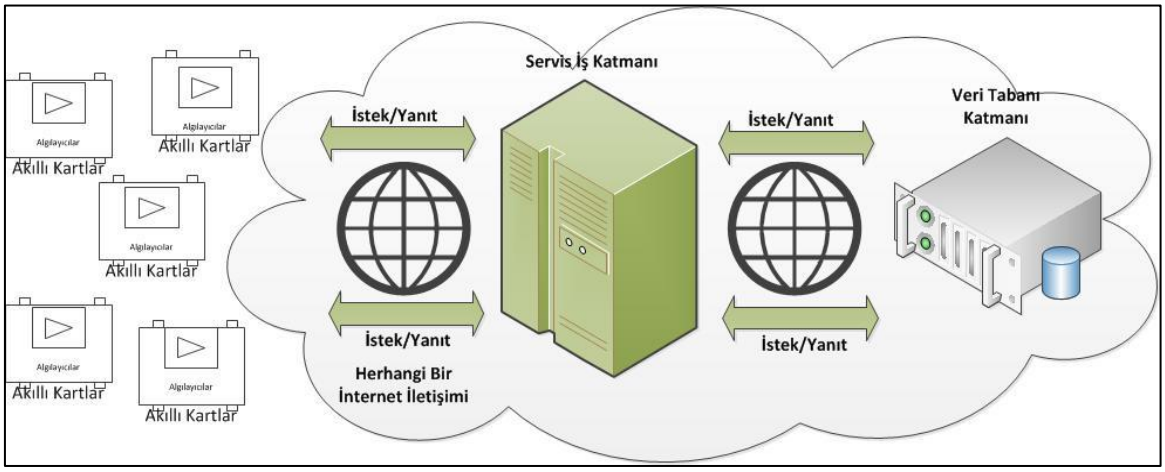
Resim 4.8. Cihazlarım ekranı

Kullanıcı kendi kullanımına uygun API’yi indirir ve ardından API dosyası ile birlikte gelen akıllı Arduino bağlantılarını gösterildiği gibi yapar. Ardından cihaz programlaması için kodlarda gerekli değişiklikleri yapar. Cihaz program kodlarında değişmesi gereken yerler şekil 4.9.’da işaretlenmiş şekilde gösterilmektedir.

Şekil 4.9. Kullanıcının değiştirmesi gereken yerler

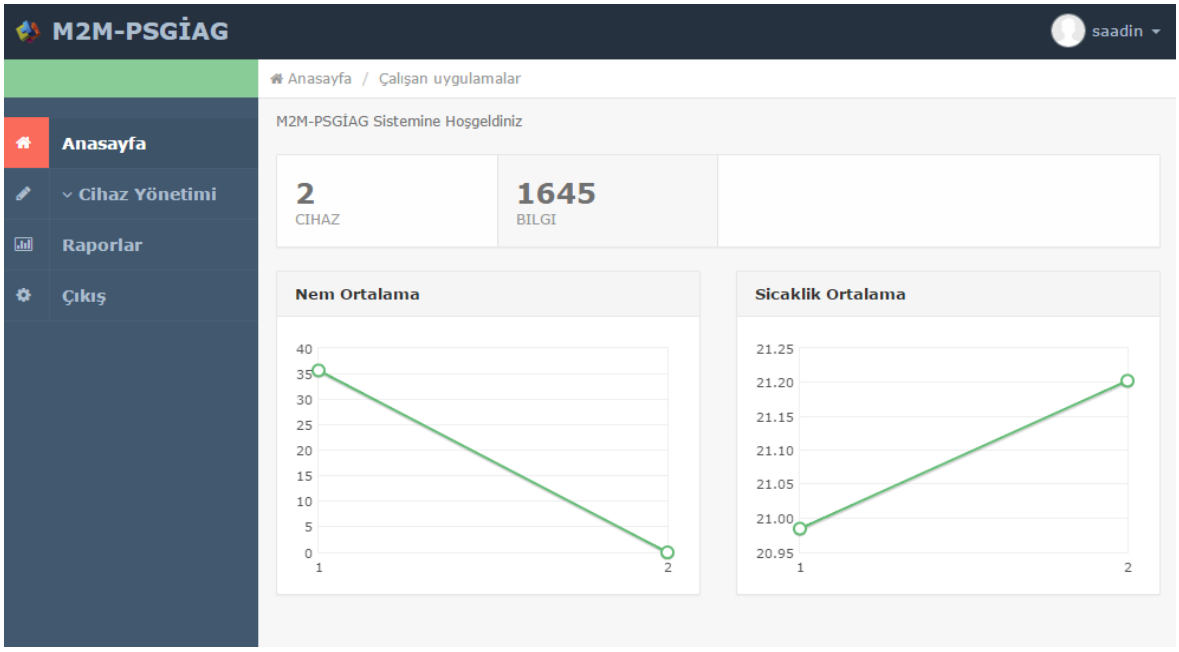
Kullanıcı gereken kart bağlantılarını ve kod içerisindeki gerekli düzenlemeleri yaptıktan sonra artık cihaz M2M platformu ile haberleşmeye başlar ve veriler düzenli olarak alınıp

kaydedilir. Artık kullanıcı sistemde çevrimiçi olmasa dahi M2M cihazlar M2M platformu ile haberleşmeyi sürdürür. Bu durum şekil 4.10.'da gösterilmektedir.



