FIWARE IOT PLATFORMU VE GRAFANA KULLANILARAK VERİ KALICILIĞI VE GÖRSELLEŞTİRME

2024
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ

Augusto GOMES JUNIOR

Tez Danışmanı
Dr. Öğr. Üyesi Omar DAKKAK

FIWARE IOT PLATFORMU VE GRAFANA KULLANILARAK VERİ KALICILIĞI VE GÖRSELLEŞTİRME

Augusto GOMES JUNIOR

Tez Danışmanı
Dr. Öğr. Üyesi Omar DAKKAK

T.C.
Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalında
Yüksek Lisans Tezi
Olarak Hazırlanmıştır

KARABÜK
Haziran 2024

Augusto GOMES JUNIOR tarafından hazırlanan "FIWARE IoT PLATFORMU VE GRAFANA KULLANILARAK VERİ KALICILIĞI VE GÖRSELLEŞTİRME" başlıklı bu tezin Yüksek Lisans Tezi olarak uygun olduğunu onaylarım.

Dr. Öğr. Üyesi Omar DAKKAK .....
Tez Danışmanı, Bilgisayar Mühendisliği Anabilim Dalı

Bu çalışma, jürimiz tarafından Oy Birliği ile Bilgisayar Mühendisliği Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir. 13/06/2024

Ünvanı, Adı SOYADI (Kurumu) İmzası

Başkan : Dr. Öğr. Üyesi Oğuzhan MENEMENCİOĞLU (KBÜ) .....

Üye : Dr. Öğr. Üyesi Yusuf Yargı BAYDİLLİ (HÜ) .....

Üye : Dr. Öğr. Üyesi Omar DAKKAK (KBÜ) .....

KBÜ Lisansüstü Eğitim Enstitüsü Yönetim Kurulu, bu tez ile, Yüksek Lisans derecesini onamıştır.

Doç. Dr. Zeynep ÖZCAN .....
Lisansüstü Eğitim Enstitüsü Müdürü

"Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim."

Augusto GOMES JUNIOR

ÖZET

Yüksek Lisans Tezi

FIWARE IOT PLATFORMU VE GRAFANA KULLANILARAK VERİ KALICILIĞI VE GÖRSELLEŞTİRME

Augusto GOMES JUNIOR

Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr. Üyesi Omar DAKKAK
Haziran 2024, 88 sayfa

Nesnelerin interneti (IoT), çeşitli cihazların birbiriyle iletişim kurarak ortak hedeflere ulaşmasını ve iş birliği yapmasını sağlayan bir paradigmadır. IoT ortamları, farklı cihazlar ve ağ protokolleri arasında büyük çeşitlilik gösterir. Bu çeşitliliği ele almak için, çeşitli IoT ara yazılım platformları önerilmiştir. Bu platformlar uygulamalar ve son kullanıcılar için cihazların özelliklerini soyutlayarak ve bunların uyumlu çalışmasını teşvik ederek işlev görmektedir. Ancak, çoğu bu tür platformlar kapalı kaynaklıdır ve lisanslama, destek ve bakım gibi gereklilikler yüksek maliyetlere neden olabilmektedir. Bu çalışmada IoT kavramı kullanılarak sensörler, düşük maliyetli bir cihaz, açık kaynak bir platformu, veri tabanları ve açık kaynak veri görselleştirme aracıyla birlikte verilerin algılanmasından görselleştirilmesine kadar olan süreci kapsayan yeni bir yaklaşım önerilmektedir. Çalışmanın geliştirilmesi için, DHT22 sensöründen elde edilen verilerin Orion Context Broker'a iletilmesi amacıyla Raspberry Pi kullanılmıştır. Üç adet FIWARE bileşeni kullanılmıştır: Orion Context

Broker, IoT Ajanı ve Cygnus. Orion Context Broker ve IoT Ajanı, verilerin kaydetmek için MongoDB teknolojisini kullanırken, Cygnus ise Orion'nun geçmiş bağlam verilerini MySQL veri tabanlarında kalıcı hale getirir. SQL veri tabanlarındaki verileri görselleştirmek için Grafana bileşeni kullanıldı. Önerilen sistem, açık kaynaklı bir IoT ara yazılım platformunda veri kalıcılığı ve görselleştirme görevlerini kolaylaştırarak, düşük maliyetli teknolojileri kullanarak zamandan ve kaynaklardan tasarruf etmeyi sağlayan avantajlar sunmaktadır.

Anahtar Sözcükler : Nesnelerin İnterneti (IoT), FIWARE, Raspberry Pi, Cygnus, MySQL, Grafana, Docker.

Bilim Kodu : 92429

ABSTRACT

Master Thesis

DATA PERSISTENCE AND VISUALIZATION USING FIWARE IOT PLATFORM AND GRAFANA

Augusto GOMES JUNIOR

Karabük University
Institute of Graduate Programs
Department of Computer Engineering

Thesis Advisor:
Assist. Prof. Dr. Omar DAKKAK
June 2024, 88 pages

The Internet of Things (IoT) is a paradigm that enables various devices to communicate with each other and collaborate to achieve common goals. IoT environments exhibit a wide diversity of devices and network protocols. To address this diversity, various IoT middleware platforms have been proposed. These platforms function by abstracting the capabilities of devices and promoting their interoperability, aiming to serve applications and end-users. However, most of these platforms are closed source, leading to high costs associated with licensing, support, and maintenance. In this study, a new approach is proposed that encompasses the process from data sensing to visualization using the concept of IoT. The development of the study involves the utilization of sensors, a low-cost device, an open-source platform, databases, and an open-source data visualization tool. Raspberry Pi is used to transmit data from the DHT22 sensor to the Orion Context Broker. Three FIWARE components are employed: the Orion Context Broker, IoT Agent, and Cygnus. While the Orion

Context Broker and IoT Agent utilize MongoDB for data persistence, Cygnus makes the historical context data from Orion persistent in MySQL databases. Grafana component is used to visualize data stored in SQL databases. The proposed system facilitates data persistence and visualization tasks in an open-source IoT middleware platform, utilizing low-cost technologies to save time and resources.

Key Word **:** Internet of Things **(**IoT), FIWARE, Raspberry Pi, Cygnus, MySQL, Grafana, Docker.

Science Code **:** 92429

TEŞEKKÜR

Babam Augusto GOMES’nin anısına, annem Segunda CAMILO SANCA'ya ve kardeşlerime tüm kalbimle teşekkür ederim. Küçük yaşlardan itibaren bana başarıların adanmışlık ve sebatla gerçekleştiğini gösterdiler.

Danışmanım Dr. Öğr. Üyesi Omar DAKKAK'a, sunduğu birçok fırsat, bana duyduğu güven ve önemli önerileri için teşekkür ederim.

Dr. Öğr. Üyesi İsa AVCI'ya ve Öğr. Gör. Bahadır Furkan KINACI’ya bu çalışmanın gerçekleşmesindeki desteği için teşekkür ederim.

Ayrıca bu üniversiteye de bilgiyi destekleyen ve tüm fikirlerin hoş karşılandığı bir yer olduğu için teşekkür etmeliyim.

Tüm aileme ve arkadaşlarıma destekleri ve iş birlikleri için teşekkür ederim.

İÇİNDEKİLER

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

KISALTMALAR DİZİNİ

KISALTMALAR

API : Application Programming Interface (Uygulama Programlama Arayüzü)
BSON : Binary JSON (İkili JSON)
CEP : Complex Event Processing (Karmaşık Olay İşleme)
CKAN : Comprehensive Knowledge Archive Network (Kapsamlı Bilgi Arşiv Ağı)
CoAP : Constrained Application Protocol (Kısıtlı Uygulama Protokolü)
CSI : Camera Serial Interface (Kamera Seri Arayüzü)
CSV : Comma Separated Values (Virgülle Ayrılmış Değerler)
DAQ : Data Acquisition (Veri Toplama)
DHT : Digital Humidity and Temperature (Dijital Nem ve Sıcaklık)
DoS : Denial of Service (Hizmet Reddi)
FI : Future Internet (Geleceğin İnterneti)
FI-PPP : Future Internet Public-Private Partnership (Geleceğin İnterneti Kamu-Özel Ortaklığı)
GE : Generic Enabler (Jenerik Etkinleştirici)
GIS : Geographic Information System (Coğrafi Bilgi Sistemi)
GPIO : General Purpose Input/Output (Genel Amaçlı Giriş/Çıkış)
GPS : Global Positioning System (Küresel Konumlandırma Sistemi)
HDMI : High-Definition Multimedia Interface (Yüksek Tanımlı Multimedya Arayüzü)
HTML : Hypertext Markup Language (Hiper Metin İşaretleme Dili)
HTTP : Hyper-Text Transfer Protocol (Hiper Metin Aktarım Protokolü)
I2C : Inter-Integrated Circuit (Entegre Devreler Arası)
I2ND : Interface to Networks and Devices (Ağlara ve Cihazlara Arayüz)
IaaS : Infrastructure as a Service (Hizmet Olarak Altyapı)
IB : IoT Backend (IoT Arka Uç)
IE : IoT Edge (IoT Uç)

IoE : Internet of Energy (Enerjinin İnterneti)

IoT : Internet of Things (Nesnelerin İnterneti)

IP : Internet Protocol (İnternet Protokolü)

IPsec : Internet Protocol Security (İnternet Protokolü Güvenliği)

JSON : JavaScript Object Notation (JavaScript Nesne Notasyonu)

KP : KnowLedge Processor (Bilgi İşlemcisi)

M2M : Machine to Machine (Makineden Makineye)

MQTT : Message Queuing Telemetry Transport (Mesaj Kuyruklama Telemetri Aktarımı)

NFC : Near Field Communication (Yakın Alan İletişimi)

NGSI : Next Generation Service Interfaces (Sonraki Nesil Servis Arayüzleri)

OASC : Open & Agile Smart Cities (Açık ve Çevik Akıllı Şehirler)

OS : Operating System (İşletim Sistemi)

P2ABC : Privacy-Preserving Attribute-Based Credentials (Gizliliği Koruyan Öznitelik Tabanlı Kimlik Bilgileri)

PaaS : Platform as a Service (Hizmet Olarak Platform)

PEP : Policy Enforcement Point (Politika Uygulama Noktası)

POI : Point of Interest (İlgi Çekici Nokta)

QoS : Quality of Services (Hizmet Kalitesi)

REST : Representational State Transfer (Temsili Durum Transferi)

RFID : Radio Frequency Identification (Radyo Frekansı ile Tanımlama)

ROS : Robotics Operating System (Robotik İşletim Sistemi)

SaaS : Software as a Service (Hizmet Olarak Yazılım)

SD : Secure Digital (Güvenli Dijital)

SDN : Software Defined Networking (Yazılım Tanımlı Ağ)

SIB : Semantic Information Broker (Semantik Bilgi Aracısı)

SOFIA : Smart Objects for Intelligent Applications (Akıllı Uygulamalar için Akıllı Nesneler)

SPI : Serial Peripheral Interface (Seri Çevresel Arayüz)

SSAP : Smart Space Access Protocol (Akıllı Alan Erişim Protokolü)

SSH : Secure Shell (Güvenli Kabuk)

TLS : Transport Layer Security (Taşıma Katmanı Güvenliği)

UART : Universal Asynchronous Receiver-Transmitter (Evrensel Asenkron Alıcı-

Verici)

URL : Uniform Resource Locators (Tekdüzen Kaynak Konum Belirleyicileri)

USB : Universal Serial Bus (Evrensel Seri Veriyolu)

VM : Virtual Machine (Sanal Makine)

XACML: eXtensible Access Control Markup Language (Genişletilebilir Erişim Kontrolü İşaretleme Dili)

XML : eXtensible Markup Language (Genişletilebilir İşaretleme Dili)

WSN : Wireless Sensor Network (Kablosuz Sensör Ağları)

WWW : World Wide Web (Dünya Çapında Ağ)

BÖLÜM 1

GİRİŞ

Nesnelerin İnterneti (IoT), farklı cihazların birbirleriyle etkileşime girerek ortak hedeflere ulaşmasını sağlayan bir paradigma olarak tanımlanmaktadır. Bu kavram, üç farklı vizyon altında sınıflandırılabilir: internet odaklı, genel nesne odaklı ve semantik odaklıdır.[1].

Veri miktarı her yıl artmakta olup, bu veriler elektronik cihazlar ve çeşitli kaynaklardan gelen sensörler aracılığıyla sağlanmaktadır; bu sensörler arasında hareket, ışık, sıcaklık ve nem gibi çeşitli ölçüm kaynakları bulunmaktadır. Bir karşılaştırmalı çalışma [2], dünya genelinde bağlı akıllı cihazların sayısının insan nüfusundaki artış oranından daha hızlı bir şekilde arttığını göstermektedir [3]. IoT, diğer faydalarının yanı sıra, cihazlardan ve çalışma koşullarından gerçek zamanlı olarak veri toplanmasını sağlar [4]. Bu verilerle ev içi görevleri otomatikleştirilebilir, şirketlerde karar verme süreci iyileştirilebilir ve IoT ağlarının bulunduğu ortamda gerçekleşen süreçleri anlamak mümkün olabilmektedir.

IoT'nin temel fikri, farklı tiplerdeki homojen ve heterojen yapıdaki elektronik cihazları birbirine bağlamak ve böylece verimli bir şekilde bilgi iletmelerini sağlamaktır ve bu cihazlar arasında örtüşen iletim aralıkları bulunabilir [5]. Veri kalıcılığı, verilerin güvenilir bir şekilde iletilmesinde kritik bir konudur. Veri kalıcılığı çözümleri, verilerin değişmeden kalmasını ve erişilebilir olmasını temin ederek güvenilir veri iletimini sağlamaya yardımcı olabilecektir [6].

Veri, genellikle kilobayt, megabayt, gigabayt ve hatta terabayt olarak ölçülür ve bu miktarlar, ortalama bir insanın günlük karşılaştığı veri miktarını temsil etmektedir. Ancak, gelecek birkaç yıl içinde, bu geleneksel birimler daha az önemli hale gelebilir. Dolayısıyla, verileri analiz etmek, keşfetmek ve görsel olarak temsil etmek için güçlü

ve etkili bir veri görselleştirme aracına sahip olmak son derece önemlidir [7]. Grafana, metrikleri görselleştirmeye, sorgulamaya, uyarlamaya ve keşfetmeye yardımcı olan ücretsiz ve açık kaynaklı bir veri görselleştirme ve analiz aracıdır [8].

Donanım ve yazılım teknolojilerinin doğal çeşitliliğinden kaynaklanan bu dinamik ve oldukça heterojen bağlam, akıllı şehirler [9] için yazılım geliştirmenin çeşitli zorluklarla karşı karşıya olduğunu göstermektedir [10]. Bu zorluklar arasında cihaz heterojenliği, çeşitli kaynaklardan gelen bilgilerin toplanması ve analiz edilmesi gibi büyük veri yönetimi, ölçeklenebilirlik, gizlilik politikaları gibi konular yer almaktadır [11]. Bu nedenle, ara yazılım platformları, uygulama geliştirmeyi kolaylaştırmak, cihazların, insanların, sistemlerin ve verilerin entegrasyonunu sağlamak için birlikte çalışabilirlik ve uygulamalar için gerekli bir dizi ek hizmet sağlamak için umut verici çözümler olarak ortaya çıkmıştır [12].

Bir ara yazılım platformu, cihazlar ve uygulamalar için soyutlamalar sağlar ve çeşitli düzeylerde şeffaflık ve birlikte çalışabilirlik sunmaktadır [13]. Ayrıca son kullanıcılar ve uygulamalar için çeşitli hizmetler sunmaktadır. Bu platformlar, uygulama geliştiricilerden altta yatan donanım, ağ protokolü katmanları, platformlar ve İşletim Sistemi (OS) bağımlılıklarıyla ilgili karmaşıklıkları ve heterojenlikleri gizler. Bunun yanı sıra, sistem kaynaklarının yönetimini kolaylaştırır ve uygulama yürütmenin öngörülebilirliğini artırmaktadır [14].

Temsili Durum Transferi (REST), veri göndermek için POST, güncellemek için PUT, istekte bulunmak için GET ve veri kaldırmak için DELETE gibi yöntemleri kullanmanıza izin veren bir dizi mimari kuraldır [15]. Bu çalışmada, Postman aracını kullanarak basit bir Uygulama Programlama Arayüzü (API) üzerinden kolayca etkileşime geçebileceğiniz bir web arayüzü üzerinden REST istemcisi test edilmiştir. Bu program, Orion Context Broker'a yapılan çeşitli istekleri simüle etmemizi sağlamaktadır.

Bu araştırmanın odak noktası, mevcut kaynakları ve düşük maliyetli teknolojileri incelemek ve bir IoT ara yazılım platformunda geçmiş verileri kalıcı hale getirip görselleştirmek için hangisinin uygun olduğunu belirlemektir. Ayrıca, verileri bir veri

tabanına kaydetmek yalnızca daha sonraki analizler için korumakla kalmaz, aynı zamanda tasarımının verileri daha karmaşık uygulamalara besleyebileceği anlamına da gelmektedir [16]. Bu çalışma, popüler bir açık kaynak görselleştirme aracı olan Grafana ile Geleceğin İnternet WARE (FIWARE)[17] ara yazılım platformunda verilerin kalıcı hale getirilmesi için düşük maliyetli bir IoT teknolojisi çözümü geliştirmeye yönelik yeni bir yaklaşım sunmaktadır.

1.1. MOTİVASYON

Yüksek lisans tez çalışması için bu konuyu seçmemin birçok nedeni bulunmaktadır. Bunlardan biri, IoT kavramının uygulanması ve teknolojilerin entegrasyonu için mevcut sınırlamaların ve teknolojik alternatiflerin neler olduğunu anlamak için ücretsiz ve açık kaynaklı bir ara yazılım platformu ile düşük maliyetli teknolojileri kullanarak bir sistem geliştirme ihtiyacıdır. Sistemin ana çerçevesi, veri algılamasından veri görselleştirmeye kadar olan bir süreci kapsaması amaçlanmaktadır.

Bir sistemin toplanan verileri bir görselleştirme aracında görüntüleyebilmesi oldukça önemlidir. Bu durum, verilerin kolayca anlaşılabilir ve merkezi bir şekilde sunulmasını sağlayacaktır. Veri miktarı her yıl artmakta ve bunların çoğu IoT cihazlarından gelmektedir [18]. IoT'nin zorluklarından biri, sensör verilerinin işlenmesi ve depolanması için standartların kullanılmasıdır [19]. Çok çeşitli sensörlerden ve/veya sensörle entegre edilmiş cihazlardan veri toplayan bir sistem, bu zorluklarla karşılaşacaktır: verilerin türünden bağımsız olarak aynı kalıcılık sisteminde saklanması gerekliliği, sensör verilerinin uzamsal ve zamansal ekseni değiştirebilir, sensörler ve cihazlar bir ağ geçidinden diğerine taşınabilmektedir [20].