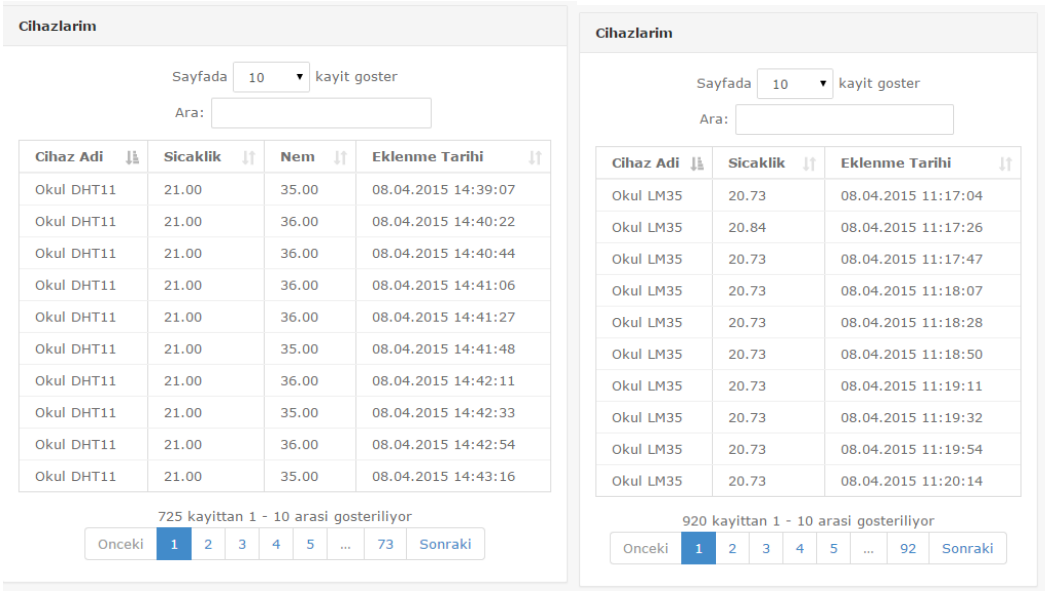
Şekil 4.10. Akıllı kartların M2M platformla haberleşmesi

Kullanıcı M2M platformuna M2M cihazını ekledikten ve başarılı bir şekilde haberleştirdikten sonra sisteme tekrar girip daha önceki verileri analiz edip detaylı bir şekilde inceleyebilir. Sisteme veri akışı olduğu ilk andan itibaren kullanıcı karşılama ekranı değişip, bu ekranda kullanıcıya platforma ekli cihaz sayısı ve bu cihazlardan gelen veri sayısı gösterilmektedir. Ayrıca ortalama sıcaklık veya nem bilgileri de kullanıcıya görsel olarak sunulmaktadır. Örnek ekran görüntüsü resim 4.9.'da gösterilmektedir.



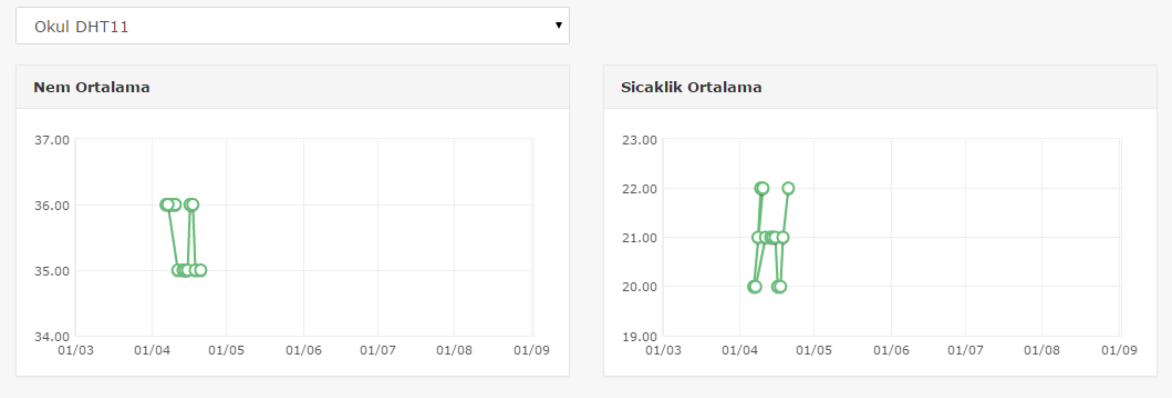
Resim 4.9. Örnek karşılama ekranı görüntüsü

Resim 4.9.'da görüldüğü gibi cihazlara ait nem ve sıcaklık ortalamaları gösterilmektedir. Kullanıcı detaylı bilgi görmek isterse raporlar kısmından günlere göre sıcaklık veya nem ortalamalarını görebilir ve analiz edebilir. Ayrıca Cihaz yönetiminden cihazlarım menüsüne gelindiğinde cihazlara ait bilgiler detaylı olarak tablolar halinde incelenip analiz edilebilir. Burada ki önemli nokta ise grafik ve tablo yapısının dinamik bir şekilde işlem yapıyor olmasıdır. Örneğin bir kullanıcı sadece sıcaklık değerini ölçüyor olabilir. Bu durumda nem bilgisi ile ilgili tablo ve grafiklerin oluşturulmaması gerekmektedir. M2M platform dinamik yapıda geliştirildiğinden bu durum net bir şekilde görülebilir. Resim 4.10.'da tablolar halinde kullanıcıya sunulan nem ve sıcaklık bilgileri sol tarafta, sadece sıcaklık bilgisinin yer aldığı tablo ise sağ tarafta gösterilmektedir.



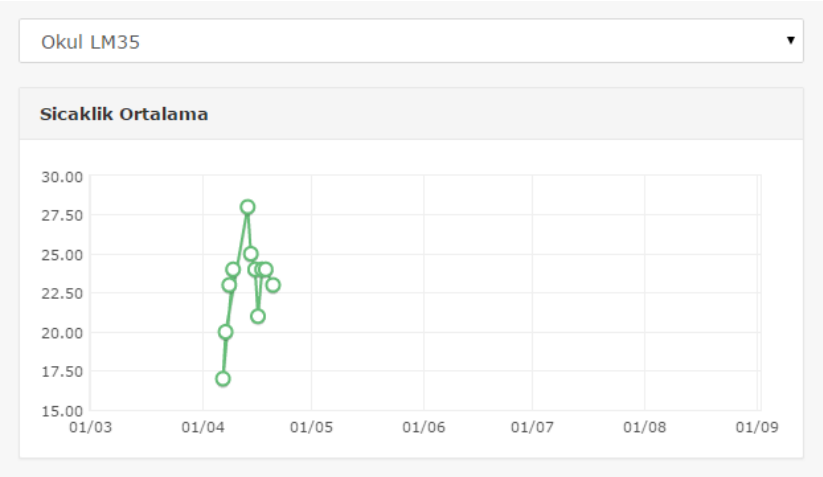
Resim 4.10. Bilgilerin tablolar halinde gösterilmesi

Resim 4.10.'daki yapı ile bilgiler kullanıcılara her cihaz için ayrı ayrı sunulmuştur. Ayrıca JQUERY kütüphanelerinin kazandırdığı dinamizm ile bilgiler her kolonun üzerindeki sıralama işaretleri ile küçükten-büyüğe ya da büyükten-küçüğe sıralanabilir, bilgiler içerisinde arama yapılabilir veya sayfa sayfa bilgiler analiz edilebilir. Resim 4.10.'da görüldüğü gibi bazı cihazlarda iki kolon olan bilgi girişi bazı cihazlarda tek kolon olarak gösterilmektedir. Verilerin grafiksel olarak analizlerinde de kullanıcıya bilgiler sunulurken cihaza göre sunum yapılmaktadır. Resim 4.11.'de platform üzerinde bulunan raporlama hizmetinin ekran görüntüsü gösterilmektedir.



Resim 4.11. Raporlar ekran görüntüsü

Resim 4.11.'de görüldüğü gibi hangi cihazın raporu isteniyorsa grafiklerin hemen üzerinde bulunan menü yardımıyla kolaylıkla seçimi sağlanmaktadır. Resim 4.11.'de "Okul DHT11" cihazına ait sıcaklık ve nem bilgileri detaylı bir şekilde rapor edilmektedir. Rapor üzerinde bulunan noktalara gelindiğinde tarih ve sıcaklık veya nem bilgileri kullanıcıya sunulmaktadır. Resim 4.12.'de ise üzerinde sadece sıcaklık bilgisini tutabilen "Okul LM35" cihazının rapor bilgisi gösterilmektedir.



Resim 4.12. Okul LM35 cihazının rapor bilgisi

Resim 4.11. ve 4.12.'de de görüldüğü gibi raporlar cihaza özel hazırlanmakta ve sunulmaktadır. Dinamik bir yapı çerçevesinde geliştirilen M2M Platformunda cihaza özel tablo sütunları ve raporlar oluşturulup kullanıcıya sunulmaktadır. Böylelikle bilgi kirliliği olmadan kullanıcı net bir şekilde cihazlarını izleyebilmektedir.

Sistemden çıkış yapmak isteyen kullanıcı hem sol tarafta bulunan menüye yerleştirilen hem de sağ üst köşedeki menüde bulunan çıkış menüsü yardımıyla rahatlıkla çıkış işlemini yapabilmektedir.

M2M platformu yüksek okuma ve yazma işlemlerinde kullanıcıları yormayacak şekilde geliştirilmeye çalışılmıştır. Çizelge 4.5.'de M2M platformunun farklı istemci araçları ile yapılan hız performansı test bilgileri yer almaktadır.

Çizelge 4. 5. M2M platform hız performansı

İstek Performansı		Yanıt Performansı			
İstek Tipi	Yanıt Süresi	Servis Katmanı Cevap Gönderme Süreleri			
Kullanıcı Kayıt	12 ms	Cihaz Listeleri		Cihaz Bilgileri	
Oturum Açma	10 ms	Adet	Gönderme Süresi	Adet	Gönderme Süresi
Cihaz Ekleme	20 ms	1	9 ms	10	8 ms
1 Cihaz Bilgisi	26 ms	2	9 ms	570	16 ms
		5	9 ms	830	17 ms
		10	10 ms	1000	21 ms

M2M platformu İntel'in 4 çekirdekli standart bir işlemcisi ve 4 GB RAM'ın bulunduğu bir masaüstü bilgisayarda üzerinde Tomcat 7.0, XAMMP PHP sunucu ve MongoDB veri tabanı aynı anda çalıştırılarak Google Rest istemcileri tarafından yapılan sorgulamalarda elde edilen sonuçlar çizelge 4.5.'deki gibidir. Görüldüğü gibi istek ve yanıtlar milisaniye cinsinden ölçülmekte ve birden fazla veri kaydının istendiği durumlarda bile üstün başarı göstermektedir. M2M platformunun her katmanı ayrı bir bilgisayar ya da sunucuda çalıştırılacak şekilde düzenlenebilir. Bu durumda ise aradaki bağlantının hızı genel anlamda bütün sistemin performansını etkileyecektir. Fakat sistemin genel işleyişi bakımından çoklu kullanıcılara ve cihazlara aynı anda istek verebilmesi adına sistem geliştirme adımlarının her birinde ayrı ayrı performans çalışmaları yapılmıştır. Bu çalışmalar ile genel anlamda sistemin performans hızı önemli derecede arttırılmıştır.

5. SONUÇ VE ÖNERİLER

Günümüz teknolojileri sadece insan değil makine ve nesnelerin de üzerine yoğunlaşmaktadır. Ayrıca algılayıcı teknolojileri sürekli yenilenerek gelişmekte ve maliyetleri düşmektedir. Bu durum M2M uygulamalarının yaygınlaşmasına neden olmaktadır. Sağlık alanından savunma alanına kadar hemen hemen her alanda kullanılan M2M uygulamaları yeni problemleri de beraberinde getirmektedir. Bu problemlerin başında ise her M2M uygulamasının izlenebilir ve kullanıcı ile etkileşime geçebilir bir yapıya sahip olabilmesi adına M2M platformuna ihtiyaç duymasıdır. Günümüzde ise birçok platform gelişigüzel hazırlanarak kullanıcıya sunulmuştur. Standartlara uymayan, kullanılabilirliği ve performansı düşük platformlar birçok farklı yöntemle kullanıcıya sunulmuş durumdadır.