Verilerin heterojenliği [21-26] IoT sistemlerinde birlikte çalışabilirliği zorlaştırdığından, bu durum uygulamalar için bir problem oluşturmaktadır [27]. Bundan dolayı, bir ara yazılım platformunda veya dışındaki bir ilişkisel veri tabanında gerçekleştirilebilen işlemlerin kalıcılığı, sorgulanması, indekslenmesi, işlenmesi ve manipülasyonu gibi yeni zorluklar ortaya çıkmıştır. Örneğin, Big Data [28, 29] ve Bulut Bilişim [30, 31], bu zorlukların bazılarını çözmek için potansiyel öneriler olarak

ortaya çıkmış ve muazzam hacimde çeşitli ve yapılandırılmamış verilerle başa çıkmayı mümkün kılmıştır.

Bulut bilişim paradigmasının hızla benimsenmesi, birçok bulut bilişim hizmet sağlayıcısının Hizmet Olarak Yazılım (SaaS) [32], Hizmet Olarak Altyapı (IaaS) ve Hizmet Olarak Platform (PaaS) gibi hizmetler sunmasıyla sonuçlanmıştır. Ancak, bu sağlayıcıların çoğu kapalı kaynaklı bir platform kullanmakta ve satıcıya bağlı çözümler sunmaktadır. Bu durum hizmetleri her zaman sağlayıcının kontrolü altında tutmakta, özelleştirme veya daha fazla geliştirme için esnekliği azaltmakta ve ek maliyetler oluşturmaktadır [33]. Bu nedenle, akıllı IoT sistemlerinin geliştirilmesini desteklemek için alternatif açık kaynak platformları ortaya çıkmaktadır ve bunlardan biri FIWARE'dir. Ara yazılım olarak hizmet veren bu platform, akıllı cihazları birleştirmekte, çevresel parametreleri izler, olayları ve kalıpları tanımlar ve buna göre harekete geçmek için algılanan verileri yönetmektedir.

IoT'nin endüstri, sağlık ve kişisel kullanımdan akıllı şehirlere ve akıllı evlere[34] kadar günlük hayatımız üzerindeki etkisi, veri tabanlarında depolanan ilgili verilerin hacmi ve türleriyle birlikte artmıştır [18]. Veri depolamaya yönelik yaygın çözümlerden biri de kalıcılık katmanlarıdır [35]. IoT cihazları, sensörler, mikrodenetleyici, ağ geçidi veya erişim noktası, internet ağı, depolama ve verileri görselleştirmek için kullanıcı platformu içermektedir [36].

Bu çalışmada, Raspberry Pi, FIWARE ara yazılım platformunu ve popüler açık kaynak görselleştirme aracı olan Grafana ile kullanılmaktadır. FIWARE, toplanan verileri düzenlemek ve bağlam hakkında daha fazla bilgi ile çevrelemek için sadece sensörlerden elde edilen verileri değil, aynı zamanda genellikle üçüncü taraf kaynaklardan elde edilen bağlam verilerini de tüketme özelliğine sahiptir [37].

1.2. PROBLEM TANIMI

Günümüzde, start-up'lardan, hobilerini ürüne dönüştüren bireylerden, akademisyenlerden ve teknoloji sektöründe faaliyet gösteren şirketlere yönelik birçok IoT cihazı bulunmaktadır. Ancak, bu çözümlerin büyük bir kısmı, IoT ortamlarında

veya akıllı şehir çözümlerinde üretilen tüm verileri etkin bir şekilde yönetmek için gerekli altyapıya sahip değildir [38]. IoT ara yazılım platformları, veri yönetimi yapısında kritik bir rol oynamaktadır [39]. Ancak, çoğu ara yazılım platformu kapalı kaynaklıdır ve lisanslama, destek ve bakım gibi yüksek maliyetlere sahiptir, bu da toplam proje maliyetlerini önemli ölçüde arttırabilmektedir.

Kalıcı geçmiş bağlam verileri, Big Data analizi için kullanışlıdır [40]; eğilimleri keşfetmek için kullanılabilir veya veriler örneklenebilir ve uzak veri ölçümlerinin etkisini ortadan kaldırmak için toplanabilir. Veri bütünlüğünü ve doğruluğunu garanti etmek için veri kalıcılığı esastır [41]. Bir IoT ara yazılım platformunda veri kalıcılığı olmadan, bir ağ üzerinden veya bir IoT cihazı tarafından gönderilen veriler bozulabilir, kaybolabilir veya değiştirilebilir [42]. Veri kalıcılığı, IoT cihazlarından güvenilir veri iletiminde önemli bir problemdir. Veri kalıcılığı, verilerin zaman içinde değişmeden ve erişilebilir kalabilme yeteneğidir [43]. Veri kalıcılığı ve görselleştirme, bir IoT ara yazılım platformunda özellikle önemlidir.

Günümüzde, veri madenciliği, istatistiksel analiz ve makine öğrenimi de dahil olmak üzere veri analizi için çeşitli teknikler bulunmaktadır. Bu tekniklerden biri, bir veri örneğinden önemli ve gerekli verilerin açık ve öz bir şekilde sunulmasını mümkün kılan veri görselleştirmedir [44]. Bir FIWARE ara yazılım platformunda geçmiş verileri görselleştirmek hiç bu kadar kolay olmamıştı, özellikle de bu veriler gerçek zamanlı olarak ortaya çıktığında daha zor hale gelmektedir. Birçok farklı kullanıcı grubu ve birden fazla heterojen veri kaynağı ile uğraşmak zorunda olduğumuz için bu problem Akıllı Şehir ortamlarında daha da büyük hale gelmektedir. Uygun bir görselleştirme metodolojisi veya aracı olmadan, farklı nitelikteki verileri içeren karmaşık gösterge panolarının anlaşılması zordur [45].

Bu araştırmanın amacı, IoT kavramını kullanan sistemlerin oluşturulmasıyla ilgili problemleri analiz etmek ve IoT'nin yaygınlaşmasına katkıda bulunabilecek basit bir sistem geliştirmektir. Geliştirilmiş sistemi, algılamadan veri görselleştirmeye kadar olan süreci kapsayan bir yapıyı, düşük maliyetli bir cihazla destekleyerek veri kalıcılığı ve görselleştirme için ücretsiz ve açık kaynaklı teknoloji çözümlerini sunmayı amaçlamaktadır.

1.3. ARAŞTIRMA SORULARI

Bu tez, aşağıdaki sorulara cevap vermeyi amaçlamaktadır:

1. IoT kavramında, bir sensörden ölçülen verileri ara yazılım platforma göndermek için Raspberry Pi kullanarak bağlam güncellendiğinde bir veri tabanındaki geçmiş durum değişikliklerini sürdürmek ve görselleştirmek için düşük maliyetli teknolojiler kullanılarak nasıl uygulanabilir?
2. Bir Raspberry Pi tarafından yakalanan veri yığını, açık kaynaklı bir IoT ara yazılım platformunda nasıl kalıcı hale getirilebilir ve görselleştirilebilir?
3. Grafana veri görselleştirme aracı kullanılarak FIWARE'de geçmiş bağlam veri kalıcılığı, IoT'nin başarılı bir şekilde konuşlandırılmasıyla ilgili zorlukların çözülmesine nasıl yardımcı olabilir?

1.4. ARAŞTIRMA HEDEFLERİ

Bu tez çalışmasının ana hedefi, FIWARE tarafından desteklenen basit bir sistem mimarisi oluşturmaktır. Düşük maliyetli bir cihaz kullanarak sensör verilerini toplamak ve veri kalıcılığı ile görselleştirme için ücretsiz ve açık kaynaklı teknolojik çözümler kullanmak hedeflenmektedir. Sistem, algılamadan veri görselleştirmeye kadar olan süreci kapsayan bir yapıyı desteklemektedir. Ayrıca, sistem mimarisinin karmaşıklığını artırmadan ve daha basit veri kalıcılığı ile görselleştirme sağlamak için gereken az sayıda FIWARE bileşenini seçerek ve kullanarak önemli bir katkı sunmayı amaçlamaktadır.

Bu amaca ulaşmak için, çalışma aşağıdaki spesifik hedefleri tanımlamaktadır:

1. FIWARE ara yazılım platformunu anlamak ve kullanılacak bileşenleri seçmek.
2. Akıllı şehirler için IoT ve ara yazılım platformları üzerine bir çalışma yürütmek.
3. FIWARE platformuna veri gönderen bir Raspberry Pi'nin nasıl bağlanacağını göstermek.

4. En yaygın IoT iletişim protokollerini araştırmak.

1.5. ARAŞTIRMANIN KAPSAMI

Bu araştırma, bağlamsal bilgilerin yönetilmesine yönelik en gelişmiş çerçevelerden biri olan FIWARE platformunun teknolojik referansına odaklanmaktadır. Yenilikçiliği nedeniyle, bu teknolojiyi operasyonel hale getirmek için sadece birkaç girişimde bulunulmuştur. Şimdiye kadar, FIWARE topluluğu tarafından piyasada, akademide, şirketler ve araştırmacılar tarafından yaygın olarak kullanılan metodolojilerin birkaç örneği, sistemleri daha kolay geliştirmek için hangi olasılıkların hala keşfedilebileceğini kontrol etmek için yayınlanmıştır. Bu çalışma, tamamen ücretsiz ve açık kaynaklı bileşenlere dayanan bir Grafana veri görselleştirme aracı ile FIWARE'de geçmiş bağlam veri kalıcılığı için bir çözümün artık erişilebilir olduğunu ve operasyonel bir ortamda Raspberry Pi gibi ilgili kullanım durumlarının üstesinden gelebileceğini göstermeyi amaçlamaktadır. Ayrıca, kullanılan yaklaşım, harici bileşenlerle birlikte çalışabilirlik ve ölçeklenebilirlik gibi önemli konuların ele alınmasını mümkün kılmıştır.

1.6. TEZİN DÜZENLENMESİ

Bu tez beş bölümden oluşmaktadır. Birinci Bölüm Giriş'tir; bu bölümde konu ve çalışmanın amaçları bağlamsal olarak ele alınmaktadır. Bölüm 2, IoT kavramı, FIWARE bileşenleri, Grafana veri görselleştirme aracı ve araştırma konusuyla ilgili literatürdeki çalışmalar gibi araştırmanın geliştirilmesi ve anlaşılması için önemli kavramların literatür taramasını sunmaktadır. Bölüm 3, veri görselleştirme için Grafana kullanarak geçmiş bağlam verileri kalıcı hale getirmek için FIWARE ara yazılım platformunu kullanan IoT tabanlı entegrasyon önerisinde kullanılan materyalleri ve metotları sunmakta ve Docker konteynerinde bileşenleri örneklenme yapılanması göstermektedir. Bölüm 4'te gerçekleştirilen deneyler, sistem konfigürasyonları ve geçmiş bağlam verilerin nasıl kalıcı hale getirileceği ve görselleştirileceği anlatılmaktadır. Bölüm 5'te sonuç ve gelecekteki çalışmalar sunulmaktadır.

BÖLÜM 2

LİTERATÜR TARAMASI

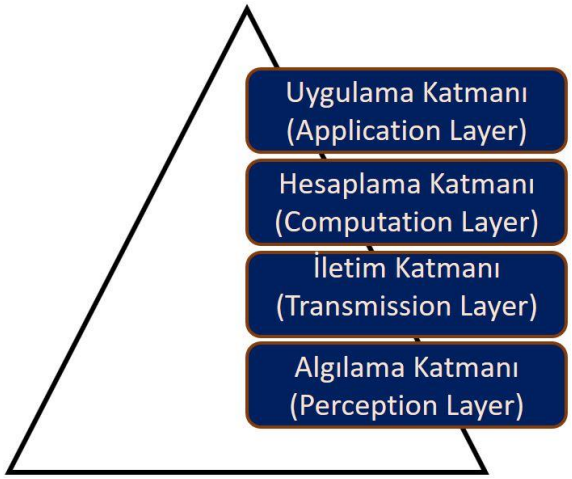
IoT, aniden ortaya çıkan bir kavram olmamakla beraber aynı zamanda gelişimine katkıda bulunan çeşitli teknolojiler bulunmaktadır. Örneğin, Kablosuz Sensör Ağları (WSN), çoklu atlama (multi-hop) iletişim kuran bir dizi kablosuz algılama düğümünden oluşmaktadır [46]. Ayrıca, Makineden Makineye (M2M) iletişim, insan müdahalesini gerektirmeyen herhangi bir iletişim kanalını kullanan cihazlar arasında doğrudan kablolu veya kablosuz iletişimi ifade etmektedir [47].

2.1. NESNELERİN İNTERNETİ (IoT)

1999 yılında Kevin Ashton, Radyo Frekansı ile Tanımlama (RFID) teknolojisi ile İnternet fikrini birleştirerek "Nesnelerin İnterneti" terimini ilk kez ortaya atmıştır [48]. Basit bir ifadeyle IoT, fiziksel ve sanal nesnelerin akıllı sensörler ve ağlar aracılığıyla birbirleriyle ve kullanıcılarla iletişim kurma ve veri alışverişi yapma şeklidir [49]. Sonuç olarak, her şeyin daha akıllı ve duyarlı bir gezegen oluşturmak amacıyla birlikte çalıştığı bir durumu ifade etmektedir.

IoT, tek bir teknolojinin sonucu değil; aksine, işlevsellik sağlayan fiziksel ve sanal dünyalar arasındaki boşluğu doldurmaya yardımcı olan çeşitli tamamlayıcı teknolojilerin bir kombinasyonudur [50]. Bu kavramları uygulanabilir kılmak için bir iletişim standardına, sistemler, makineler ve insanlar arasında bilgiyi temsil etmenin bir yoluna ihtiyaç vardır; böylece, herkes belirsiz yorumlar olmadan bilgi alabilir ve iletebilir. Anlambilimin kullanımı, bu probleme uygulanabilir bir çözüm olabilmektedir [51]. Genellikle heterojen olan gerçek dünya nesneleri arasında iletişim ve etkileşim için bu kapasiteyi sağlamak amacıyla esnek bir mimariye sahip olmak gerekmektedir.

Şu anda bir dizi önerilen mimari bulunmaktadır, ancak bunların hiçbiri referans model olarak kabul edilmemektedir. TRAPPEY vd. [52] tarafından yapılan çalışma, IoT ile ilgili standartların bir incelemesini sunmakta ve üç katmandan dokuz katmana kadar bir dizi mimari içinde yazar, unsurların mantıksal olarak Algılama (Perception), İletim (Transmission), Hesaplama (Computation) ve Uygulama (Application) katmanları arasında bölünebileceği sonucuna varmaktadır. Şekil 2.1'de dört katmanlı mimari gösterilmektedir.

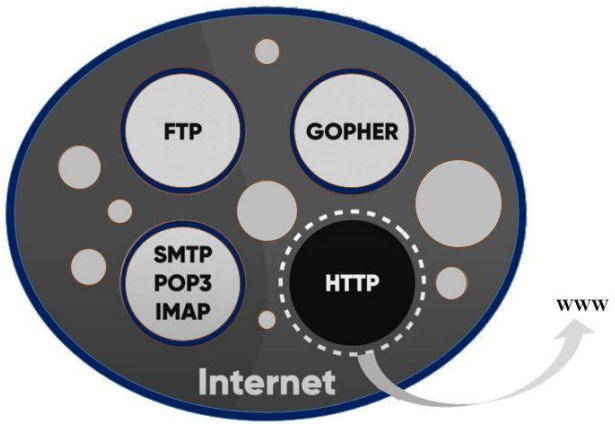


Şekil 2.1. IoT katmanları.

Algılama katmanının temel görevi, nesneleri tanımak ve algılamak ile bu nesnelerden bilgi toplamaktır. Bu katman, düşük maliyetli ve düşük güç tüketimine sahip cihazlarla uyumluluğu ön planda tutar. İletim katmanı, toplanan bilgilerin İnternet üzerinden iletilmesinden sorumludur. Hesaplama katmanı, verilerin alınması, depolanması ve istemci uygulamalarının erişimine sunulmasıyla ilgilenmektedir. Uygulama katmanı ise, kullanıcılar arasında bilgi paylaşımını sağlayan ve bu bilgilerin güvenliğini sağlayan bir araç görevi görmektedir [52].

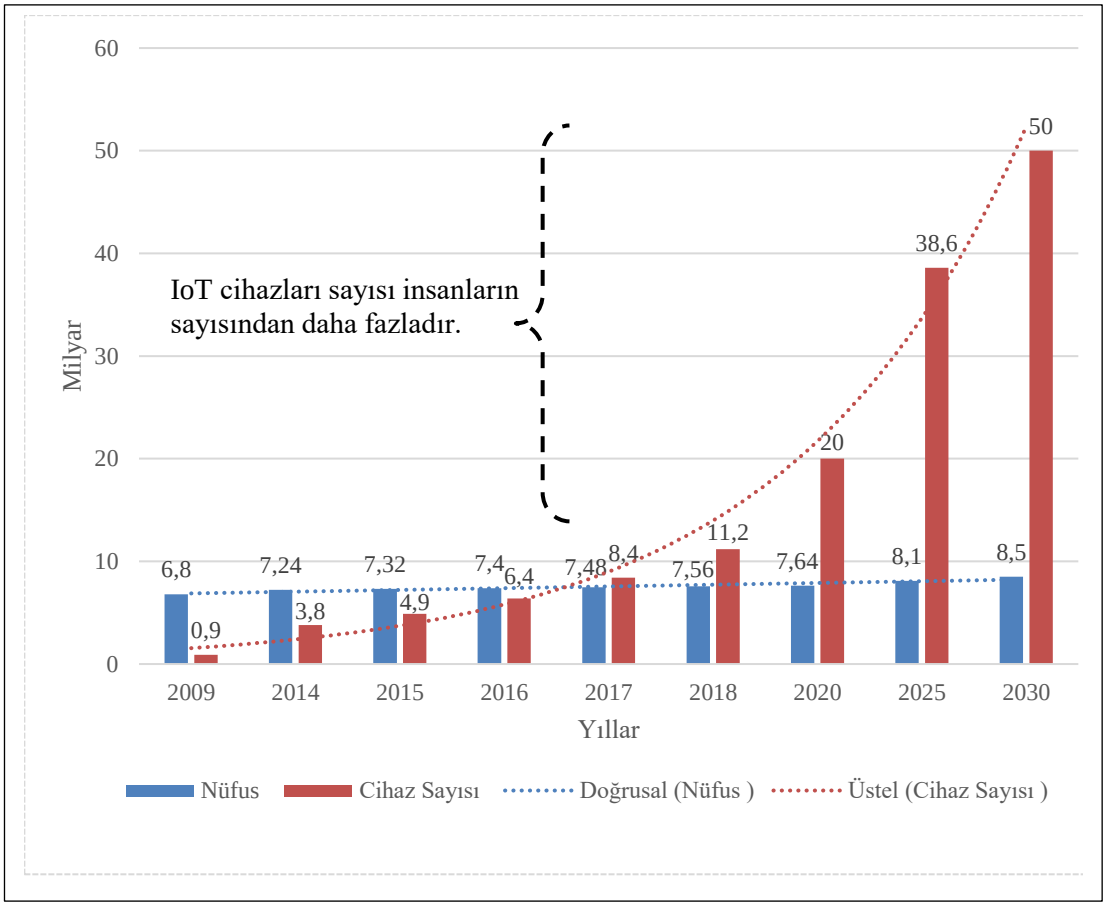
IoT'nin önemini kavramak için ilk adım, Dünya Çapında Ağ (WWW) ile İnternet arasındaki farkı anlamaktır. İnternet, ağların ağıdır [53] ve belirli bir hizmet veya protokol türünde uzmanlaşmış bir dizi sunucudan oluşmaktadır. İnternet, anahtarlar, yönlendiriciler ve diğer fiziksel cihazlardan oluşan bir altyapıya dayanmaktadır [54]. İnternetin temel görevi, bilgiyi hızlı, güvenilir ve güvenli bir şekilde bir noktadan

diğerine iletmektir [55]. Hiper Metin Aktarım Protokolü (HTTP) gibi belirli protokollere özgü sunucular bulunmaktadır. WWW, İnternet'in bir alt ağıdır [56] ve özellikle HTTP protokolüne uzmanlaşmış bölümü ifade eder. Web, İnternet üzerinde çalışan bir uygulama katmanıdır ve bilgiyi erişilebilir içeriğe dönüştürmek için bir arayüz sağlamak amacıyla tasarlanmıştır. Şekil 2.2'de, WWW ile İnternet arasındaki farkı göstermektedir.