Geleneksel M2M platformu proje ve çalışmaları incelendiğinde alana özel uygulamalar geliştirilmiş ve o uygulamaya özel platformlar hazırlanmıştır. Akademik çalışmalar incelendiğinde genel olarak bir M2M uygulama alanı olan akıllı ev projesi geliştirilmiş ve bu projeye uygun platformlar hazırlanmıştır. Bu tip yaklaşımlarda platformlar sadece alana özel hazırlandığı için farklı alanlarda hizmet verememektedir. Ayrıca akademik çalışmaların çoğu hazır platformlar üzerinde gerçekleştirilmiştir.

Bu tez çalışmasında ise literatürde ve uygulamada yer alan geleneksel yaklaşımların ve daha önce yapılan uygulamaların aksine farklı bir yaklaşım ile M2M platformları ele alınmaya çalışılmıştır. Önerilen yaklaşım ile birden fazla kullanıcının ve birden fazla M2M cihazının aynı anda tek platform üzerinden haberleşebilmesi, izlenebilmesi ve bilgilerin analiz edilebilmesi sağlanmaktadır. Bu yaklaşıma göre yola çıkılan M2M platformunun mimarisi oluşturulurken küresel çapta M2M üzerine standartlar üreten kuruluşların mimari yaklaşımları ele alınmış ve iş katmanına eklenen özellikler ile M2M platformlarının yetenekleri arttırılmıştır. Böylelikle M2M platformuna kayıt olan her kullanıcının kendi M2M cihazını sisteme ekleyebilmesine olanak sağlanmıştır.

Geliştirilen M2M platformu alana özel geliştirilen platformların aksine birden fazla M2M cihazının ve birden fazla algılayıcının sisteme eklenmesine olanak sağlamaktadır. Bu durumda ise farklı problemler meydana çıkmaktadır. Bu problemlerin başında M2M

platformunu kullanan kullanıcıların her birinin ayrı ayrı veriler üretmesi, verilerin ise hangi veri tabanında nasıl ve ne şekilde tutulacağı gelmektedir. Bu tez çalışmasında bu problemi çözmek için literatürde ve uygulamada farklılık oluşturan bir veri tabanı sistemine başvurulmuş ve ilişkisel olmayan veri tabanları sisteme eklenmiştir. Bu kullanım şekli ile M2M platformu veri sisteminin ölçeklenebilirliği ve performansı arttırılmış ayrıca ücretsiz olması sebebiyle maliyeti düşürülmüştür. Diğer bir problem ise çoklu okuma-yazma isteklerinde bulunacak olan kullanıcıların ve M2M cihazların istek-yanıtlarına sorunsuz ve gecikmesiz bir şekilde hizmet verebilmeyi sağlamaktır. Bu problem ise RestFul web servis mimarisi standartlarına uygun bir şekilde geliştirilen web servisler ile çözüme kavuşturulmuştur.

Güvenlik çalışmaları internet tabanlı her uygulama için önem arz etmektedir. Bu tez çalışmasında ise RestFul mimarisinin HTTP üzerinden haberleşmesi nedeniyle güvenlik zafiyetleri oluşması kaçınılmazdır. Fakat web servisler geliştirilirken jeton bazlı kimlik kontrolü ile güvenlik üst seviyeye çıkarılmaya çalışılmıştır. Ayrıca GET metodu yerine sadece POST metodunun kullanılması da güvenliği arttıran diğer bir faktör olmuştur. Bu kullanım tarzı için Twitter mimari yapısı ve OneM2M güvenlik standartları detaylı bir şekilde incelenmiş ve güvenlik çalışmaları bu doğrultuda gerçekleştirilmiştir. Tez çalışması kapsamında yapılan web servis testlerinde jeton bazlı kimlik kontrolünün sorunsuz olarak çalıştığı görülmüştür. Böylelikle yetkili istekler ve yetkisiz istekler prensibi çerçevesinde genel sistem güvenliği bir kat daha arttırılmıştır. Ayrıca servisler geliştirilirken katmanlı bir yapının kullanılması ilerleyen zamanlarda ortaya çıkabilecek güvenlik zafiyetlerine müdahale etmeyi kolaylaştırmıştır. Genel sistem mimarisinin katmanlı bir yapıda olması, her katmanın ayrı ayrı sunucularda ve ayrı yerlerde muhafaza edilebilecek nitelikte olması, her katmanın güvenliğinin hem donanım hem de yazılım bakımından güçlendirilebilirliğini arttırmaktadır.

Geliştirilen M2M platformu kullanıcı dostu bir tasarımla kullanıcıya sunulmuştur. Böylelikle her sistem için ayrı ayrı tasarlanması gereken mobil uygulamalar yerini web sayfalarına bırakmıştır. Kullanıcının internet bağlantısı ve internet tarayıcısını kullanabileceği her sistemden platforma giriş sağlanabilmektedir. Ara yüz testleri bölümünde yapılan çalışmalarda M2M platformu ara yüzlerinin başarılı bir şekilde kullanıcıya hizmet verdiği görülmektedir. Ayrıca farklı istek ve yanıtlar altında sistemin cevap verme süreleri detaylı bir şekilde tez çalışmasının son bölümde sunulmuştur.

Veri dönüşümü yani kaynaktan alınan bir veriyi XML'e çevirip daha sonra başka metotlarla saklamaya karşı çıkarak geliştirilen M2M platformunda, istemciler tarafından veriler JSON olarak servis katmanına gönderilmekte, ardından servis katmanı kedisine gelen istekleri JSON olarak karşılamakta ve yanıtları yine JSON olarak kullanıcıya sunmaktadır. Ayrıca veri tabanında veriler JSON dokümanı olarak saklanmaktadır. Böylelikle herhangi bir aşamada veri formatı değiştirilmemiş ve dönüştürme işlemleri ile zaman kaybedilmemiştir. Bu kullanım tarzı maliyeti azaltmakta, performansı ve ölçeklenebilirliği arttırmaktadır.