Şekil 2.2. WWW ve İnternet arasındaki fark.

IoT, insanlığın ilerlemesi için temel bir öneme sahiptir, çünkü önümüzdeki on yıllarda hayatta kalmak ve gelişmek için gereken bilgi ve bilgelik kaynaklarını elde etme sürecinde verilerin tespit edilmesi, toplanması, iletilmesi, analiz edilmesi ve dağıtılması gibi beceriler, insanların bilgi işleme biçimiyle birleştirilir. Nand vd. [3] tarafından ifade edildiği üzere, IoT insanlardan daha fazla "şey veya nesnenin" internete bağlı olduğu bir paradigmayı temsil etmektedir. Şekil 2.3'te, son birkaç yılda dünya çapında IoT'ye bağlanan insan ve cihaz sayısını ve 2030'a kadar olan projeksiyonu karşılaştırmaktadır.



Şekil 2.3. Son birkaç yılda dünya çapında IoT'ye bağlanan insan ve cihaz sayısını ve 2030'a kadar olan projeksiyonu karşılaştırmaktadır [3].

2.1.1. IPv4 ve IPv6

İnternet, TCP/IP adı verilen bir dizi protokole dayalı olarak çalışmaktadır [57]. Bir protokol, tüm iletişimlerin aynı standartları takip etmesini sağlayarak, farklı teknolojilere sahip farklı cihazların mesaj alışverişinde bulunmasına olanak tanımaktadır. TCP/IP'nin, daha spesifik olarak IP'nin işlevlerinden biri, ağa bağlı her bir noktayı tanımlamaktır [58]. İnternete bağlandığımızda, benzersiz bir kimlik alınır ve bu kimlik bir IP adresidir. Eski IP adresleri (IPv4), noktalarla ayrılmış 8 bitlik gruplar halinde olan 4 oktet kullanır ve her bir tanımlayıcı başına toplam 32-bit uzunluğa sahiptir.

Örn: 123.45.67.89 = 01111011.00101101.01000011.01011001

Ancak, internete bağlı çok sayıda cihazın (örneğin bilgisayarlar, saatler, ev aletleri, vb.) artmasıyla birlikte, IPv4 adresleri tükenmektedir. Bu duruma çözüm olarak, IPv6 kullanımı önerilmektedir [59]. IPv6, IPv4'ten 4 kat daha fazla bit kullanarak toplamda 128 bitlik bir adresleme alanı sağlamaktadır.

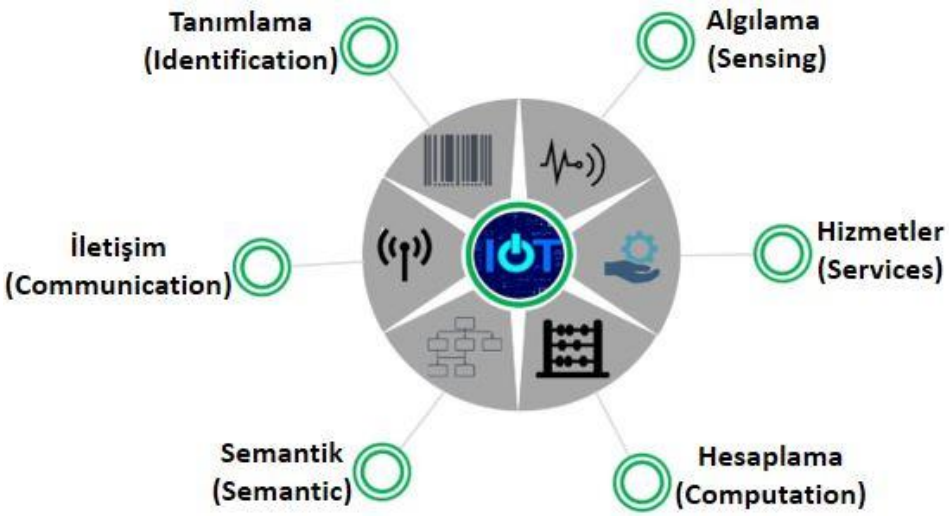
Örn: 2001:0db8:85a3:08d3:1319:8a2e: 0370:7344

IPv6, adres alanının 128-bit olması ve çok sayıda benzersiz IP adresini adresleyebilmesi nedeniyle İnternet'in ihtiyaçlarını uzun süre karşılayacaktır. Hala en yaygın kullanılan sürüm olan IPv4'e rağmen, birçok şirket IPv6 kullanmaktadır ve bazı ağ ekipmanları standart olarak IPv6 desteği ile üretilmektedir. Yavaş yavaş IPv4, IPv6'ya yerini bırakmakta ve bu da daha fazla insanın ve nesnenin internete erişmesine olanak tanımaktadır.

IPv4 ve IPv6 doğrudan uyumlu değildir çünkü IPv6, IPv4'ün bir uzantısı veya tamamlayıcısı olarak değil, adres tükenmesi problemini çözmek için bir yedek olarak tasarlanmıştır [60]. Bununla birlikte, bu protokoller birlikte çalışabilir olmasalar da, aynı cihaz üzerinde yan yana çalışabilirler ve kademeli bir geçişe izin verirler. IPv6'nın başarılı bir şekilde yaygınlaştırılması IoT'nin büyümesi için temel bir öneme sahiptir.

2.1.2. IoT'nin Temel Yapı Bloklar

IoT, çeşitli teknolojilerin birleşimidir. Al-Fuqaha vd. [15]'ne göre, Şekil 2.4'te, IoT'nin temel yapı bloklarını göstermektedir.



Şekil 2.4. IoT'nin unsurları.

1. Tanımlama (Identification): Bilgiyi çevremizdeki gerçek varlıklarla ilişkilendirmenin önemli bir ön koşulu, nesnelerin ve insanların net bir şekilde tanımlanmasıdır. Çünkü, nesneleri internete bağlamak için onları benzersiz bir şekilde tanımlamak esastır. Nesneleri tanımlamak için RFID, Yakın Alan İletişimi (NFC) ve IP adresleme gibi teknolojiler kullanılmaktadır.

2. Sensörler/Aktüatörler (Sensors/Actuators): Sensörler, belirli kimyasal veya fiziksel değişkenlerin ve özelliklerin (örneğin sıcaklık, ışık, ivme, elektrik vb.) nitel veya nicel ölçümü için kullanılan teknik bileşenlerdir. Sensörler aracılığıyla nesneler çevrelerinden bilgi toplar ve bu veriler depolanabilir veya veri ambarlarına, bulutlara veya depolama merkezlerine iletilir. Aktüatörler söz konusu olduğunda, elektrik sinyallerini (örneğin kontrol bilgisayarından gelen komutlar) mekanik harekete veya diğer fiziksel değişkenlere (basınç veya sıcaklık gibi) dönüştürerek kontrol sistemine aktif müdahalede bulunur ve/veya değişkenlerin yapılandırılmasına izin verirler. Nesneler, çevreyi manipüle edebilir veya toplanan verilere göre tepki verebilirler.

3. Hizmetler (Services): IoT, çeşitli hizmet türleri sunmaktadır; bunların en temel ve önemlisi tanımlama hizmetleridir. Uygulamalar, fiziksel varlıkları sanal varlıklarla eşleştirmek için bunları kullanılmakta ve gerçek dünyadaki

nesneleri Sanal Dünyadaki karşılık gelen nesnelerle eşleştirmeyi kolaylaştırmaktadır. Ayrıca, akıllı nesnelerden ham verilerin toplanması ve bir araya getirilmesinden oluşan Bilgi Toplama Hizmetleri bulunmaktadır. Daha sonra bilgi toplama hizmetleri üzerinde çalışan, alınan ve işlenen verileri karar vermek ve uygun şekilde yanıt vermek için kullanılan işbirliği ve istihbarat hizmetleri de vardır. Son olarak, ihtiyaç duyulan her yerde işbirliği ve istihbarat hizmetleri sağlayan Ubiquity Hizmetleri mevcuttur.

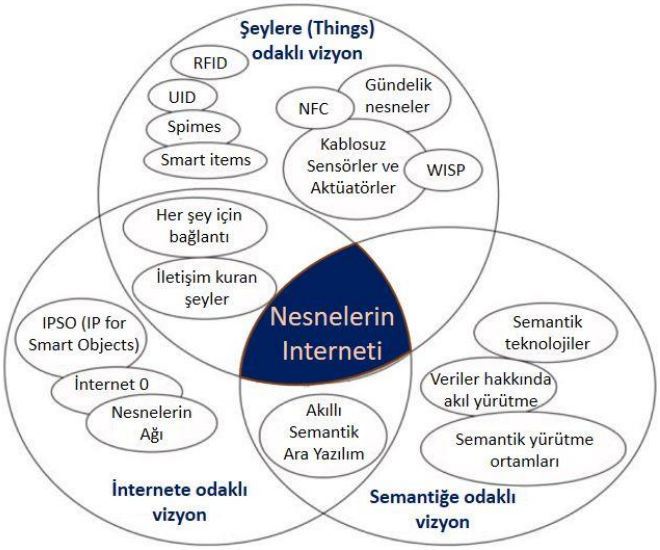
4. Hesaplama (Computation): Mikrodenetleyiciler ve işlemciler gibi işlem birimleri, akıllı nesnelerin kendi kendilerini hesaplamalarını sağlamaktadır.

5. Semantikler (Semantics): Bu, IoT'deki çok çeşitli kaynaklardan bilgi çıkarma yeteneğini içermektedir. Başka bir deyişle, anlambilim kaynaklarını keşfetmeyi ve belirli hizmetlerin sunulmasını sağlamak için bunları akıllıca kullanmayı içermektedir. Anlambilim ayrıca verileri öğrenir ve analiz eder, böylece kararları doğru bir şekilde uygulamak mümkün hale gelmektedir.

6. İletişim (Communication): İletişim sayesinde akıllı nesneler birbirine bağlanmaktadır. Bağlanabilirlik aynı zamanda nesnelerin enerji tüketimini de etkilemekte; çünkü enerji, çoğu nesne ve sensör için sınırlı bir kaynaktır, bu yüzden IoT için kritik ve zorlu bir faktördür. İletişim için kullanılan teknolojilere örnek olarak kısa menzilli (Bluetooth, Zigbee, ANT ve RFID), orta menzilli (Wi-Fi ve hücresel) ve uzun menzilli (LoRa[9], NB-IoT ve SigFox) teknolojileri verilebilmektedir.

2.1.3. IoT Perspektifleri veya Vizyonları

IoT, üç ana perspektifin birleşimi olarak anlaşılabilmektedir. Şeyler (Things) odaklı bir vizyon, İnternet odaklı bir vizyon ve semantik odaklı bir vizyon [1]. Şekil 2.5'te bu perspektifler ve her bir vizyona ait bazı unsurlar gösterilmektedir.



Şekil 2.5. Farklı vizyonların bir araya gelmesinin bir sonucu olarak "Nesnelerin İnterneti" paradigması.

1. Şeylere (Things) odaklı bir vizyon: Bu vizyon, nesnelerin benzersiz bir şekilde tanımlanması, eylemlerin uygulanması ve fiziksel cihazların durumunun okunması gibi verileri gerçek ve sanal dünyalar arasında tercümesi bulunmaktadır. Bu alanda, cihazın yerleştirildiği ortama duyarlılık artmaktadır. İnsanların internet üzerinden iletilen verilerin sağlayıcıları ve tüketicileri arasında azınlık haline geleceğine, yani ağın makineler ve cihazlar tarafından daha fazla kullanılacağına inanılmaktadır.

2. İnternete odaklı bir vizyon: İnternette kullanılan bilgisayarlar arasındaki iletişimin çoğunu standartlaştıran TCP/IP gibi teknolojilerin ortaya çıkmasıyla, hepsi tek bir ağa bağlı ve benzersiz tanımlamaya sahip cihazlar oluşturmak mümkün olmuştur. Ancak bu bakış açısına göre, cihazlar ucuzlayıp küçüldükçe ve bunun sonucunda daha az enerji tükettikçe, cihazlardan enerji talebi de artmaktadır.

3. Semantiğe odaklı bir vizyon: Farklı cihazlar ve sistemlerin birbirine bağlanabildiği ve bilgi alışverişinde bulunabildiği için birlikte çalışabilirlik IoT'de kilit bir kavramdır. Ontolojiler, mantık ve ontolojik arama gibi uygulamalar, bu alana katkı sağlamaktadır.

2.1.4. Bulut İletişim Protokolleri

IoT, son yıllardaki patlayıcı büyümesiyle tarihimizde benzersiz bir duruma yol açmıştır. Sürekli olarak çalışan birçok bağlantılı cihazla çevrili olarak çevremizi daha güvenli ve daha konforlu hale getirmeye çaba gösterilmektedir ve çoğu zaman bu cihazlar görünmezdir. Bu cihazların potansiyeli, iletişim kurabildikleri zaman ortaya çıkar ve bu da düşük maliyetli cihazlar için hafif ve ölçeklenebilir bir iletişim protokolüne olan talebi artırmaktadır. IoT'de, bağlı cihazların veri alışverişi yapmak ve komut almak için bulut platformlarıyla iletişim kurmaları gerekmektedir. Bu bilgi alışverişini kolaylaştırmak için kullanılan çeşitli iletişim protokolleri bulunmaktadır [61]. IoT'de bulut iletişimine yönelik en yaygın protokollerden bazıları şunlardır:

2.1.4.1. HTTP

HTTP, istemciden gelen bir istek yoluyla veri elde edilmesini sağlayan, uygulaması kolay bir istemci-sunucu iletişim protokolüdür [62]. Bu protokol, web sitelerinde gezinirken tarayıcılar tarafından yaygın olarak kullanılmaktadır. İstemci-sunucu, TCP katmanında işleyen ve Taşıma Katmanı Güvenliği (TLS) kullanılarak şifrelenebilen istekler ve yanıtlar adı verilen mesajlar aracılığıyla iletişim kurulmaktadır [63]. İstemci bir hizmete istekte bulunur ve sunucu bu isteği aldıktan sonra onu işler ve istenen şeyi geri döndürmektedir. HTTP protokolünde farklı istek yöntemleri vardır [64]. En yaygın kullanılanları şunlardır:

1. GET: Genellikle, belirli bir kaynağı talep etmek için "GET" yöntemi kullanılır. Bu yöntem, genellikle veri isteklerinde bulunmak için kullanılmaktadır. Parametreler, bu yöntemin kullanıldığı Tekdüzen Kaynak Konum Belirleyicileri (URL)nin gövdesinde sağlanmaktadır. Örneğin:

   ```
   www.myexample.com/?parameter=value&parameter2=value2...
   ```

2. POST: Belirli bir kaynağa veri göndermek için kullanılmaktadır. Genellikle form verilerini sunucuya göndermek için tercih edilir ve genellikle bir durum değişikliğine veya yeni bir kaynak oluşturmaktadır.

2.1.4.2. Mesaj Kuyruklama Telemetri Aktarımı (MQTT)

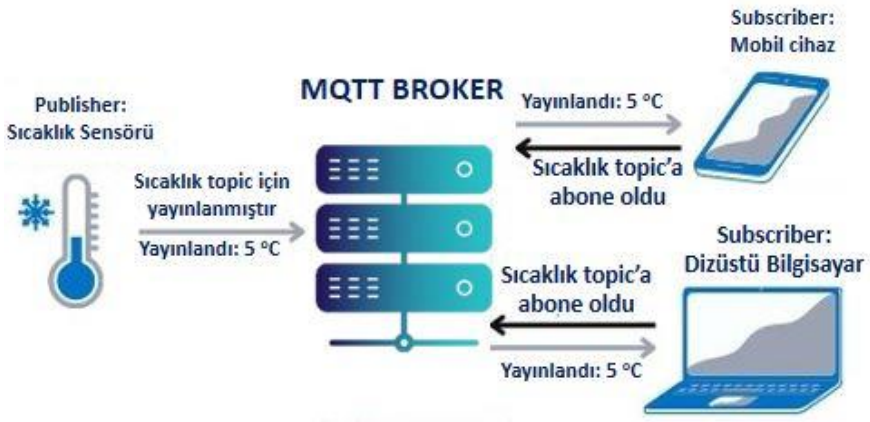
MQTT, başlangıçta IBM ve Eurotech mühendisleri Andy Stanford-Clark ve Arlen Nipper tarafından geliştirilen bir Publish/Subscribe iletişim protokolüdür [65]. İlk olarak, boru hattı telemetri sistemlerini uydu üzerinden bağlamak amacıyla tasarlanmıştır. Günümüzde ise hızla yaygınlaşmakta olup IoT cihazları arasındaki ana iletişim araçlarından biri haline gelmektedir [66].

TCP/IP kullanan MQTT, düşük bant genişliği oranlarında bile iyi performans sergileyen ve az işlem gücü gerektiren hafif bir protokol olarak tasarlanmıştır [65]. Bu başarı, MQTT mesajlarının taşıdığı verilerin yalnızca mesaj yükünden oluşmasına dayanmaktadır. MQTT'yi uygulamak için 3 temel aktör gerekir: yayıncı (publisher), abone (subscriber) ve broker [67]. Bu aktörler, topic aracılığıyla iletişim kurulmaktadır(Şekil 2.6).

Topic: Mesajların iletilmesinde bir kanal gibi işlevi görmektedir [67]. Topic, bir URL gibi eğik çizgilerle ayrılır ve içeriği kategorize ederek iletişimi organize etmektedir. Örneğin:

```
üniversite/ $üniversite_adı / $kurs / öğrenci
```

Yukarıdaki konu başlığı, yalnızca öğrencilere ait bilgi trafiği için kullanılabilen üniversitenin adı ve kursunun belirlendiği mesaj yayını sırasında kullanılmaktadır.



Şekil 2.6. MQTT protokol aktörleri.