Bu tez çalışması yeni çalışma alanlarını da beraberinde getirmektedir. Daha sonra yapılacak çalışmalarda M2M platformu geliştirilirken çalışılan her yöntem ve her metot farklı çalışma alanlarını belirttiği için ayrı ayrı çalışma alanı olarak ele alınabilecektir. Ayrıca geliştirilen M2M platformu özel alanlara uygulanarak farklı kullanım yerleri için çalışmalar yapılacaktır. Aynı zamanda da geliştirilen her M2M platformunun güvenliğinin sağlanması, başlı başına bir çalışma alanı olarak karşımıza çıkmaktadır.

KAYNAKLAR

1. Bilgi Teknolojileri ve İletişim Kurumu. (2013). *Makineler Arası İletişim(M2M)*. Ankara: Bilgi Teknolojileri ve İletişim Kurumu, 3-4.

2. İnternet: METU MEMS. Microbolometer Type Uncooled Infrared Detectors in Standard CMOS Technology. *METU MEMS.* URL: http://www.mems.metu.edu.tr/index.php?option=com_content&view=article&id=125 &Itemid=104&lang=en, Son Erişim Tarihi: 20.11.2014.

3. İnternet: GIZMAG. FLIR Turns the Heat Up on Smartphone Thermal Imaging. *GIZMAG.* URL: http://www.gizmag.com/flir-iphone-thermal-imaging/30447/ , Son Erişim Tarihi: 20.11.2014.

4. Tosunoğlu, O. (2009). *Akıllı Ev Sistemlerinde Merkezi Veri Toplama ve Cihaz Yönetimi*, Yüksek Lisans Tezi, Gebze İleri teknoloji Enstitüsü, Gebze.

5. Zhang, S. K., Zhang, J. W., & Li, W. (2010). Design of M2M Platform Based on J2EE and SOA. In E-Business and E-Government (ICEE), *2010 International Conference on (pp. 2029-2032). IEEE.*

6. Kim, E. J., & Youm, S. (2013). Machine-to-machine platform architecture for horizontal service integration. *EURASIP Journal on Wireless Communications and Networking*, 2013(1), 1-9.

7. Soliman, M., Abiodun, T., Hamouda, T., Zhou, J., & Lung, C. H. (2013). Smart home: Integrating internet of things with web services and cloud computing. In Cloud Computing Technology and Science (CloudCom), *2013 IEEE 5th International Conference on (Vol. 2, pp. 317-320).* IEEE.

8. Akın, A. (2014). *Akıllı Ev Kavramı ve Otomasyon Sistemleri*, Yüksek Lisans Tezi, Haliç Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

9. Algoiare, O, T. (2014). *GSM Şebekesi Kullanılarak Akıllı Ev Dizaynı ve Uygulaması*, Yüksek Lisans Tezi, Çankaya Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

10. Lee, S., Jo, J., Kim, Y., & Stephen, H. (2014). A Framework for Environmental Monitoring with Arduino-Based Sensors Using Restful Web Service. In Services Computing (SCC), *2014 IEEE International Conference on(pp. 275-282).* IEEE.

11. Uluslararası Yatırımcılar Derneği. (2012). *2023 Hedefleri yolunda Bilgi ve İletişim Teknolojileri*. İstanbul: Uluslararası Yatırımcılar Derneği, 18-24.

12. Bilgi Teknolojileri ve İletişim Kurumu. (2013). *Makineler Arası İletişim(M2M)*. Ankara: Bilgi Teknolojileri ve İletişim Kurumu, 5-30.

13. İnternet: Verifone. Taşınabilir Yazarkasa. *Verifone.* URL: http://www.verifone.com.tr/products/hardware/mobile/, Son Erişim Tarihi: 16.05.2015.

14. İnternet: Sprint. (2014). Sprint Mobile Advertising and Reporting & Analytics Programs. *Sprint*. URL: http://newsroom.sprint.com/consumer-resources/sprint-mobile-advertising-and-reporting-analytics-programs.htm, Son Erişim Tarihi: 16.05.2015.

15. European Telecommunications Standards Institute. Technical Report. (2012). Document Number; *ETSI TR 102 935 V2.1.1*. Sophia-Antipolis: European Telecommunications Standards Institute.

16. Bilgi Teknolojileri ve İletişim Kurumu. (2014). *Uluslararası Elektronik Haberleşme Sektöründe Gelişmeler Bülteni*. Ankara: Bilgi Teknolojileri ve İletişim Kurumu Sektörel Araştırma ve Strateji Geliştirme Dairesi Başkanlığı, 3-22.

17. İnternet: OneM2M. M2M İçin Dünyanın İlk Küresel Standartları. (2015). *OneM2M*. URL: http://www.onem2m.org/news-events/news/53-the-rise-of-the-achines-world-s-first-global-standards-for-m2m-eployment, Son Erişim Tarihi: 16.05.2015.

18. OneM2M. Technical Report. (2013). Document Number; *OneM2M-TR-0002-Architecture_Analysis_Part_1-V-0.2.0*. OneM2M.

19. OneM2M. Technical Report. (2013). Document Number; *oneM2M-TR-0001-UseCase*. OneM2M.

20. International Data Corporation. (2013). *Worldwide Internet of Things (IoT) 2013-2020 Forecast: Billions of Things trillions Of Dollars*. Belge No. 243661.

21. Deloitte Türkiye. (2011). *Elektronik haberleşme ve eğilimler 2011,* Teknoloji, Medya ve Telekomünikasyon Endüstrisi. Deloitte Türkiye, 3-15.

22. Gökhan F. (2014). İnternet ve her şey: Güvenlik, gençlik ve spektrum. *Türkiye Bilişim Dergisi*, 106-109.

23. İnternet: Deloitte Türkiye. 2015 Teknoloji Medya Telekomünikasyon Öngörüleri Raporu. *Deloitte Türkiye*. URL: http://www2.deloitte.com/tr/tr/pages/about-deloitte/articles/teknoloji-tuketiciden-isletmelere-dogru-el-degistiriyor.html, Son Erişim Tarihi: 16.05.2015.

24. Yishan L., Sathiamoorthy M., (2013). A performance comparison of SQL and NoSQL databases, Communications, *Computers and Signal Processing (PACRIM),* 2013 IEEE Pacific Rim, Conference on, 27-29 .

25. Lior O., Nurit G., Yaron G., Ehud G., Jenny A., (2011). Security Issues in NoSQL Databases, Trust, Security and Privacy in Computing and Communications (TrustCom), *2011 IEEE 10th International Conference on*, 16-18 Nov. (2011).

26. Jan Sipke V., Bram W., Robert J. M., (2012). Sensor Data Storage Performance: SQL or NoSQL, Physical or Virtual, *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, 24-29 June (2012).

27. Han J., Haihong E., Le G., Du J., (2011). Survey on NoSQL database, Pervasive Computingand Applications (ICPCA), *2011 6th International Conference on*, 26-28 Oct. (2011).

28. İnternet: Kahraman, Ö. (2003). Web Servisleri ve Kayıtçılar (UDDI & RDF). *Csharpnedir*. URL: HTTP://www.csharpnedir.com/articles/read/?id=106, Son Erişim Tarihi: 16.05.2015.

29. Çiçek, A, N. (2009). *Restful Web Servisleri İle E-Sağlık Sistemleri Gerçekleştirimi,* Yüksek Lisans Tezi, TOBB Ekonomi ve Teknoloji Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

30. Kaya, Y. Ç., Kaya, M., Yıldırım, S. Ö. (2014) Lokasyon Tabanlı Mobil Kampus Uygulaması ve Kullanılabilirlik Değerlendirmesi. *VIII. Ulusal Yazılım Mühendisliği Sempozyumu*, KKTC.

31. Özkaya, Ö. (2013*). Restful Web Servisleri Üzerine Bir İnceleme,* Yüksek Lisans Tezi, Ege Üniversitesi Fen Bilimleri Enstitüsü, İzmir.

32. Yüksek, Y, (2009). Anlamsal Web Servisleri Temelinde Örnek Bir Servis Tanımı, *Akademik Bilişim 2009*, Harran Üniversitesi, Şanlıurfa.

33. Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures,* Doctoral Dissertation, University of California, Irvine.

34. Alpay, O., Hatem, V., Sunel, S. (2011). JMX için Restful Adaptör. *5. Ulusal Yazılım Mühendisliği Sempozyumu*. ODTÜ, Ankara.

35. İnternet: Akıllı Yazılım. (2013). *Neden Spring, Akıllı Yazılım*. URL: http://www.akilliyazilim.org/spring-framework-dersleri/spring-nedir-neden-spring.html, Son Erişim Tarihi: 16.05.2015.

36. İnternet: Spring. Spring Freamework. *Spring*. URL: http://projects.spring.io/spring-framework/, Son Erişim Tarihi: 16.05.2015.

37. İnternet: Programmingfree. Spring MVC 4.0 RESTFUL web Services Simple Example. *Programmingfree*. URL: http://www.programming-free.com/2014/01/spring-mvc-40-restful-web-services.html, Son Erişim Tarihi: 16.05.2015.

38. İnternet: Spring. Understanding REST. *Spring*. URL: http://spring.io/understanding/rest, Son Erişim Tarihi: 16.05.2015.

EKLER

EK-1. Jeton bilgilerinin üretilmesi

```
@Override
	public LoginAccountResultBean loginAccount(LoginAccountRequestBean
requestBean) {
		LoginAccountResultBean validAccount = new LoginAccountResultBean();
		String userName = requestBean.getUserName();
		String password = requestBean.getPassword();
		Account account = accountDBService.login(userName, password);

		if (account == null) {
			validAccount.setIsAuth(false);
			return validAccount;
		}
		Token objectToSave=new Token();
		objectToSave.setCreatedDate(new Date());
		Calendar expire= Calendar.getInstance();
		expire.add(Calendar.HOUR, 1);
		objectToSave.setExpireDate(expire.getTime());
		objectToSave.setToken(UUID.randomUUID().toString());
		objectToSave.setUserID(account.getId());
		tokenDBService.save(objectToSave);
		validAccount.setIsAuth(true);
		validAccount.setToken(objectToSave.getToken());
		validAccount.setUserId(objectToSave.getUserID());

		return validAccount;
	}
```

EK-1.(devam) Jeton bilgilerinin üretilmesi

Jeton bilgilerinin kontrolü (cihaz ekleme işlemi için örnek kod)

```
@RequestMapping(value = "/CihazEkle", method = RequestMethod.POST, produces = "application/json", consumes = "application/json")
	@ResponseBody
	public CihazEkleAPIResponse login(@RequestBody CihazEkleAPIRequest apiRequest, ModelMap model) {

		CihazEkleAPIResponse apiResponse=new CihazEkleAPIResponse();

		Boolean checkAuth = tokenDBService.checkAuth(apiRequest.getToken(), apiRequest.getUserId());

		if (checkAuth == false) {
			apiResponse.setExceptionCode(101);
			apiResponse.setHasException(true);
			apiResponse.setValidatormessage("Auth Error");
			return apiResponse;
		}
		CihazEkleRequestBean requestBean = new CihazEkleRequestBean();

requestBean.setCihazAdi(apiRequest.getCihazAdi());
		requestBean.setUserId(apiRequest.getUserId());
		CihazEkleResultBean CihazEkleResBean = cihazBusinessService.cihazEkle(requestBean);
		apiResponse.setCihazId(CihazEkleResBean.getCihazId());
		apiResponse.setCihazAdi(apiRequest.getCihazAdi());
		apiResponse.setUserId(apiRequest.getUserId());
		return apiResponse;
	}
```

EK-2. Servis dokümanı

Çizelge EK-2.1. Kullanıcı kaydı

Servis Adres	http://localhost:8080/saadin/service/account/save

İstek Formatı	{
	"username" : "deger",
	"fulname" : " deger ",
	"password" : " deger ",
	"eposta" : " deger "
	}

Örnek İstek	{
	"username" : "saadin",
	"fulname" : "saadin oyucu",
	"password" : "123456",
	"eposta" : "saadinoyucu@gazi.edu.tr"
	}

Cevap Formatı	{
	exceptionCode: 0
	hasException: false
	validatormessage: null
	}

Örnek Cevap	{
	exceptionCode: 0
	hasException: false
	validatormessage: null
	}

Servis Açıklama	Cevap olarak gelen hata detayları false ve null ise sorun yok demektir.