1. Publisher: Bir cihaz, bir topic aracılığıyla brokera mesaj gönderebilir. Mesaj yayınlandıktan sonra broker tarafından alınır ve topic'e abone olan tüm abonelere dağıtılır.

2. Subscriber: Bir publisher tarafından gönderilen mesajları almak için bir topic'e abone olan herhangi bir cihazdır. Abone, brokera bağlanmalı ve ilgili topic'i seçmelidir; böylece bu topic'te bir mesaj yayınlandığında, mesajlar yeniden yönlendirilir.

3. Broker: Bu, MQTT iletişiminin merkezidir. Tüm mesajların topic abonelerine iletilmek üzere geçtiği yerdir. Broker bir sunucu gibi davranır ve yayıncıdan aldığı bilgileri yöneterek mesaj tarafından belirtilen topic'deki abonelere iletir. Aracı, mesajları topic aracılığıyla depolar ve filtreler, böylece yalnızca bu bilgi kaynağını talep eden alıcılara teslim edilirler.

TCP/IP üzerinden iletişimi içerdiği için, MQTT, "publisher x broker" ve "broker x subscriber" arasındaki iletişim için ayrı ayrı Hizmet Kalitesi (QoS) [68] standartlarının tanımlanmasına izin verir. Yani, mesaj alışverişi için her bir bağlantı farklı bir QoS seviyesine sahip olmaktadır. MQTT protokolünde 3 QoS standardı bulunmaktadır [69]:

1. QoS 0: "Best Effort" olarak adlandırılan bu seviye, UDP taşıma protokolüne benzer şekilde çalışır; yani, mesaj teslim onayı olmadan işlev görür.
2. QoS 1: QoS 0'dan farklı olarak, bu seviye mesaj teslimatının onaylanmasını içerir.
3. QoS 2: Bu seviyede ayrıca alındı onayının alındığına dair bilgi de bulunur, yani iletişime dahil olan her iki taraf da onayı sağlar. Böylelikle, her iki taraf da mesajın doğru bir şekilde teslim edildiğini bilir.

2.1.4.3. Kısıtlı Uygulama Protokolü (CoAP)

CoAP, bire bir request/response etkileşimi sağlayan ve çoklu yayını destekleyen bir protokoldür. Bu protokol client/server modelini kullanmaktadır [70]. MQTT'den farklı

olarak, CoAP, IoT protokollerinin HTTP ve RESTful mimarileriyle uyumlu olmasını sağlamak için basit proxy'ler aracılığıyla geliştirilmiş ve İnternet ile uyumlu hale getirilmiştir [71]. CoAP, düşük hesaplama ve enerji tüketimi için UDP protokolünü kullanır. Bu protokol, tetiklendiğinde daha kısa bir yanıt süresi sağlayarak düğümler arasında aktif bir bağlantı sağlamaktadır [72]. CoAP'ın mimarisi HTTP'ye benzediği için yerel düğümlere komut göndermek için daha uygundur [64]. Ayrıca, yukarıda açıklanan özellikler nedeniyle, daha az bilgi işlem kaynağına sahip cihazlarda en iyi şekilde kullanılmaktadır.

2.1.5. IoT Güvenliği

Shelby vd. [73] belirtildiği üzere, IoT güvenliği için en az üç ana hedef grubu arzu edilmektedir:

1. Gizlilik (Confidentiality): Bu gereklilik, iletilen verilerin yalnızca iletişime katılan unsurlar tarafından "duyulması" ve anlaşılması gerektiğini belirtmektedir; yani, yetkisiz taraflar iletişimin gerçekleştiğini bilebilir, ancak içeriğine erişemezler.

2. Bütünlük (Integrity): Bu, uygun yetkilendirme olmadan ağ elemanları tarafından değiştirilemeyen veriyi ifade etmektedir. Bütünlüğü sağlamanın bir yolu, mesajları şifrelemek ve alıcı uçta kontrol etmektir.

3. Kullanılabilirlik (Availability): Sistemin her zaman kullanılabilir olması ve kötü niyetli saldırılara karşı güvenli olması arzulanır. Kablosuz ağlar, bu güvenlik açığından yararlanan bilgisayar korsanları tarafından ele geçirilebilir. Bu nedenle, bir IoT sistemi, Hizmet Reddi (DoS)[74] saldırıları gibi problemleri tanımlamaya ve bunlarla başa çıkmaya hazır olması gerekmektedir.

Bütünlüğü sağlamak için İnternet Protokolü Güvenliği (IPsec) kullanılabilir [75], ancak standart IPsec'in yaygın IoT uygulamaları için çok büyük olduğu düşünülmektedir, bu nedenle bazı uyarlamalar gerekmektedir. IoT güvenlik

gereksinimlerinin uygulamadan uygulamaya değiştiğini belirtmek gerekir, bu yüzden bir uygulamayı geliştirirken belirtilen hedefleri analiz etmek önemlidir [76, 77].

2.2. AKILLI ŞEHİRLER İÇİN IoT ARA YAZILIM PLATFORMLARI

IoT ile ilgili problemlerden biri, sistemi oluşturan unsurların heterojenliğidir [27], çünkü bu kavram farklı uygulamalara sahip olabilecek farklı "şeylerin" entegrasyonunu önermektedir [78]. Bu problem, cihazlar, OS ve mimariler arasında bir iletişim arayüzü sağlayarak, bu "şeyler" ve uygulamalar arasında ara yazılım kullanımı yoluyla çözülebilmektedir [79]. Bu çalışmada analiz edilen platformlara ek olarak, Pires vd. [80], akıllı şehirler bağlamında uygulanabilir on bir IoT platformunu analiz etmiş ve her birinin IoT gereksinimlerini nasıl karşıladığını bildirmiştir. EcoDiF, Xively, Carriots, LinkSmart, OpenIoT, RestThing, WoT Enabler, S3OIA, UbiWare, WSO2 ve INRIA ARLES platformlarını inceleyen yazarlar, bunların hiçbirinin IoT ara yazılım platformları tarafından ele alınması gereken tüm önemli gereksinimleri karşılamadığı sonucuna varmışlardır. Bu nedenle, bu çalışmada SOFIA, FIWARE ve CityHub gibi diğer üç platform da analiz edilmektedir. Akıllı şehirlerin IoT paradigmasının bir uygulama alanını temsil ettiği düşünüldüğünde, akıllı şehirler için bir ara yazılım platformunun doğası gereği IoT odaklı platformlarla aynı gereksinimlere ve bu uygulama alanına özgü ek gereksinimlere sahip olması gerekmektedir.

Bu bölümde, akıllı şehir uygulamalarını geliştirmek için kullanılan ara yazılım platformları sunulmaktadır. Bölüm 2.2.1'de, Kanada'daki Vancouver ve Birleşik Krallık'taki Lancaster şehirleri tarafından kullanılan ve akıllı şehirler alanında yazılım çözümlerinin oluşturulmasını ve uygulanmasını sağlayan CityHub [81] platformu tanıtılmaktadır. Bölüm 2.2.2, etkileşim halindeki nesnelerden oluşan bir ekosistemi temsil eden akıllı ortam kavramını kullanan bir ara yazılım mimarisi olan SOFIA'yı sunmaktadır [82]. Bölüm 2.2.3, API'ler tarafından kullanıma sunulan ve jenerik etkinleştiriciler (GEs) olarak adlandırılan bileşenlerde uygulanan bir dizi açık spesifikasyonu tanımlayan, Avrupa Birliği'nin desteğiyle önerilen genel ve genişletilebilir bir platform olan FIWARE'i sunmaktadır [83]. Son olarak, bölüm 2.2.4'te, akıllı şehirler için ara yazılım gereksinimlerine genel bir bakış sunulmakta ve

açıklanan platformların her birinin desteklediği gereksinimlere göre konumlandırılması yapılmaktadır.

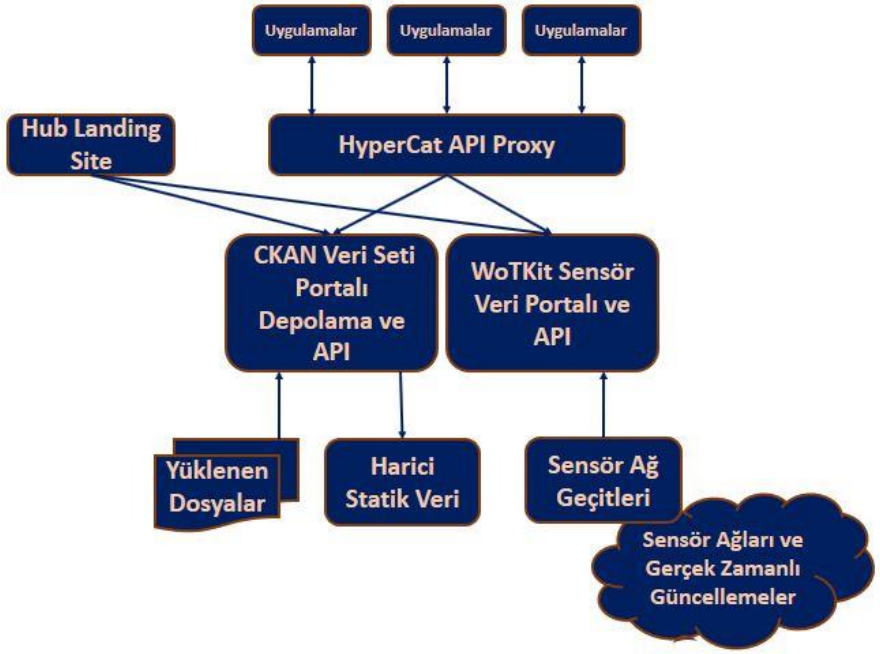
2.2.1. CityHub

CityHub platformu [81], şehir içindeki veri akışı için bir çözüm sağlama ihtiyacından doğmuştur. Böylece, bu verilere bir IoT altyapısı aracılığıyla kolayca erişilebilir hale gelir ve bulut bilişime dayalı bir merkez, şehrin yazılım altyapısını oluşturan çeşitli alt sistemleri toplu olarak entegre edebilmesi için bir sistemler metodolojisi desteklenmektedir. Platform, Kanada'daki Vancouver ve Birleşik Krallık'taki Lancaster şehirleri tarafından kullanılmış ve akıllı şehirler alanında yazılım çözümlerinin oluşturulmasına ve uygulanmasına olanak sağlamıştır. CityHub, hem statik verilere yönelik açık veri platformundaki gibi hem de IoT teknolojileriyle gerçek zamanlı olarak yakalanan verilere, kitle kaynak kullanımı yoluyla elde edilen verilere gibi dinamik verilere erişimi yönetmektedir [84].

CityHub, şehir ortamında üretilen verilere merkezi bir erişim noktası olarak PaaS hizmetleri sunmaktadır [85]. Her şehrin, çeşitli şehir sistemlerini bir araya getiren kendi merkezi vardır, böylece uygulamalar bu sistemlerden veri tüketebilmektedir. Bu gereklidir çünkü tek bir hub'ın farklı kamu ve/veya özel kuruluşlar tarafından sunulan hizmetler için sunucu olması pratik değildir. Bu nedenle, CityHub, hub'ların birbirleriyle uyumlu olmasını sağlayarak birden fazla şehirde çalışan sistem ve uygulamaların yeniden kullanılmasına olanak tanımaktadır. Bu amaçla, CityHub, hibrit bulutlar arasında uyumluluğu ele almak için soyutlamalar sunmaktadır.

CityHub tarafından ele alınan temel zorluk, geleneksel sistemlerden gelen statik verilerden yüksek veri üretim ve güncelleme oranlarına sahip kritik sistemlerden gelen gerçek zamanlı verilere kadar çeşitli türlerdeki verileri toplayabilme, yönetebilme ve sunabilme yeteneğidir [84]. Farklı veri türlerini işlemenin yanı sıra, CityHub'ın farklı veri sağlayıcılarının farklı özellikleriyle etkileşime girmesi gerektiğinden, bu önemsiz bir görev değildir. Tüm bunlar, geliştiricinin bu verileri tek tip ve RESTful bir API aracılığıyla tüketen uygulamalar geliştirebileceği şekilde yapılmaktadır.

Şekil 2.7'de CityHub mimarisinin bir örneği gösterilmektedir. Bu mimaride, Kapsamlı Bilgi Arşiv Ağı (CKAN) [86] şehrin açık ve statik verilerinin depolanmasından ve bunlara erişimin yönetilmesinden sorumludur. CKAN, birçok şehir tarafından kullanılan açık veri platformlarından biridir. Temel özelliklerinden biri statik ve zamansal verilerin depolanmasıdır. Hem insan tarafından okunabilir raporlar (örneğin PDF formatında) gibi dosyaları hem de Virgülle Ayrılmış Değerler (CSV) dosyaları gibi yazılım tarafından işlenebilir formatlardaki elektronik tabloları depolayabilmektedir. Genel olarak, raporlara doğrudan CKAN tarafından sağlanan web portalı üzerinden erişilebilmekte, yazılımla işlenebilir dosyalara ise Web API'leri aracılığıyla erişilebilmektedir.

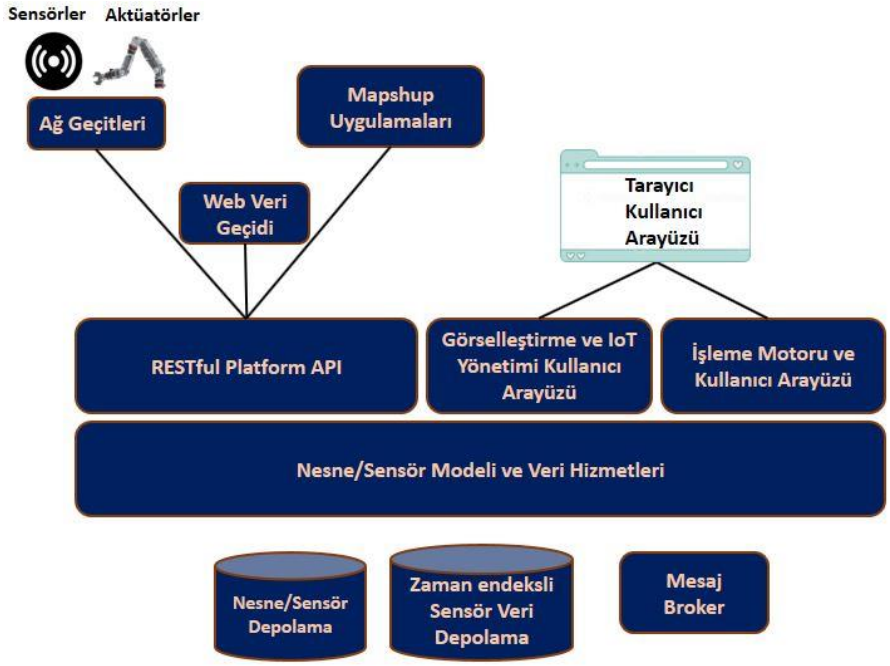


Şekil 2.7. Çekirdek IoT altyapısı, gerçek zamanlı ve gerçek zamanlı olmayan kentsel veri akışlarına erişimi destekler niteliktedir.

Şeyleri yönetmek ve davranışlarını gerçek zamanlı olarak izlemek için kullanılan bir IoT araç seti olan WoTKit [87], kullanıcıların ve uygulama geliştiricilerinin donanım veya yazılıma dayalı sensör soyutlamaları oluşturabilecekleri bir API veya grafik arayüz sağlamaktadır. Bir sensör soyutlaması, hem gerçek bir sensörden veri alabilir hem de bir aktüatöre komutlar verilebilmektedir. Ayrıca, sensörler gruplandırılabilir

ve bulunmalarını kolaylaştırmak için meta verilerle ilişkilendirilebilir. WotKit, uygulamalara gerçek zamanlı, büyük ölçekli veriler sağlamak üzere tasarlanmıştır.

WoTKit'in mimarisi, Şekil 2.8'de görüldüğü gibi, her katmanın belirli işlevleri yerine getirmekten sorumlu olduğu katmanlara dayanmaktadır [87]. En alt katman, sensörler tarafından yakalanan verilerin depolanmasından ve mesaj broker'ın (platformun katmanları arasında mesaj alışverişi yapabilen bileşenin) işlevlerinden sorumludur. Ara katman, veri ve model yönetiminden sorumludur. En üstteki son katman ise tarayıcı tarafından erişilen kullanıcı arayüzünde görüntülenecek verilerin işlenmesi ve yönetilmesinden, RESTful API aracılığıyla mashup uygulamalarının oluşturulmasından ve ağ geçitleri aracılığıyla sensörlerin/aktüatörlerin bağlanmasından sorumludur.



Şekil 2.8. WoTKit mimarisi.

Görünüşe göre CKAN ve WoTKit, CityHub'ın şehrin tüm veri kaynaklarını yönetmek için kullandığı platformlar birbirlerini tamamlamaktadırlar. Ancak, bu iki bileşen arasında iletişim kurmaması platformun bazı zorluklarla karşılaşmasına neden olmaktadır. Bu durum, statik ve dinamik verilerin platform düzeyinde birleştirilmesini zorlaştırır. Örneğin, bir uygulamanın toplu taşıma ile ilgili hem statik hem de dinamik

verilere ihtiyacı varsa, uygulamanın CKAN'dan gelen statik verileri WotKit'ten gelen dinamik verilerle ilişkilendirmesi gerekebilir. Bu, veri bütünlüğü ve tutarlılığı açısından ek bir zorluk oluşturabilmektedir.

Hypercat [88], uygulamaların her türlü bilgiye erişimini sağlayan bir bileşendir, bu bilgiler statik veri, dinamik veri, donanım veya yazılıma dayalı sensörlerden gelen veri gibi farklı kaynaklardan olabilmektedir. Bu bileşen, veri ve meta veri sorguları için bir hipermedya kataloğu belirtir. Görüntülenen kaynaklar, mevcut her bir kaynağın erişimi için formatı ve anlamı hakkında bilgi sağlayan komutların bir listesi olarak tanımlanmaktadır.

2.2.2. SOFIA

Akıllı Uygulamalar için Akıllı Nesneler (SOFIA) Ocak 2009'da başlayıp üç yıl süren ve Mart 2012'de sona eren ARTEMIS projesinin sonucunda ortaya çıkan bir platformdur [82]. Platform, etkileşim halindeki nesnelerden oluşan bir ekosistemi temsil eden akıllı bir ortam kavramı etrafında inşa edilmiştir: sensörler, mobil cihazlar, uygulamalar, gömülü sistemler. Bu nesneler kendi kendilerini organize etme, birleştirilmiş hizmetler sunma, karmaşık verileri işleme ve sağlama yeteneğine sahiptir.

SOFIA'nın amacı, farklı sistem ve cihazların birlikte çalışabilirliğini sağlayan bir ara yazılım mimarisi olmaktır [89]. Bu sistemler kümesi, bilgi depolamanın açık ve dağıtık olduğu ve arama prosedürlerinin, bunları geliştirmek için kullanılan teknolojilerden bağımsız olarak tüm uygulamalar için ortak olduğu akıllı bir ortam oluşturmaktadır [90]. Platform, bu bilgileri akıllı hizmetler için kullanılabilir hale getirerek fiziksel dünya ile bilgi dünyası arasında bir bağlantı sağlamayı amaçlamaktadır. Ayrıca yeni kullanıcı arayüzü ve etkileşim konseptleri oluşturarak, kullanıcı deneyimini zenginleştirerek ve akıllı ortamlardan faydalanmalarını sağlayarak inovasyonu teşvik etmeyi amaçlamaktadır.