EK-2.(devam) Servis dokümanı

Çizelge EK-2.2. Kullanıcı girişi

Servis Adres	http://localhost:8080/saadin/service/account/login

İstek Formatı	{
	"username" : "deger",
	"password" : "deger"
	}

Örnek İstek	{
	"username" : "saadin",
	"password" : "123456"
	}

Cevap Formatı	{
	exceptionCode: 0
	hasException: false
	token: "deger"
	userId: "deger"
	validatormessage: null
	}

Örnek Cevap	{
	exceptionCode: 0
	hasException: false
	token: "3e500d06-6c53-444a-8e81-18abc99fb34f"
	userId: "5507d0b8510dccbfbb9b9fc0"
	validatormessage: null
	}

Servis Açıklama	Cevap olarak gelen hata detayları false ve null ise sorun yok demektir. Token sistemde login kalma süresinin de belirtildiği değişken olup aynı zamanda da platform üzerinden başka işlem yapılırken her seferinde kontrol edilmektedir. UserID is kullanıcının ID bilgisidir.

EK-2.(devam) Servis dokümanı

Çizelge EK-2.3. Cihaz kaydı (cihaz ekle)

Servis Adres	http://localhost:8080/saadin/service/cihaz/CihazEkle

İstek Formatı	{
	"cihazAdi" : "deger",
	"token" : "deger",
	"userId" : "deger"
	
	}

Örnek İstek	{
	"cihazAdi" : "arduino 1",
	"token" : "3e500d06-6c53-444a-8e81-18abc99fb34f",
	"userId" : "5507d0b8510dccbfbb9b9fc0"
	}

Cevap Formatı	{
	exceptionCode: 0
	hasException: false
	cihazId: "deger"
	cihazAdi: "deger"
	userId: "deger"
	validatormessage: null
	}

Örnek Cevap	{
	exceptionCode: 0
	hasException: false
	cihazId: "792a1d92-4924-4644-9e39-8ee2fccdf9d4"
	cihazAdi: "arduino 1"
	userId: "5507d0b8510dccbfbb9b9fc0"
	validatormessage: null
	}

Servis Açıklama	Cevap olarak cihazId, cihazAdi ve userId kullanıcıya verilecektir ve kullanıcı aynı sayfada bulunan arduino kodunu indirecektir. Burada verilen bilgiler arduino kodu içerisinde gerekli yerlere yazılacaktır.

EK-2.(devam) Servis dokümanı

Çizelge EK-2.4. Cihazlarım (cihaz listele)

Servis Adres	http://localhost:8080/saadin/service/cihaz/cihazlarim

İstek Formatı	{
	"token" : "deger",
	"userId" : "deger"
	}

Örnek İstek	{
	"token" : "3e500d06-6c53-444a-8e81-18abc99fb34f",
	"userId" : "5507d0b8510dccbfbb9b9fc0"
	}

Cevap Formatı	exceptionCode: 0
	hasException: false
	-cihazlarim:[3]
	-0:
	{
	cihazID: "deger"
	cihazAdi: "deger"
	}

Örnek Cevap	exceptionCode: 0
	hasException: false
	-cihazlarim:[3]
	-0:
	{
	cihazID: "da3c4d69-99d9-42eb-9e22-7e9bcdf79067"
	cihazAdi: "arduino 1"
	}

Servis Açıklama	userId ile sorgulanan cihazlara Cevap olarak cihazID ve cihazAdi gelmektedir. Liste halinde alınan bilgiler örnekteki gibidir.

EK-2.(devam) Servis dokümanı

Çizelge EK-2.5. Cihaz bilgilerini listele (cihaz liste)

Servis Adres	http://localhost:8080/saadin/service/cihaz/list

İstek Formatı	{
	"cihazAdi" : "deger",
	"token" : "deger",
	"userId" : "deger"
	}

Örnek İstek	{
	"cihazAdi" : "arduino okul",
	"token" : "80719408-8725-4ffc-8ade-d5af2d75d2b1",
	"userId" : "5507d0b8510dccbfbb9b9fc0"
	}

Cevap Formatı	{
	exceptionCode: 0
	hasException: false
	cihazlar:
	[1054]
	-0:
	{
	kullaniciID: "deger"
	cihazID: " deger "
	cihaz_adi: " deger "
	sicaklik: " deger "
	nem: " deger "
	createdDate: 1427109040290
	}

Örnek Cevap	{
	exceptionCode: 0
	hasException: false
	cihazlar:
	[1054]
	-0:
	{
	kullaniciID: "5507d0b8510dccbfbb9b9fc0"
	cihazID: "c31dda9c3-9e0f-4c56-a376-7c991bebeab6"
	cihaz_adi: "arduino okul"
	sicaklik: "22.00"
	nem: "35.00"
	createdDate: 1427109040290
	}

Servis Açıklama	cihazAdi ile sorgulanan cihaza Cevap olarak kullaniciID , cihazID , cihaz_adi, sicaklik, nem ve createdDate bilgileri cevap olarak gönderilmektedir.

EK-2.(devam) Servis dokümanı

Çizelge EK-2.6. Cihaz bilgilerini gönder (arduino kart için hazırlanan servis)

Servis Adres	http://localhost:8080/saadin/service/cihaz/save

İstek Formatı	{
	"cihazID" : "deger",
	"sicaklik" : "deger",
	"nem" : "deger",
	"cihaz_adi" : "deger",
	"userId": "deger"
	}

Örnek İstek	{
	"cihazID" : "123456789",
	"sicaklik" : "32",
	"nem" : "44",
	"cihaz_adi" : "arduino 1",
	"userId": "5507d0b8510dccbfbb9b9fc0"
	}

Cevap Formatı	{
	exceptionCode: 0
	hasException: false
	validatormessage: null
	}

Örnek Cevap	{
	exceptionCode: 0
	hasException: false
	validatormessage: null
	}

Servis Açıklama	Bu servis Arduino kart için tasarlanmıştır. Arduino için geliştirilen API sayesinde sıcaklık ve nem bilgileri sürekli olarak veri tabanına kaydedilecektir.

EK-3. M2M cihazlar için örnek API kodları (LM35 algılayıcısı için)

```
#include <SPI.h>
#include <Ethernet.h>

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
char server[] = "10.12.86.249";
IPAddress ip(10,12,83,177);
EthernetClient client;

//kullanıcı ayarları
String kullaniciID = "\"kullanıcıID\"";
String cihazID = "\"cihazinID\"";
String cihaz_adi = "\"cihazinAdi\"";

const int analogInPin = A0;
int sensorValue = 0;
float reading = 0;
float tempC;

void setup() {
   analogReference(INTERNAL);
  Serial.begin(9600);
   while (!Serial) {
    ;
  }
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    Ethernet.begin(mac, ip);
  }
}
int *i=0;
char *sum;
void loop()
{
          reading = analogRead(analogInPin);
          tempC = reading / 9.31;

  Serial.println("connecting...");

  if (client.connect(server, 8080)) {
    Serial.println("connected");
  String body = "{\"userId\":" + kullaniciID + ", \"sicaklik\":" + tempC + ", \"cihazID\":" +
cihazID + ", \"cihaz_adi\":" + cihaz_adi + "}";
    client.print("POST /saadin/service/cihaz/save HTTP/1.0\n");
    client.print("Host: 10.12.86.249\n");
    client.print("Content-length: ");
    client.println(body.length());
    client.print("Content-Type: application/json\n");
    client.println("Connection: close\n");
```

EK-3.(devam) M2M cihazlar için örnek API kodları (LM35 algılayıcısı için)

```
    client.println(body);
    Serial.println("Post made");
    delay(1000);
  }
  else {
    Serial.println("connection failed");
  }

  while (client.available()) {
    char c = client.read();
    Serial.print(c);
  }

  if (!client.connected()) {
    Serial.println();
    Serial.println("disconnecting.");
    client.stop();
  }
     delay(2000);
}
```

EK-4. İstemci web uygulaması yönlendirme kodları (giriş ve cihaz ekleme işlemleri için)

```
<?php
function getLogin() {
	$app = getSlimInstance();
	$app->render('login.php', array('link' => $app->urlFor('yeni-kayit')));
}
function postLogin() {
	$app = getSlimInstance();
	$username = $app->request->post('username');
	$requestArr['username'] = $username;
	$requestArr['password'] = $app->request->post('password');
	$url = $app->serviceUrl . 'account/login';
	$responseArray = sendPostRequest($url, $requestArr);
	if ($responseArray['hasException'] == false && $responseArray['token'] != null) {
		$_SESSION['login'] = true;
		$_SESSION['token'] = $responseArray['token'];
		$_SESSION['userId'] = $responseArray['userId'];
		$_SESSION['username'] = $username;
		$app->response->redirect($app->urlFor('home'));
	} else {
		$app->flash('error', 'Kullanici Ismi/Parola Hatali');
		$app->flash('username', $requestArr['username']);
		$app->response->redirect($app->urlFor('login'));
	}
}
function getCihazEkle() {
	checkAuth();
	$app = getSlimInstance();
	$app->render('cihazEkle.php');
}
function postCihazEkle() {
	checkAuth();
	$app = getSlimInstance();
	$requestArr['cihazAdi'] = $app->request->post('cihazAdi');
	$requestArr['token'] = $_SESSION['token'];
	$requestArr['userId'] = $_SESSION['userId'];
	$url = $app->serviceUrl . 'cihaz/CihazEkle';
	$responseArray = sendPostRequest($url, $requestArr);

	if ($responseArray['hasException'] == false &&
$responseArray['validatormessage'] == null) {
		$app->flash('status', 'Cihaz Basariyla Eklendi!');
		$app->response->redirect($app->urlFor('cihaz-ekle'));
	} else {
		$app->flash('error', $responseArray['validatormessage']);
		$app->response->redirect($app->urlFor('cihaz-ekle'));
	}
}?>
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: OYUCU, Saadin
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 04.01.1988, Nizip
Medeni hali	: Evli
Telefon	: 0 (312) 202 85 59
Faks	: 0 (312) 202 89 47
e-mail	: saadinoyucu@gazi.edu.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek lisans	Gazi Üniversitesi /Bil. Müh. (Tek. Fak.)	Devam Ediyor
Lisans	Gazi Üniversitesi/ Bil. Sist. Öğrt.	2012

İş Deneyimi

Yıl	Yer	Görev
2014-Halen	Gazi Üniversitesi	Araştırma Görevlisi
2012-2014	LST Yazılım A.Ş.	Front End Developer – Yazılım Mühendisi

Yabancı Dil

İngilizce

Yayınlar

Oyucu, S., Polat, H. (2015). M2M Sistemlerde SQL Veya NOSQL Kullanımı. *Akademik Bilişim Konferansı*. Eskişehir.

Hobiler

Futbol, Yüzme, Kültür Gezileri

GAZİ GELECEKTİR...