SOFIA açık kaynaklıdır; çoklu dil, Java, C++, JavaScript, Arduino için taşınabilir; çoklu platform, Windows, iOS, Android ve Linux için kullanılabilir. SOFIA Platform

TCP, MQTT, HTTP, REST ve WebServices protokollerinin uygulamalarına sahiptir. ARTEMIS projesi tamamlandıktan sonra Indra, orijinal SOFIA projesini geliştirmeye devam ederek iş kullanımına odaklanan bir platform oluşturdu. Bu platformun mevcut versiyonu SOFIA2 olarak adlandırılmaktadır [91]. Platform, acil durum ve tahliye yönetimi, binaların, kamu altyapılarının ve endüstriyel süreçlerin işbirliğine dayalı otomasyonu, Enerjinin İnterneti (IoE), enerji [92] ağlarının internet üzerinden güvenli bir şekilde akıllı yönetimi, entegre ağ teknolojileri için insan-makine arayüzlerinin iyileştirilmesi gibi yeni özellikler de dahil olmak üzere SOFIA'nın ana kavramlarını korumuştur [93].

Platformun işlevleri şunları içerir:

1. Akıllı bir şehirde bulunan sensörlerden ve cihazlardan veri toplama.
2. Şehirdeki mevcut sistemlerle entegrasyon.
3. Kuralların ve CEP Motorunun değerlendirilmesi yoluyla karar verme sürecinde yardım.
4. Olaylara, alarmlara vb. abonelik.
5. Birden fazla cihaz için destek.
6. Evlerdeki cihazlardan bilgi toplama.
7. Büyük hacimli bilgilerin depolanması, işlenmesi ve karar verilmesi.
8. Evlerde bulunan çeşitli cihazların yönetimi.
9. Olaylara dayalı olarak alınacak önlemler için kurallar.
10. Sağlık sistemleri arasında bir iletişim kanalı olarak çalışır.
11. Geçmiş veri depolama.
12. Farklı platform ve dil türleri için API'ler.

SOFIA platformu dört kavram üzerinden kavramsallaştırılabilir [82]:

1. Akıllı Alan (Smart Space): Farklı uygulamaların ve cihazların daha karmaşık işlevsellik sağlamak üzere birbirleriyle etkileşime girdiği sanal bir ortamı temsil etmektedir. Akıllı Alanın çekirdeği SIB'dir, genellikle sadece bir SIB'ye sahiptir, ancak bazı durumlarda bir SIB federasyonuna sahip olabilir. Bir Akıllı Alan diğer Akıllı Alanlarla iletişim kurabilmektedir.

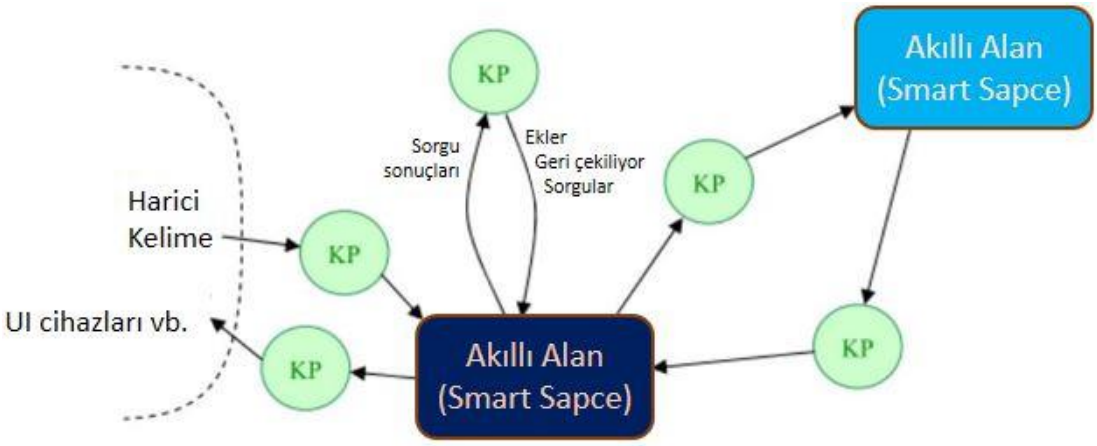
2. Semantik Bilgi Broker (SIB): Platforma bağlı uygulamalardan gelen tüm bilgilerin alınmasından, işlenmesinden ve saklanmasından sorumludur ve bir tür veri yolu görevi görmektedir. Etki alanındaki tüm mevcut kavramlar (ontolojilerde yansıtılır) ve bunların mevcut durumları (ontolojilerin belirli örnekleri) SIB'de yansıtmaktadır. Platform, bilgi alışverişi ve ontolojilerin tanımlanması için JSON kullanımını önermektedir. SIB'lerin çeşitli dillerde ve platformlarda uygulamaları vardır ve SIB'ler farklı istemci türleriyle iletişim kurmak için farklı türde bağlayıcılar sunar: Java Script ve akıllı telefon kullanan istemciler için REST, iki yönlü iletişim ve sınırlı cihazlar için MQTT, iş uygulamaları için Web Services/JMS, Bluetooth ve Zigbee vb.

3. Bilgi İşlemcisi (KP): bir SIB aracılığıyla Akıllı Alan ile etkileşime giren her bir unsuru (uygulama veya cihaz) temsil etmektedir. Her uygulama, atandığı etki alanındaki (ontolojiler aracılığıyla temsil edilen) ilgili kavramların örnekleriyle çalışmaktadır. Üç tür KP vardır: Üretici (Producer), SIB'ye bilgi ekleyen ancak ondan herhangi bir bilgi almayan bir KP'dir; Tüketici (Consumer), SIB'den bilgi alan ancak ona herhangi bir bilgi eklemeyen bir KP'dir; Prosumer, SIB'ye bilgi ekleyen ve aynı zamanda bilgi alan bir KP'dir.

4. Akıllı Alan Erişim Protokolü (SSAP): SIB'ler ve KP'ler arasındaki iletişim için standartlaştırılmış bir mesajlaşma protokolüdür. Bu protokol kullanılan ağ türünden bağımsızdır (GPRS, 3G, WIFI, Bluetooth, HFC, Zigbee). Protokolün iki tür uygulaması vardır: SSAP-XML, daha fazla bant genişliği olduğunda XML formatını kullanır; SSAP-JSON, mobil cihazlar, web tarayıcıları vb. ile iletişim kurarken mesajlar JSON formatına göre uyarlanmaktadır.

Şekil 2.9'da yukarıda sunulan dört kavramın nasıl etkileşime girdiğini göstermektedir. Şekilde farklı KP türlerinin SSAP protokolünü kullanarak SIB'ler aracılığıyla Akıllı Alan ile etkileşime girdiğini görülebilmektedir. KP'ler bilgi almak ya da eklemek için birden fazla Akıllı Alan bağlanabilmektedir. Böylece akıllı alanları diğer akıllı alanlara iletişim kurulabilmektedir. Bu şekilden SOFIA'nın KP'ler tarafından temsil

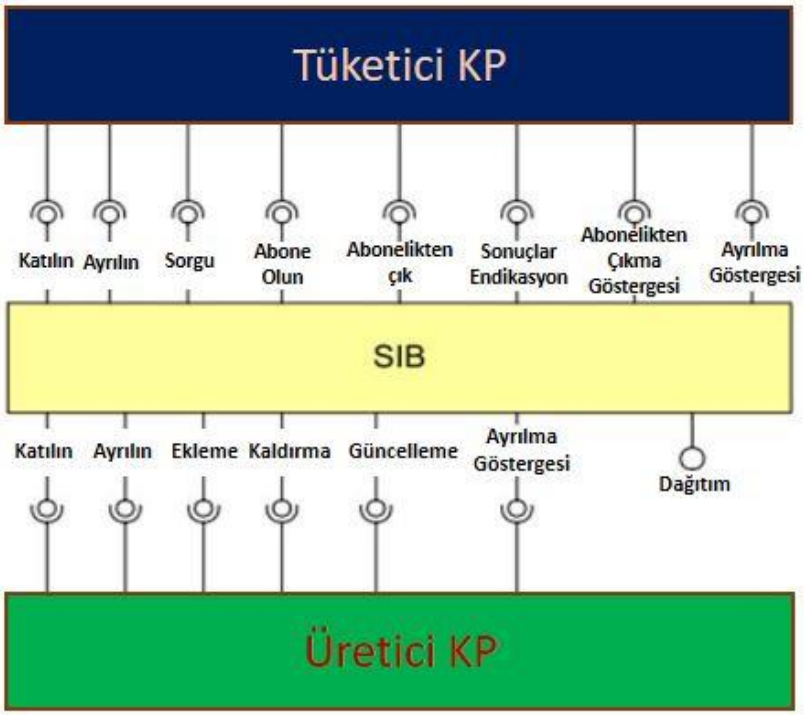
edilen fiziksel dünya ile Akıllı Alan tarafından temsil edilen bilgi dünyası arasında bir bağlantı kurulmasını sağlayan önerisini açıkça görülebilmektedir.



Şekil 2.9. SOFIA'nın ana kavramları.

Şekil 2.10’da, bir KP ile SIB arasındaki iletişim için protokol tarafından tanımlanan bazı işlemleri göstermektedir. Şekilde SIB ile etkileşime giren iki tür KP olduğunu görülebilmektedir. Tüketici KP, SIB'den bilgi almak için protokol tarafından tanımlanan işlemleri gerçekleştirmektedir. Örneğin Katılın işlemi, KP'nin SIB'ye bağlanması için kullanılır (kimlik doğrulama, yetkilendirme ve Akıllı Alan’da bir oturum oluşturulmasını gerektirir). KP SIB ile bağlantısını kesmek istediğinde Ayrılın işlemi kullanılır, KP SIB'den bilgi almak istediğinde Sorgu işlemi kullanılır, bu bilgiler geçmiş verileri veya gerçek zamanlı verileri içeren bir veri tabanından almaktadır.

Üretici KP ise SIB'ye bilgi eklemek için işlemler gerçekleştirmektedir. Şekil 2.10'da görebileceğimiz gibi, tüketici KP ile Katılın ve Ayrılın işlemlerini gerçekleştirmekte ve şekildeki Ekleme, Kaldirma ve Güncelleme işlemleri ile temsil edilen SIB'ye veri eklemek, güncellemek ve kaldırmak için belirli işlemleri gerçekleştirmektedir.



Şekil 2.10. SSAP operasyonları.

Pires vd. [80] çalışmasında sunulan akıllı şehirler için ara yazılım gereksinimlerinden SOFIA, bağlam bilimi gereksinimi hariç hepsini karşılamaktadır. Platform bu gereksinimi doğrudan desteklemese de, çalışma [94] bazı SOFIA bileşenlerini kullanarak bağlam bilimi için bir mikro mimari önermiştir.

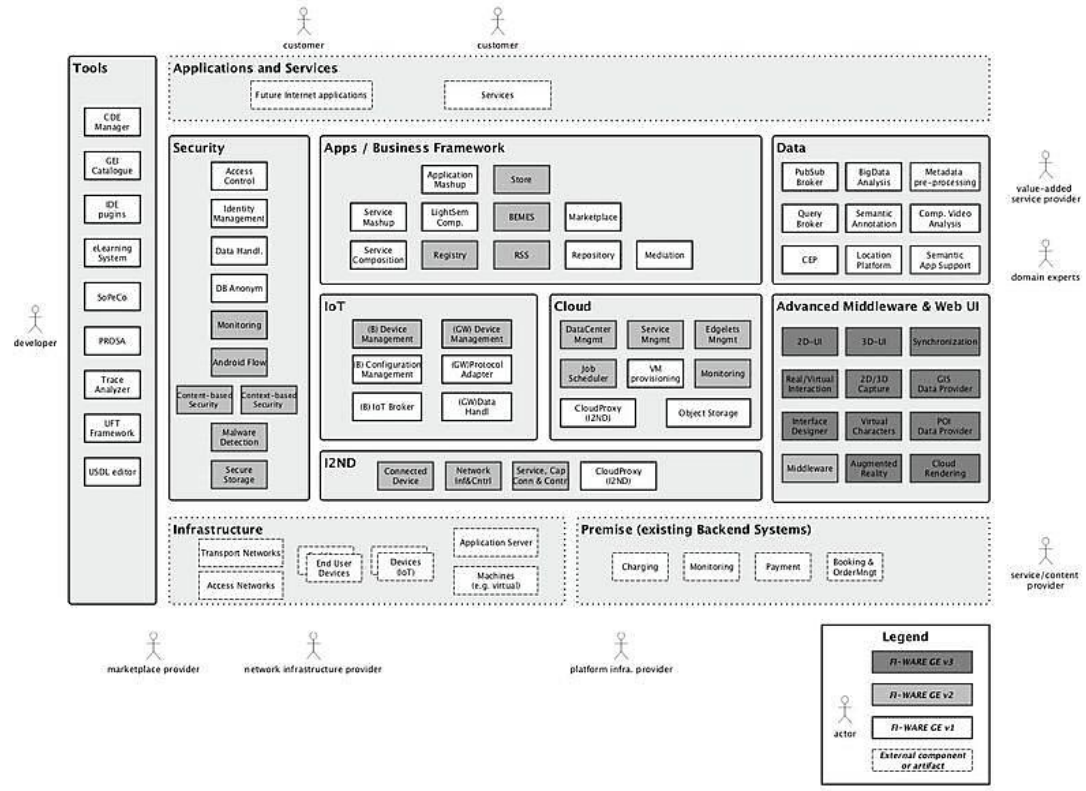
2.2.3. FIWARE

Geleceğin İnterneti (FI), gelecekteki İnternet'in mimarisini belirlemeyi hedefleyen araştırmaları bir araya getirmektedir [95, 96]. Fiziksel altyapı, kablosuz ağlar, mobil ağlar, ağ protokolleri gibi teknolojik konuların yanı sıra iş modelleri ve akıllı uygulamalar/cihazlar (her yerde bulunan bilişim) ile ilgili konuları kapsamaktadır. Avrupa Birliği'nin bir yürütme organı olan Avrupa Komisyonu, FI'nın geleceğiyle ilgili zorlukların farkında olarak Geleceğin İnterneti Kamu-Özel Ortaklığı'nı (FI-PPP) başlatmıştır. FI-PPP kapsamında, FI için önemli bir platform oluşturmayı amaçlayan Geleceğin İnternet Ware (FI-WARE) projesi geliştirilmiştir [97]. FI-WARE platformu, kısaca FIWARE olarak adlandırılmaktadır.

FIWARE platformu, geleceğin interneti için teknolojiler geliştirmek amacıyla oluşturulmuştur [98]. Günümüz internetinin hızlı büyümesi, ağ iletişim teknolojilerindeki (fiber optik, kablosuz ağlar, 3G, 4G ve 5G gibi), yeni bilgisayarlar ve mobil cihazlardaki ilerlemeler sayesinde gerçekleşmiştir. Bu yüksek ve hızlı gelişme aynı zamanda e-ticaretin, sosyal ağların, gerçek zamanlı uygulamaların, bulut bilişiminin ve IoT'nin ortaya çıkmasını sağlamıştır; sadece web sayfalarının değil aynı zamanda web medyasının, web hizmetlerinin, uygulamaların vb. tüketimini de içermiştir. Bu astronomik artışın bir sonucu olarak, şu anda internette kullanılan teknolojilerin kaynakları tükenmektedir, dolayısıyla mevcut panoramayı değişmesi gerekmekte ve yeni teknolojilerin geliştirilmesi için bir fırsat yaratmaktadır [99, 100].

FIWARE platformunun temel amacı, API'ler aracılığıyla erişilebilen ve GE'ler olarak adlandırılan bileşenlerde uygulanan bir dizi açık spesifikasyon aracılığıyla FI hizmetleri için genişletilebilir ve genel bir platform sunmaktır [101]. GE'lerin geliştirilme amacı, FIWARE platformunun çekirdeğini oluşturur ve resmi olarak GE'ler, FIWARE'in işlevsel yapısının temel yapı taşlarını oluşturmaktadır [102]. Her bir GE'nin uygulamaları, API'ler aracılığıyla sağlanan belirli bir işlevsellik kümesini birlikte destekleyen ve böylece her bir GE için tanımlanan açık spesifikasyonlara uygun olarak birlikte çalışabilir arayüzler sağlayan bir dizi bileşen tarafından oluşturulmaktadır. Bir GE'nin spesifikasyonu, uyumlu ve diğer GE'lerle çalışabilen ürünlerin geliştirilmesi için gerekli olan tüm bilgileri içermektedir [103] ve FIWARE tarafından geliştirilen bir GE'nin referans uygulamasını değiştirme olasılığını kapsamaktadır.

GE'ler, FIWARE tarafından sağlanan özel işlevlerle ilgili teknik bölümlerde toplanmıştır. Örneğin, güvenlik, altyapı vb. yapılardır. Şu anda FIWARE spesifikasyonuyla ilgili yedi teknik bölüm bulunmaktadır [104]: Bulut Barındırma (Cloud Hosting), Veri/Bağlam Yönetimi (Data/Context Management), IoT Hizmetlerinin Etkinleştirilmesi (IoT Services Enablement), Uygulamalar, Hizmetler ve Veri Dağıtımı (Applications, Services and Data Delivery), Veri Güvenliği (Security of data), Ağlara ve Cihazlara Arayüz (I2ND) ve Gelişmiş Web Tabanlı Kullanıcı Arayüzü (Advanced Web-based User Interface). Teknik bölümler aynı zamanda Şekil 2.11'de gösterilen FIWARE mimarisinin ana hatlarını belirleyen unsurlardır.



Şekil 2.11. FIWARE platformunun tüm ana bölümleriyle birlikte şematik gösterimi [104].

FIWARE platformunun şematik gösterimi, teknik bölümleri (çerçeveler: Security, I2ND, IoT, Cloud, Apps/Business Framework, Data, Advanced Middleware & Web UI) ve FIWARE'in GE'leri için geliştirmeyi destekleyen araçları (Tools) temsil eden tam kenarlı çerçevelerden oluşan bir ızgara gibidir. Bitişik noktalı kenar çerçeveleri, FIWARE tarafından kullanılan (Infrastructure ve Premise) veya FIWARE'i destek olarak kullanan (Applications and Services) harici unsurları temsil etmektedir.

FI-PPP'nin geniş kapsamlı mimarisi, açık bileşenleri ve yüksek yatırım seviyesi, FIWARE'in uluslararası bir erişimle büyüyen bir faaliyet alanına sahip olmasını sağlamıştır. Ayrıca, GE'lerin geniş çeşitliliği ve farklı alanlar tarafından kullanılabilen unsurlar sağlama odaklanması, FIWARE'in kıtalararası seviyelere yayılmasını sağlamıştır.

FIWARE'in halihazırda elde ettiği başarının yanı sıra, Açık ve Çevik Akıllı Şehirler (OASC) girişimi Finlandiya, Danimarka, Belçika, Portekiz, İtalya, İspanya ve Brezilya'daki 31 şehir tarafından imzalanmıştır [105]. OASC'nin amacı, şehirleri standart bir açık lisans API'sini benimsemeye teşvik etmek ve bunu FIWARE platformunda uygulamak suretiyle, herhangi bir zamanda şehirde neler olup bittiğini açıklayan bağlamsal bilgilerin toplanması, yayınlanması, danışılması ve bunlara abone olunması için hafif ve basit bir araç sağlamaktır. Bu şekilde, FIWARE'in akıllı şehirler için ara yazılımı olarak kullanılması tamamen teşvik edilmekte ve bazı durumlarda olduğu gibi halihazırda belirgin bir şekilde ortaya çıkmaktadır. FIWARE'in akıllı şehirler için ara yazılımı olarak benimsenmesiyle son derece ilgili olan bir diğer nokta da gerekli birkaç gereksinimi karşılamasıdır.

Birlikte Çalışabilirlik (Interoperability), Cihaz Keşfi ve Yönetimi (Device Discovery and Management), Dinamik Uyarlama (Dynamic Adaptation) ve Bağlam Bilimi (Context Science) gereksinimleri Veri/Bağlam Yönetimi ve IoT Hizmetlerinin Etkinleştirilmesi bölümlerinin GE'leri tarafından karşılanmaktadır. Özellikle, birlikte çalışabilirlik, Keşif ve Cihaz Yönetimi gereksinimleri DeviceManagement ve ProtocolAdapter GE'leri tarafından karşılanmaktadır. Dinamik Uyarlama IoTDiscovery'nin sorumluluğundadır ve Bağlam Bilimi IoTBroker, Publish/Subscribe Broker ve Karmaşık Olay İşleme (CEP) tarafından kapsanmaktadır.

Ölçeklenebilirlik (Scalability), Bulut Barındırma bölümü tarafından, özellikle de GE'ler: IaaS ve Cloud Application Management Service tarafından kapsanmaktadır. Bir gereklilikle doğrudan ilgili bir diğer bölüm ise Güvenlik (Security). Büyük Veri İşleme (Processing Large Volumes of Data) ve Veri Yönetimi (Data Management) gereksinimleri, Veri/Bağlam Yönetimi ve Uygulamalar, Hizmetler ve Veri Dağıtımı bölümlerinden BigData Analysis, Query broker ve DataVisualization tarafından karşılanmaktadır.

Günümüzde sistem entegrasyonu probleme yönelik pek çok çözüm bulunmaktadır. Google, Microsoft, Amazon veya IBM gibi büyük şirketler veri depolama, sunucular, Big Data ve hatta IoT çözümleri için araçlar ve hizmetler sunmaktadır [106]. Bu çözümlerin çoğu geneldir ve her problemin ihtiyaçlarına göre özelleştirilebilir, ancak

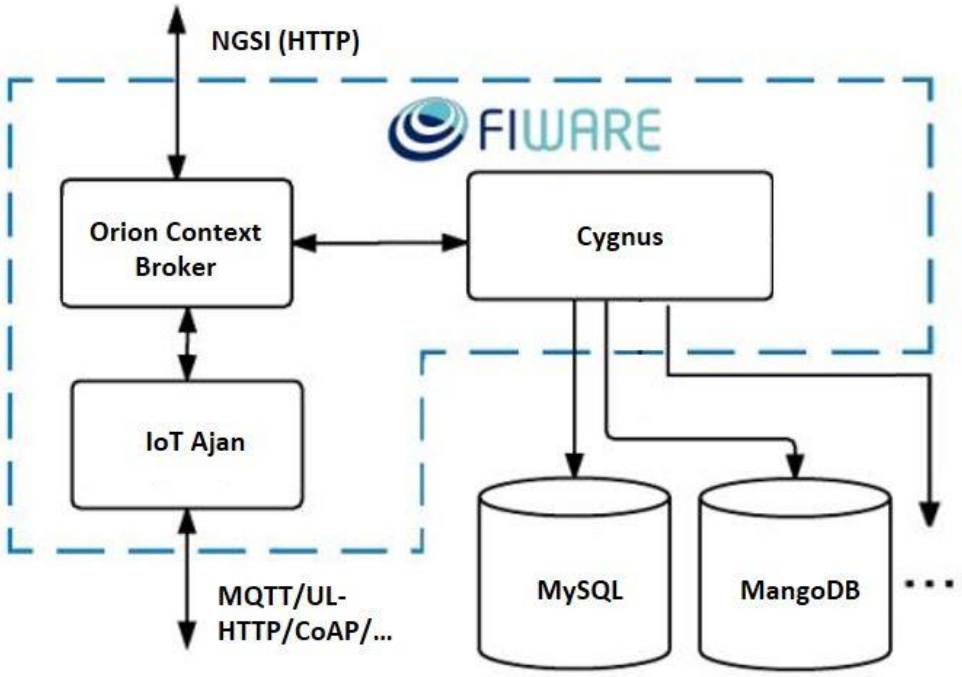
soruna özel hizmetler için çok az olanak vardır. Dahası, bu şirketler tarafından sunulan ücretsiz kaynaklar oldukça sınırlıdır ve bu kaynakların kullanımına da kota uygulamaktadırlar. Sistem büyüdükçe maliyet de artacağı için bu bir problem haline gelmektedir.

Ölçeklenebilirlik, IoT sistemlerinin temel direklerinden biridir [107], çünkü herhangi bir nesneyi bağlayarak önemli veriler sağlayabilir, bu da bu sistemlerin büyümesinin kaçınılmaz olduğu anlamına gelmektedir. Bu noktaları göz önünde bulundurarak, bu çalışma, "çeşitli sektörlerde akıllı uygulamaların geliştirilmesini kolaylaştıran bir dizi basit ve güçlü API sağlayan" FIWARE platformunu tercih etmektedir. Bu API'lerin açıklamaları halka açık ve ücretsizdir, ayrıca proje açık kaynaklıdır, bu da farklı projelerden geliştiricilerin yeni işlevler veya düzeltmeler yapmasına izin vermektedir.

2.2.3.1. FIWARE Ana Bileşenleri

FIWARE, farklı alanlardaki akıllı uygulamalar için yapı taşlarını oluşturan GE adı verilen bileşenleri belirtmektedir. Bu GE'ler, Güvenlik, Veri Yönetimi, Bulut Barındırma ve Cihaz Yönetimi dahil olmak üzere ancak bunlarla sınırlı olmayan hususları kapsamaktadır. FIWARE, akıllı şehirler, akıllı endüstri, akıllı tarım ve akıllı enerji gibi farklı alanlar için akıllı çözümlerin geliştirilmesini kolaylaştıran bağlam veri yönetimi için evrensel bir dizi standart tanımlamaktadır.

FIWARE, bağlam bilgilerinin yönetilmesini, bu bilgilerin işlenmesini ve dış aktörlerin bilgilendirilmesini mümkün kılarak, onların harekete geçmesine ve dolayısıyla mevcut bağlamı değiştirmesine veya iyileştirmesine olanak tanınmaktadır. Context Broker platformun ana bileşenidir; FIWARE Orion Context Broker'ın aksine, tamamlayıcı FIWARE GE'lerin kullanımı zorunlu değildir, çünkü Orion GE ve üçüncü taraf bileşenlerle "Powered by FIWARE" çözümü geliştirmek mümkündür [108]. Bu çalışmada üç FIWARE bileşeni kullanılmıştır: Orion Context Broker, IoT Ajanı ve Cygnus. Şekil 2.12'de bu tez çalışmasında kullanılan FIWARE'in ana bileşenleri bir diyagramla gösterilmektedir.



Şekil 2.12. Fiware mimari şeması.

A. FIWARE Orion Context Broker

Orion Context Broker, GE'nin Publish/Subscribe Context Broker'ının bir uygulamasıdır ve bir Sonraki Nesil Hizmet Arayüzleri (NGSI) sağlamaktadır [109]. Orion Context Broker, güncellemeler, sorgular, kayıtlar ve abonelikler dahil olmak üzere bağlam bilgilerinin tüm yaşam döngüsünü yönetmenize olanak tanımaktadır [110]. Bağlam bilgilerini ve kullanılabilirliğini yönetmek için bir NGSI sunucu uygulamasıdır. Orion Context Broker'ın kullanılması, bir uygulamanın "Powered by FIWARE" olarak nitelendirilmesi için yeterlidir. Müşteriler NGSI arayüzünü kullanarak çeşitli işlemler gerçekleştirebilir:

1. Orion Context Broker, uygulamalar için güncel bağlam bilgilerini depolar; böylece sorgular bu bilgilere dayalı olarak çözülebilmektedir. Bağlam bilgileri, varlıklardan (örneğin bir araba) ve bunların özniteliklerinden (örneğin arabanın hızı veya konumu) oluşmaktadır. Bağlam bilgilerini güncelleme işlemi, örneğin sıcaklık güncellemeleri gönderme gibi yöntemlerle gerçekleştirilebilmektedir.

2. Bir bağlam bilgisinde değişiklik meydana geldiğinde (örneğin, sıcaklık değiştiğinde), bildirim alınır
3. Bağlam sağlayıcı uygulamaları kaydetme, örneğin bir odanın içindeki sıcaklık sensörünün sağlayıcısıdır.

B. FIWARE IoT Ajanı (IoT Agent)

IoT Ajanı, bir cihaz grubunun verilerini iletmek ve kendi yerel protokollerini kullanarak bir Context Broker'dan yönetilmek üzere tasarlanmış bir bileşendir [111]. Her IoT Ajanı, tek bir yük biçimi için tanımlanmış olsa da, bu yükü iletmek için birkaç farklı iletişim protokolü kullanabilir. Ayrıca, IoT Ajanı, FIWARE platformunun güvenlik yönlerini (kimlik doğrulama ve kanal yetkilendirme) ele alabilmeli ve cihaz geliştiricisine diğer ortak hizmetleri sağlayabilmelidir.

C. FIWARE Cygnus

Cygnus, FIWARE platformundaki mevcut verilerin kalıcı hale getirilmesi için bir bileşendir [111]. Bu bileşen, Apache Flume'u kullanarak üçüncü taraf veri tabanlarına bağlam verilerini aktarır ve bağlamın geçmiş bir görünümünü oluşturmaktadır.

2.2.4. Ara Yazılım Platformları vs Gereksinimler

Pires vd. [80]'da Akıllı Şehirler için bazı ara yazılım gereksinimleri sunulmuştur: Birlikte Çalışabilirlik, Cihaz Keşfi ve Yönetimi, Dinamik Adaptasyon, Bağlam Bilimi, Ölçeklenebilirlik, Büyük Hacimli Verilerin İşlenmesi, Güvenlik, Veri Yönetimi ve Geliştirme Araçları. Aşağıdaki Tablo 2.1, yukarıda belirtilen platformların her biri tarafından karşılanan gereksinimleri ortaya koymaktadır. Burada ✓ sembolü gereksinimin tamamen karşılandığını, ○ sembolü gereksinimin kısmen karşılandığını ve ✖ sembolü gereksinimin karşılanmadığını göstermektedir.

Çizelge 2.1. Akıllı şehirler için gereksinimler ve ara yazılım platformlarının karşılaştırmalı tablosu.

		ARA YAZILIM PLATFORMLARI		
		CityHub	FIWARE	SOFIA
GEREKSİNİMLER	Birlikte Çalışabilirlik	✓	✓	✓
	Cihaz Keşfi ve Yönetimi	○	✓	✓
	Dinamik Adaptasyon	○	✓	✓
	Bağlam Bilimi	○	✓	x
	Ölçeklenebilirlik	✓	✓	✓
	Büyük Hacimli Verilerin İşlenmesi	✓	✓	✓
	Güvenlik	x	✓	✓
	Veri Yönetimi	✓	✓	✓
	Geliştirme Araçları	✓	✓	✓

2.3. VERİ GÖRSELLEŞTİRME: GRAFANA

Dijital bir toplumda yaşadığımız için, veri birçok bilgi alanında mevcut ve çeşitli teknolojiler tarafından kullanılmaktadır. Medya, sosyal ağlar veya birçok şirket tarafından kullanılsın, bu veriler son derece önemlidir ve analiz edilmelidir. Günümüzde, veri madenciliği, istatistiksel analiz ve makine öğrenimi gibi teknikler, verileri analiz etmek için bulunmaktadır. Bu tekniklerden biri de, bir veri örneğinden elde edilen önemli ve gerekli verilerin açık ve öz bir şekilde sunulmasını mümkün kılan veri görselleştirmedir [44].

Veri görselleştirme, çoğunlukla grafiklerle temsil edilen, pasta grafikler, çizgi grafikler, çubuk grafikler, alan grafikleri vb. gibi nicel verilerin görsel temsilleriyle ilgilidir. Özellikle, soyut verileri görsel modellere dönüştürmek için bilgisayar teknolojilerini kullanma sürecidir [112]. Veri görselleştirmede, bir sunumun görsel unsuru, bir karar alma sürecindeki tüm tarafların kendilerine sunulan bilgileri anlamalarını ve doğru bir şekilde yorumlayabilmelerini sağlamak için tasarlanmıştır.

Birçok durumda, veri görselleştirme bilgiyi anlamanın en iyi yoludur; diğerleri için ise veri görselleştirme sadece sözlü veya yazılı olarak sunulan gerçekleri bağlam

sallaştırmaya yardımcı olmaktadır. Herhangi bir veri görselleştirmesi için, görselleştirme grafiklerini oluşturacak bir araca ihtiyacınız olacaktır. Bu tür veri işleme için özel dillerin yanı sıra veri görselleştirme için kütüphaneler de bulunmaktadır. Veri görselleştirmede kullanılan çok sayıda kütüphane veya uygulama bulunmaktadır. Verileri kolayca anlamak ve gösterge tablolarında (dashboards) temsil etmek için birçok görselleştirme aracı mevcuttur. Şu anda, Grafana gibi araçlar işgücü piyasasında yaygın olarak kullanılmakta ve şirket yöneticilerinin dikkatini çekmektedir.

Hava durumu verileri, eylemler, web sitelerinden gelen istekler ve giden yanıtlar gibi canlı yayın verilerini görselleştirmek için son derece etkili ve verimli bir araç olan Grafana aracı bu çalışmada kullanılmaktadır. Grafana, açık kaynaklı ve çok platformlu bir veri analizi ve görselleştirme aracıdır [8]. Metriklerin görselleştirilmesine, sorgulanmasına, uyarılmasına ve keşfedilmesine yardımcı olmaktadır. Grafana, gerçek zamanlı olarak iletilen verileri doğrudan gösterge tablosunda kesin sonuçlarla verimli bir şekilde görselleştiren en etkili araçlardan biridir. Günümüzde ağ izleme, gözle görülür bir şekilde yükselişte ve bu yönde çaba sarf eden şirketler arasında giderek daha popüler hale gelmektedir. Zabbix, ağ yöneticilerine bu konudaki görevlerinde büyük destek sağlamıştır [113]. Yerel olarak sağladığı sayısız faydaya ek olarak, daha güzel, daha basit ve daha etkileşimli arayüzler sunmak için hala "daha fazlasını" arayan birçok kişi bulunmaktadır.

Mehdi vd. [114]'e göre Grafana, çeşitli uzmanlık alanlarından üst düzey yöneticileri büyüleyen grafikleri aracılığıyla veri görselleştirmede büyük yardım sağlayan bir araç olarak gelecek vaat etmektedir. Ücretsiz ve açık kaynaklı olmasının yanı sıra, hafif ve çoklu veri kaynağı yapısı sayesinde kullanıcılara MySQL, PostgreSQL ve InfluxDB gibi popüler seçenekler de dahil olmak üzere geniş bir veri tabanı entegrasyonu sunarak çok yönlülük sağlanmaktadır [115]. Bu özellik, farklı toplama kaynaklarından gelen çeşitli verilerle görsel gösterge tabloları oluşturmanın mümkün olduğu anlamına gelmektedir. Örneğin, telefon, sohbet, çağrı aracı ve izleme aracı aracılığıyla elde edilen ana hizmet metrikleri hakkında bilgi içeren tek bir gösterge tablosu, bir destek merkezini izlemek için kullanılabilmektedir.

Grafana'nın önemli bir avantajı, açık kaynak kodlu olması nedeniyle topluluğun gelişimine aktif katılımı ve belgeleme süreçlerine yoğun ilgi göstermesidir. Ayrıca, dünya genelinde binlerce aktif kullanıcıya sahip olan canlı bir foruma sahip olması da dikkatleri üzerine çekmektedir. Bu nedenle, uygulama ile ilgili herhangi bir işletme veya bakım problemi ortaya çıktığında, yardım almak oldukça kolaydır. Şekil 2.13'te Grafana'nın görsel bir panosu görülmektedir.

Şekil 2.13. Bu çalışmada sıcaklık ve nem görsel bir panosu.

2.4. DOCKER

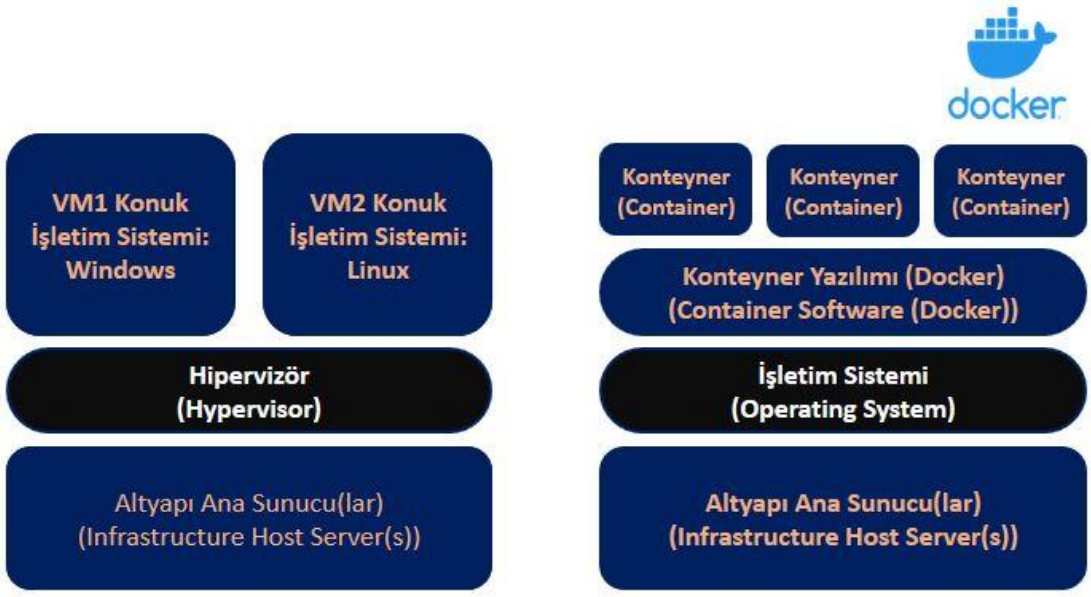
Reis vd. [116]'e göre Docker, uygulamaların yazılım konteynerlerinde otomatik dağıtımını sağlayarak, ek bir soyutlama katmanı sağlayan ve çeşitli işletim sistemlerinde uygulamaların sanallaştırılmasını otomatikleştiren açık kaynaklı bir projedir. Docker, işletim sistemini etkilemeden programları çalıştırmak için izole edilmiş bir ortam "emülatörü" olarak işlev görür ve diğer işletim sistemlerine veya ortamlara taşınabilir ve çoğaltılabilmektedir. Docker, imajlardan ve konteynerlerden oluşmaktadır:

1. İmaj, atıl ve değişmez bir spesifikasyondur; çalıştırmak istediğimiz uygulamadan kütüphanelere ve onu içeren işletim sisteminin üzerinde çalışması için gereken her şeyi içeren durumun ve yazılım parçalarının anlık görüntüsüdür.

2. Bir konteyner, yapılandırılabilen bir imajın örneklendiği ve izole edildiği bir ortamdır.

İlgili konteynerlerden oluşan uygulamaları başlatmak için Docker-compose[117] aracını kullanmak gerekmektedir.

Literatürde sanal makinelerle konteynerler, özellikle Docker konteynerleri arasında birçok performans karşılaştırması rapor edilmiştir. Çoğu karşılaştırmada, Docker konteynerlerinin performansının sanal makinelerden daha üstün olduğu bildirilmiştir. Konteynerlerin kullanılmasıyla konuşlandırılan hizmetlerin yürütülmesinin daha az zaman alması beklenir, bu da [118]'te olduğu gibi sanal makinelerde daha az gecikmeye neden olmaktadır. Buna ek olarak, konteynerlerin sanallaştırılmış teknolojilere kıyasla daha düşük enerji tüketimine sahip olduğu gösterilmiştir [119]. Şekil 2.14'te, sanal makinelerle Docker konteyner teknolojisi arasındaki mimari yaklaşımdaki farkı göstermektedir.



Şekil 2.14. VM ve Docker konteyner mimari yaklaşımdaki fark.

Konteynerler ve VM'ler aynı kaynak tahsisini ve izolasyonunu kullanmaktadır. Ancak konteynerler, OS çekirdeğinin diğer konteynerlerle paylaşılmasına izin verdiği için farklı bir uygulama kullanır ve her biri yalıtılmış bir süreç olarak birkaç konteynerin

çalıştırılmasına izin verilmektedir. VM'ler, CPU, bellek, depolama ve ağ gibi tüm donanımı sanallaştırarak bağımsız bir makine oluşturur. Bu, VM'ler arasında yüksek bir izolasyon seviyesine neden olurken, sanallaştırma aşırı yükü nedeniyle daha fazla sistem kaynağı tüketebilmektedir.

2.5. İLGİLİ ÇALIŞMALAR

Günümüzde, IoT kavramına dayalı sistemlerin oluşturulması için birçok çalışmalar yapılmaktadır. Bu bölümde, sistemler incelenerek yapılan çalışmaların katkıları ve yapılabilecek çalışmalar analiz etmiştir. Çizelge 2.2' de ilgili çalışmalar dört ana unsur bakımından bir karşılaştırma sunmaktadır: Platform, Veri toplama, Veri kalıcılığı ve Veri görselleştirme. Platform sütunu, bu çalışma ile ilgili çalışmalarda kullanılan ara yazılım platformunu gösterir; bu durumda araştırmanın odak noktası FIWARE platformudur. Veri toplama sütunu, sensörlerden veri toplamak için kullanılan Broker, mikrodenetleyici, IoT geliştirme kartının veya cihazlar türünü gösterir. Geçmiş veri kalıcılığı sütunu, Orion Context Broker'dan veri almaktan ve bunları harici depolama sistemlerinde kalıcı hale getirmekten sorumlu bileşeni ve geçmiş verileri depolamak için kullanılan veri tabanını tanımlamaktadır. Son olarak, Veri görselleştirme sütunu, geçmiş verileri görselleştirmek için kullanılan aracı göstermektedir.

Çizelge 2.2. Araştırma konusu ile ilgili çalışmalar.

Yazarlar	Açıklama	Platform	Veri Toplama	Geçmiş Veriler Kalıcılığı	Veri Görselleştirme
J.A. LÓPEZ-RIQUELME, N. PAVÓN-PULIDO, H. NAVARRO-HELLÍN, F. SOTO-VALLES, AND R. TORRES-	Yapılan çalışmalarda hassas tarım için FIWARE ara yazılımı kullanılmakta ve bu IoT platformundan GE'lerle çalışma	FIWARE	"FIWARE Smart Agricultural Node" adı verilen otonom bir cihaz geliştirildi.	Cygnus Enjektör ile Cosmos	Widget Fispace

SÁNCHEZ (2017) [32]	seçeneği vurgulanmaktadır.				
PAU ARCE, DAVID SALVO, GEMA PIÑERO, ALBERTO GONZALEZ (2021) [120]	Kentsel ortamları izleyen bir kablosuz akustik sensör ağı (WASN) sunuldu ve kentsel ses olaylarını sınıflandırmak için gereken tüm süreci, ses sınıflandırmasına kadar evrişimli sinir ağı kullanarak gerçekleştirdi.	FIWARE	Raspberry Pi 3 Model B	Quantum Leap ile CrateDB	Grafana
RENATO OLIVEIRA AND CARLOS A. KAMIENSKI (2022) [121]	Bağlam yönetimini kullanan bir platform olan FIWARE'e dayalı bir mimari olan IoT Redirector adlı bir yazılım çözümü geliştirilmiştir.	FIWARE	MQTT Broker	Quantum Leap ile CrateDB	Yok
J. FERNANDES, J. SA SILVA, A. RODRIGUES, F. BOAVIDA, R. GASPAR, C. GODINHO VE R. FRANCISCO (2022) [122]	SarsCov-2 pandemisi ve gelecekteki olası pandemiler bağlamında kullanılabilecek bir akıllı telefon ve akıllı saat uygulaması aracılığıyla insanların fiziksel ve duygusal durumlarını izlemek için	FIWARE	smartphone ve smartwatch	Cygnus ile STH-Comet, MySQL ve MongoDB	Vitoria Uygulaması

	yenilikçi bir çözüm sunuldu.				
DIOGO ALEXANDRE RODRIGUES LOPES (2018) [108]	Kablosuz bir ortamda IoT cihazları ile iletişim kurabilen, bunları kontrol edebilen ve veri toplayabilen evrensel bir modüler IoT sistemi geliştirmek için FIWARE platformunun ve teknolojilerinin nasıl kullanılacağı gösterildi.	FIWARE	NodeMcu Devkit v1.0	Cygnus ile STH-Comet ve MongoDB	MongoDB Compass
ABDELKADER SALLEMINE (2019) [33]	FIWARE adı verilen bağlamsal bir bilgi yönetimi çerçevesinden yararlanarak hassas tarımda su kaynakları yönetimine yönelik yeni bir yaklaşım ortaya koymaktadır. Veri görselleştirme için Grafana aracını kullanmıştır.	FIWARE	Adafruit Metro 328	Quantum Leap ile CrateDB	Grafana
G. VAGLICA, F. BONO, G. RENALDI (2020) [123]	Politecnico di Torino'da, heterojen sensörlerin ve kablosuz ağların gelecekteki entegrasyonu için Ortak Araştırma	FIWARE	WSDA Base 101 LXRS Data Gateway	Cygnus ile PostgreSQL	Grafana

	Merkezi (JRC)'nin E.4 Ünitesinde bir FIWARE testbed uygulamak amacıyla platformun performansını ve kısıtlamalarını değerlendirmek için gerçekleştirilmiştir.				
T. STOREK, J. LOHMÖLLER, A. KÜMPEL, M. BARANSKI AND D. MÜLLER (2019) [124]	Akıllı bina enerji yönetim sistemleri (BEMS) için bulut teknolojilerindeki en son gelişmelerden yararlanmak amacıyla, özelleştirilebilir bir FIWARE açık kaynak bulut platformu ve mevcut BEMS ile platform arasında iletişim için bir ağ geçidi sunmuşlardır.	FIWARE	BACnet cihazları	Quantum Leap ile CrateDB	Grafana
TEZ ÇALIŞMASININ ÖNERİSİ	Algılamadan veri görselleştirmeye kadar olan süreci kapsayan, geçmiş verilerin kalıcılığına ve görselleştirilmesine yönelik IoT tabanlı bir yaklaşım	FIWARE	Raspberry Pi 3 Model B	Cygnus ile MySQL	Grafana

	benimsenmiştir. Sistem, veri kalıcılığı ve görselleştirme için düşük maliyetli bir cihaz ile ücretsiz ve açık kaynaklı teknolojileri kullanmaktadır.

Bu tez çalışmasında, ele alınan IoT kavramlarını kullananlar da dahil olmak üzere, FIWARE platformunu kullanarak sistem geliştiren çeşitli çalışmalar olduğu görülmektedir. Bu bağlamda, bu tür sistemlere özgü birçok faktörün dikkate alınması gerekmektedir. Örneğin, iletişim protokolleri, veri toplama cihazları, veri kalıcılığı ve veri görselleştirme gibi konulardır. Elde edilen bu sistemler, tez önerisinde olduğu gibi algıdan veri görselleştirmeye kadar olan süreci kapsayan sensörler, cihazlar, FIWARE platformu, veri tabanları ve veri görselleştirme bileşenleri kullanarak çözümler sunmaktadır. Ancak, çeşitli işletim sistemlerini destekleyen ve diğer IoT cihazları ve sunucularıyla iletişimi kolaylaştıran Ethernet ve Wi-Fi gibi ağ bağlantısı için yerleşik desteğe sahip düşük maliyetli bir cihaz kullanmanın ve ayrıca veri kalıcılığı ve görselleştirme için ücretsiz ve açık kaynaklı teknolojilerle birlikte kullanılmasının önemini ele alan çok az çözüm bulunmaktadır.

Bu çalışma, düşük maliyetli bir cihaz olan Raspberry Pi'yi kullanarak bir yaklaşım sunmaktadır. Aynı zamanda sistemin mimarisinin karmaşıklığını artırmamak ve veri kalıcılığını ile görselleştirmeyi daha basit bir şekilde sağlamak için gerekli olan az sayıda FIWARE bileşenlerini seçip kullanarak tam olarak katkıda bulunmayı amaçlamaktadır. Bununla birlikte, kullanılan veri tabanları ve veri görselleştirme teknolojiler açık kaynak kodlu ve ücretsizdir. Bu durumda, MySQL veri tabanı tercih edilmiş ve geçmiş bağlam verilerini kalıcı hale getirmek için Cygnus kullanılmıştır.

Cygnus, Orion Context Broker ile sorunsuz bir şekilde entegre olacak şekilde özel olarak tasarlanmıştır ve verilerin gerçek zamanlı olarak kalıcı hale getirilmesini

sağlayacak şekilde işlevseldir [125]. Ayrıca, Cygnus, çeşitli harici depolama sistemlerini destekler [123]; bu çalışmada ise MySQL tercih edilmiştir.

MySQL, ücretsiz ve açık kaynak kodlu bir yazılımdır ve geniş bir kullanıcı topluluğuna, çok sayıda dokümantasyona, destek forumlarına ve çevrimiçi kaynaklara sahiptir [126]. Ayrıca, büyük hacimli verileri ve yoğun iş yüklerini idare edebilme kapasitesine sahiptir [127], bu da onu büyük ölçekli veri depolama ve işleme gerektiren IoT projeleri için uygun hale getirmiştir.

Veri görselleştirme için açık kaynak aracı Grafana kullanılmıştır. Grafana, kullanıcıların görsel olarak çekici ve bilgilendirici gösterge tabloları oluşturmasına olanak tanıyan sezgisel ve son derece özelleştirilebilir bir kullanıcı arayüzüne sahiptir. Ayrıca, belirli koşullara göre uyarı kurallarının ayarlanmasına olanak tanıyan entegre uyarı ve bildirim özelliklerine sahiptir. Grafana, çizgi, çubuk, dağılım, histogram, harita ve coğrafi grafikler gibi çok çeşitli görselleştirme seçenekleri sunmaktadır [128]. Ayrıca, ilişkisel veri tabanları (MySQL, PostgreSQL gibi) ve NoSQL veri tabanları (MongoDB, InfluxDB gibi) dahil olmak üzere çok çeşitli veri kaynaklarına destek sağlamaktadır [115].

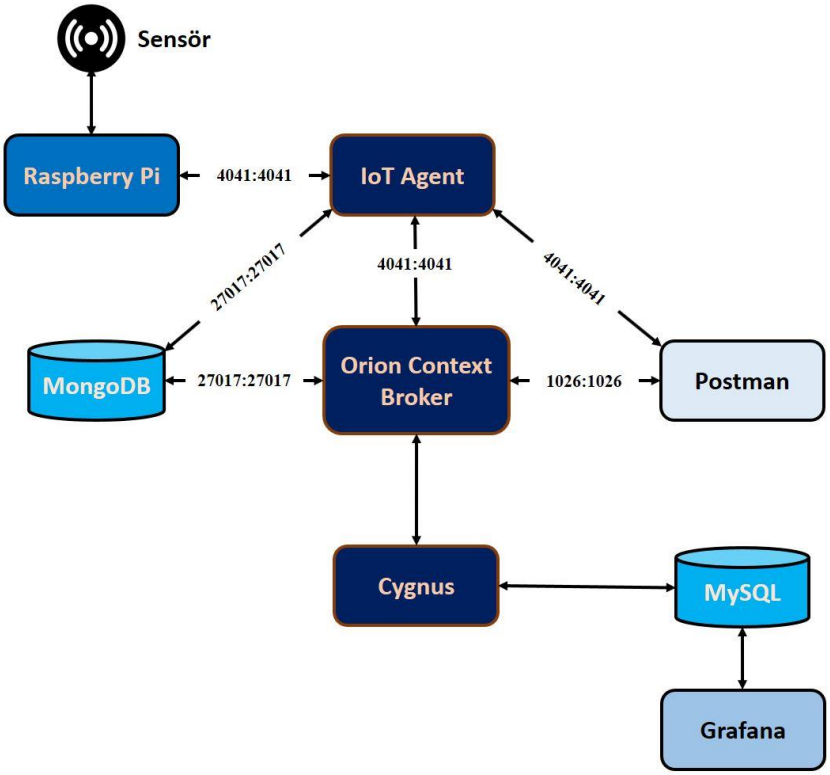
BÖLÜM 3

MATERYAL VE METOD

Bu bölümde, bağlam bilgisi yönetimi yaklaşımından yararlanan, bir sensörden ölçülen verileri ara yazılım platforma göndermek için düşük maliyetli bir cihaz kullanarak, geçmiş verileri kalıcı hale getirmek ve görselleştirmek için açık kaynaklı bir platform üzerinde bir IoT sistemi gerçekleştirmek için benimsenen materyaller, metodolojiler ve stratejiler raporlanmaktadır. Mimari anlatıldıktan sonra donanım ve yazılım bileşenlerinin konfigürasyonları ayrıntılı olarak anlatılmaktadır. Son olarak önerilen mimarinin veri akışını açıklayan bir diyagramı sunulmaktadır.

3.1. MİMARİ TASARIM

Bu Mimari, DHT22'den ölçülen verileri Orion Context Broker'a göndermek için Raspberry Pi kullanılmaktadır. Üç FIWARE bileşeni kullanılmıştır: Orion Context Broker, IoT Ajanı ve geçmiş bağlam verilerini bir veri tabanına kalıcı hale getirmek için Cygnus. Hem Orion Context Broker hem de IoT Ajanı, tuttukları son durumu, bağlam bilgisi saklamak için MongoDB teknolojisine güveniyor ve diğer veri tabanından (bu durumda MySQL) gelen verileri geçmiş bağlamda tutmaktadır. Ayrıca SQL veri tabanındaki verileri görselleştirmek için Grafana bileşenini kullandı. Şekil 3.1’de, gerçekleştirilen IoT prototip mimarisinin tasarımı göstermektedir.



Şekil 3.1. Bu çalışmada gerçekleştirilen IoT prototip mimarisinin tasarımı.

1. Raspberry Pi: Bu bileşen, DHT22 sensörü tarafından ölçülen ortam sıcaklığı ve bağıl nem verilerini toplar ve platforma göndermektedir.

2. FIWARE IoT Ajanı: FIWARE platformu içinde kullanılan NGSI protokolü ile JSON protokolü arasında bir köprü oluşturan bir ajanı ifade etmektedir.

3. FIWARE Orion Context Broker: Bu bileşen, güncellemeler, sorgular, kayıtlar ve abonelikler de dahil olmak üzere bağlam bilgilerinin tüm yaşam döngüsünü yönetmeyi sağlamaktadır.

4. FIWARE Cygnus: Belirli üçüncü taraf veri tabanlarında, bu çalışmada kullanılan MySQL gibi veri kaynaklarını kalıcı hale getirmekten sorumlu bir bağlayıcıdır, böylece bu verilere geçmiş bir bakış sağlamaktadır.

5. MySQL Veri tabanı: Diğer bileşenlerden toplanan zaman serisi verilerini depolayan ilişkisel bir veri tabanıdır.

6. MongoDB Veri tabanı: Verileri belgelere benzer şekilde depolayan bir ilişkisel olmayan veri tabanıdır. Genellikle, Orion Context Broker ve IoT ajanlarında kullanılan varlık yapılarını saklamak ve her bir varlığın özelliğinin son değerini saklamak için kullanmaktadır.

7. Postman : Bu bileşen, IoT ajanında ve Orion Context Broker'da bilgi modelinin yapılandırmasını yapar ve yalnızca platformun başlatılma sürecinde kullanılmaktadır.

8. Grafana: Verilerin analizini ve görselleştirmesini tablolar, grafikler ve uyarılar aracılığıyla sağlayan açık kaynaklı ve çok platformlu bir web uygulamasıdır. Diğer bileşenlerden toplanan verilerin görüntülenmesi için kullanılmaktadır

3.2. ÜÇ JENERİK FIWARE AKTİVATÖRÜ

FIWARE'nin jenerik aktivatörleri, FIWARE ara yazılım platformunun yenilik ve uygulama geliştirmeyi destekleyen temel bileşenleridir. Bu üç yaygın FIWARE aktivatörü:

3.2.1. FIWARE Orion Context Broker

Orion Context Broker, platformun ana bileşenidir ve bağlam bilgilerinin tüm yaşam döngüsünü, güncellemeleri, sorguları, kayıtları ve abonelikleri API'si aracılığıyla ve ilişkisel ve ilişkisel olmayan veri tabanlarıyla bağlantılar üzerinden yönetmek için kullanılmaktadır. Orion Context Broker, bir Docker konteynerinde örneklenebilmektedir. Gerekli yapılandırma aşağıda gösterilmiştir:

```
orion:
   image: fiware/orion:3.7.0
   hostname: orion
   container_name: fiware-orion
   depends_on:
     - mongo-db
   networks:
     - default
```

```
    ports:
      - "1026:1026"
```

1. `image: fiware/orion:3.7.0` , bu görüntüyü Docker Hub'dan yükler.
2. `hostname: orion` , ağda konteyneri bulmayı kolaylaştırmak için bir hostname oluşturur.
3. `container_name: fiware-orion` , bilgi almayı kolaylaştırmak için bir konteyner adı oluşturur.
4. `depends_on - mongo-db` , Orion'dan varlık bilgilerini saklamak için veri tabanı.
5. `Networks - default` , kullanıcılar varsayılan olarak adlandırılan bir yerel ağ.
6. `ports - "1026:1026"`, makinedeki 1026 numaralı bağlantı noktasını konteynerdekiyle eşleştirir.

3.2.2. FIWARE Cygnus

Cygnus, Orion Context Broker'daki verileri MySQL veri tabanında kalıcı hale getirerek, bu verilerin geçmiş bir görünümünü oluşturmaktadır. Cygnus, Orion Context Broker'daki veri değişikliklerini dinlemek için mevcut publish/subscribe mekanizmasını kullanır ve ardından verileri veri tabanına göndermektedir. Cygnus, bir Docker konteynerinde örneklenebilir. Gerekli yapılandırma aşağıda gösterilmiştir:

```
cygnus:
    image: fiware/cygnus-ngsi:2.16.0
    hostname: cygnus
    container_name: fiware-cygnus
    networks:
      - default
    depends_on:
      - mysql-db
    expose:
      - "5050"
      - "5080"
    ports:
      - "5050:5050"
      - "5080:5080"
```

Cygnus konteynerı iki portu dinlemektedir:

1. Cygnus için abonelik `port 5050`, hizmetin Orion bağlam aracısından gelen bildirimleri dinleyeceği yer olarak bilinmektedir.

2. Cygnus için yönetim `port 5080`, Postman'ın aynı ağın bir parçası olmadan provizyon komutları verebilmesi için açığa çıkarılmıştır.

3.2.3. FIWARE IoT Ajanı: JSON

Bu bileşen, JSON protokolünü Orion Context Broker'da kullanılan NGSI protokolüne çevirmektedir. Postman bileşeni tarafından yapılandırma işlemi tamamlandıktan sonra, IoT Ajanı, Raspberry Pi (fiziksel nesne) ile Orion Context Broker'da bir varlık olarak dijital temsilini (dijital nesne) arasında iletişim kurar. NGSI kullanarak Güneye giden istekleri alır ve bunları cihazlar için JSON komutlarına dönüşümü sağlamaktadır. Ayrıca cihazlardan JSON biçiminde Kuzeye giden ölçümleri alır ve bunları bağlam aracısının bağlam varlıklarının durumunu değiştirmesi için NGSI isteklerine dönüştürmektedir. IoT Ajan JSON, bir Docker konteynerinde örneklenebilir. Gerekli yapılandırma aşağıda belirtilmiştir:

```
iot-agent-json:
    image: fiware/iotagent-json
    hostname: fiware-iot-agent-json
    container_name: fiware-iot-agent-json
    depends_on:
      - mongo-db
    networks:
      - default
    expose:
      - "4041"
      - "7896"
    ports:
      - "4041:4041"
      - "7896:7896"
```

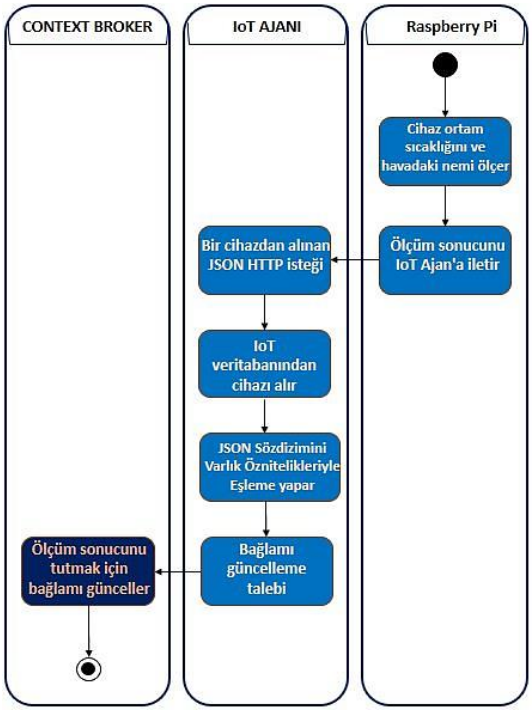
`iot-agent-json` konteyneri Orion Context Broker'ın varlığına bağlıdır ve URL'ler ve cihaz anahtarları gibi cihaz bilgilerini depolamak için bir MongoDB veri tabanı kullanır. Konteyner iki portu dinlemektedir:

1. `port 7896`, Raspberry Pi'dan HTTP ölçümleri üzerinden JSON almak için açığa çıkmaktadır.

2. `port 4041`, Postman'ın aynı ağın bir parçası olmadan provizyon komutları verebilmesi için kullanılmaktadır.

3.2.3.1. Kuzeye Bağlı Trafik (North Bound Traffic) – Ölçüm

Kuzey Bağlantılı Trafik, JSON IoT Ajan'dan geçen ve Orion Context Broker'a ulaşan sensör okumalarından gelen veri trafiğini temsil eder. IoT Ajanı, Orion Context Broker'a ulaşıyor. Şekil 3.2'deki etkinlik diyagramı, Raspberry Pi'dan Orion'a giden verileri açıklamaktadır.



Şekil 3.2. JSON IoT ajanı etkinlik diyagramı - Kuzey trafiği (ölçüm).

Adımlar aşağıda açıklanmıştır:

1. Raspberry Pi, sıcaklık ve nem ölçüm verilerini gönderir ve sonucu IoT Ajanına iletmektedir.

2. IoT Ajanı, bu Northbound isteğini alır, sonucu JSON syntax'ına dönüştürür ve etkileşimin sonucunu bağlama aktararak Context Broker'a bir NGSI isteği yapmaktadır.

3. Context Broker, bu Northbound isteğini almakta ve bağlamı ölçüm sonucuyla güncellemektedir.

3.3. VERİ DEPOLAMA

Bu çalışmada iki veri tabanına ihtiyaç duyulmaktadır. Varlıklar tarafından iletilen verilerin son durumu, bağlam bilgisi olarak ekli MongoDB veri tabanında saklanmaktadır. Ancak, Context Broker'ın geçmiş verileri depolamak için tasarlanmadığı belirtilmesi gerekmektedir. Bu nedenle, geçmiş verileri kalıcı hale getirmek için başka bir FIWARE GE kullanılmıştır. Bu amaçla, temel bir veri tabanı olan MySQL gibi bir veri tabanını kullanan Cygnus GE tercih edilmiştir. Cygnus bileşeni, Context Broker'daki varlıklarla ilgili güncellemelere abone olur ve alınan yeni veriler saklanmaktadır. Bu yöntem, verilere daha sonra erişim, analiz ve temsil için bir kaynak sağlamaktadır.

3.3.1. MongoDB Veri tabanı

Győrödi vd. [129]'e göre MongoDB, belgelere dayalı bir dağıtılmış veri tabanıdır. Bu belgeler, JSON'un ikili bir gösterimi (BSON)'da saklanır. Bu çalışmada, hem Orion Context Broker hem de IoT Ajanı, tuttukları son durumu, bağlam bilgisini saklamak için MongoDB kullanmaktadır. MongoDB, bir Docker konteynerinde örneklenebilmektedir. Gerekli yapılandırma aşağıda gösterilmiştir:

```
mongo-db:
    image: mongo:4.4
    hostname: mongo-db
    container_name: db-mongo
    expose:
      - "27017"
    ports:
      - "27017:27017"
    networks:
      - default
    volumes:
      - mongo-db:/data
```

1. `ports - "27017:27017"` , makinedeki 27017 numaralı bağlantı noktasını konteynerı ile eşleştirmektedir.

2. `volumes - mongo-db:/data` , veri tabanındaki verileri kalıcı hale getirmek için bir birim kullanmaktadır.

3.3.2. MySQL Veri tabanı

MySQL, SQL dilini arayüz olarak kullanan ilişkisel bir veri tabanı yönetim sistemidir. Bu çalışmada, veri geçmişinin kalıcılığını sağlamak için kullanılmaktadır. MySQL, bir Docker konteynerinde örneklenebilmektedir. Gerekli yapılandırma aşağıda belirtilmiştir:

```
mysql-db:
    restart: always
    image: mysql:5.7
    hostname: mysql-db
    container_name: db-mysql
    expose:
      - "3306"
    ports:
      - "3306:3306"
    networks:
      - default
```

MySQL-db konteyneri tek bir portu dinlemektedir:

`port 3306` , bir MySQL sunucusu için varsayılan porttur. İsterseniz verileri görüntülemek için diğer veri tabanı araçlarını çalıştırabilmeniz için açık hale getirilmiştir.

3.4. RASPBERRY PI

Raspberry Pi, çok yönlülüğü, düşük maliyeti ve çeşitli giriş/çıkış (I/O) arayüzleri sayesinde veri toplama için güçlü bir araçtır. İşte Raspberry Pi'nin veri toplama için bazı özellikleri:

1. Genel Amaçlı Giriş/Çıkış (GPIO): Raspberry Pi, programlanabilir GPIO pinleri aracılığıyla dijital girişler veya çıkışlar olarak işlev görebilir. Bu özellik, anahtarlar, hareket sensörleri, sıcaklık ve nem sensörleri, varlık dedektörleri vb. gibi dijital sensörlerin bağlanmasını mümkün kılmaktadır.

2. Seri iletişim: Raspberry Pi, sensörler, Küresel Konumlandırma Sistemi (GPS) modülleri, ekranlar vb. gibi çeşitli harici cihazlara bağlanmayı sağlayan seri iletişim protokolleri olan Evrensel Asenkron Alıcı-Verici (UART), Seri Çevresel Arayüz (SPI) ve Entegre Devreler Arası (I2C) desteklemektedir.

3. Ağ arayüzleri: Raspberry Pi, Wi-Fi ve Ethernet gibi yerleşik ağ arayüzlerine sahiptir. Bu özellik, yerel ağdaki diğer cihazlarla iletişim kurmayı ve hatta web API'leri gibi harici kaynaklardan veri almayı mümkün kılmaktadır.

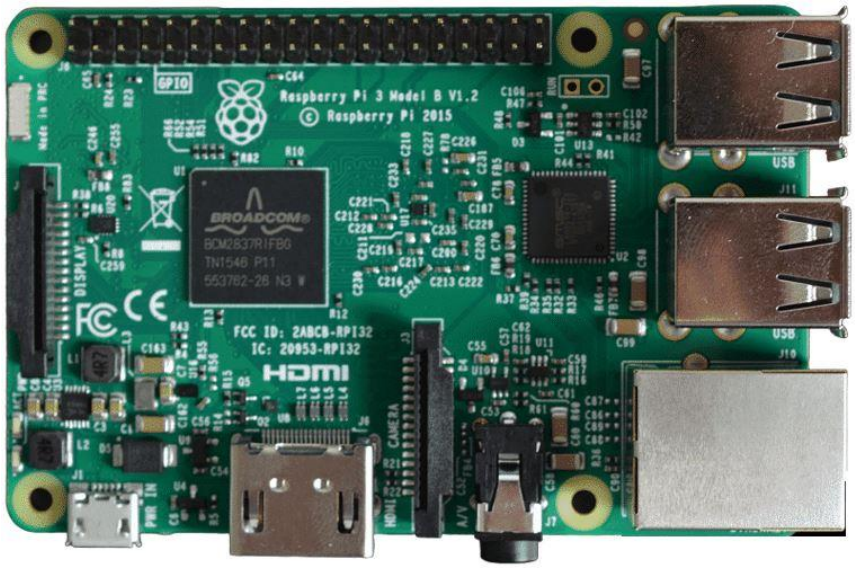
4. Kamera: Raspberry Pi, görüntü analizi, bilgisayar görüşü, gözetim vb. amaçlar için görüntü ve video yakalamayı mümkün kılan Kamera Seri Arayüzü (CSI) kameralarını desteklemektedir.

5. Yerel depolama: Raspberry Pi, verileri yerel olarak Güvenli Dijital (SD) kartında veya bağlı bir Evrensel Seri Veriyolu (USB) depolama cihazında saklayabilmektedir.

6. Yerel hesaplama: Raspberry Pi, Python, C/C++ gibi programlama dillerini kullanarak verileri yerel olarak işlemektedir. Bu özellik, veri tabanlı sistemlerin gerçek zamanlı analizi veya kontrolü için kullanışlıdır.

Tüm bu özellikleriyle Raspberry Pi, basit ev IoT projelerinden daha karmaşık endüstriyel izleme sistemlerine kadar geniş bir yelpazede veri toplama uygulamaları için popüler bir seçenektir.

Bu tez çalışmasında, kullanılan donanımın seçimi büyük önem taşımaktadır. Kullanım senaryosu için Wi-Fi kullanabilme özelliğine sahip bir Raspberry Pi kullanılması tavsiye edilmektedir. Bu çalışmada, DHT22 sensöründen ölçülen verilerin Orion Context Broker'a iletilmesi için Raspberry Pi 3 Model B kullanılmıştır. Veri iletimini başlatmak ve durdurmak için komutlar alır ve veri toplama aralığını ayarlar. Şekil 3.3'te, Raspberry Pi 3 Model B'nin bir görselini göstermektedir.



Şekil 3.3. Raspberry Pi 3 model B.

3.4.1. Raspberry Pi 3 Model B Özellikleri

Raspberry Pi 3 Model B, SoC'de 1.2 GHz hızında 4 ARM Cortex-A53 64-bit işlemci içeren bir Broadcom BCM2837'ye sahiptir. RAM miktarı 1 GB'dir. Videocore IV GPU

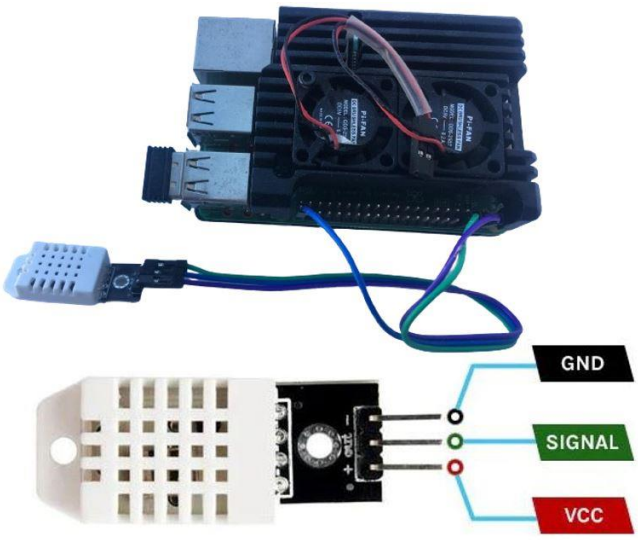
400 MHz clock hızındadır. Wi-Fi 802.11n, Bluetooth 4.1 ve BLE sağlar. Ayrıca devre dışı bırakılmış olsa da bir FM alıcısı bulunmaktadır.

Çizelge 3.1. Raspberry Pi 3 Model B özelliklerinin özeti.

Board	Raspberry Pi 3 Model B
İşlemci	Broadcom BCM2837
CPU	Quadcore ARM Cortex-A53, 64Bit
Clock	1.2 GHz
RAM	1 GB
GPU	400 MHz VideoCore IV®
Ağ bağlantısı	1 x 10 / 100 Ethernet (RJ45 Port)
Kablosuz bağlantı	802.11n Wireless LAN (Wi-Fi) and Bluetooth 4.1, BLE
USB portları	4 x USB 2.0
GPIO'lar	40 pins
Kamera arayüzü	15-pin MIPI
Ekran arayüzü	DSI 15 Pin / HDMI Out / Composite RCA
Güç kaynağı (Akım)	2.5 A

3.5. SICAKLIK VE NEM SENSÖRÜ (DHT22)

Önerilen prototipte, izleme süreci boyunca iki ana parametre tespit edilecektir: ortam sıcaklığı ve bağıl nem, ölçebilen sensörlerin kullanımını değerlendirdik. Bununla birlikte, kullanıcının özel uygulamasına bağlı olarak uygulanabilecek farklı sensör türlerini entegre etmek mümkündür. Bu çalışmada DHT22 ortam sıcaklığını ve hava nemini basit ve doğru bir şekilde tespit etmek için kullanılmaktadır. Şekil 3.4'te DHT22 modülü sensörünün Raspberry Pi pinlerine jumper kablolar aracılığıyla bağlantısını göstermektedir.



Şekil 3.4. DHT22 sensörünün Raspberry Pi'ye bağlanması.

Çoğu DHT22 sensör modülü, entegre bir dirençle birlikte gelir, böylece ek kablo bağlantısı veya lehimleme gereksinimini ortadan kaldırmaktadır.

DHT22 modülünü Raspberry Pi'ye aşağıdaki gibi bağlayın:

1. DHT22 VCC pininden Raspberry Pi 5V'a (5V GPIO pini)
2. DHT22 GND pininden Raspberry Pi GND'ye (GND GPIO pini)
3. DHT22 DATA pini Raspberry Pi üzerindeki bir GPIO pinine (GPIO pin 21)

DHT22 Sensörü, adından da anlaşılacağı gibi, -40 ila +80 ° Santigrat aralığındaki sıcaklığı ve 0 ila %100 aralığındaki hava nemini 2 ila %5 arasında değişen doğrulukla ölçmek için kullanılmaktadır. Arduino, Raspberry, ARM, AVR, PIC, vb. dahil olmak üzere mikrodenetleyici kartları kullanan elektronik ve robotik projelerin geliştirilmesinde yaygın olarak kullanılmaktadır. DHT22 sensörünün parametreleri için aşağıdaki tabloda gösterilmektedir.

Çizelge 3.2. DHT22 sensör parametreleri (referans değerler).

	DHT22
Sıcaklık aralığı	-40ºC to 80ºC
Sıcaklık hassasiyeti	± 0.5 °C

Sıcaklık çözünürlüğü	0.1ºC
Bağıl nem aralığı	0 to 100%
Nem hassasiyeti	± 2% RH
Nem çözünürlüğü	0.1% RH
Yanıt süresi	2s
Güç kaynağı	3.3 V to 5.5 V
Ölçüm Sırasında Maksimum Akım	2.5 mA

3.6. GRAFANA

Grafana, diğer bileşenlerden toplanan verileri tablolar ve grafikler şeklinde görüntülemek için kullanılmaktadır. İlk olarak, bu bileşen MySQL veri tabanıyla bir bağlantı oluşturur ve ardından panosunu güncellemek için sürekli olarak veri tabanını sorgulamaktadır. Pano, paydaşların ihtiyaçlarına en iyi şekilde cevap verecek şekilde yapılandırılabilmektedir. Bu çalışmada, Grafana, mevcut ortam sıcaklığı ve bağıl nem hakkında bilgileri görüntülemek için kullanılmaktadır.

Grafana bir Docker konteynerinde örneklenebilir. Gerekli yapılandırma aşağıda görülmektedir:

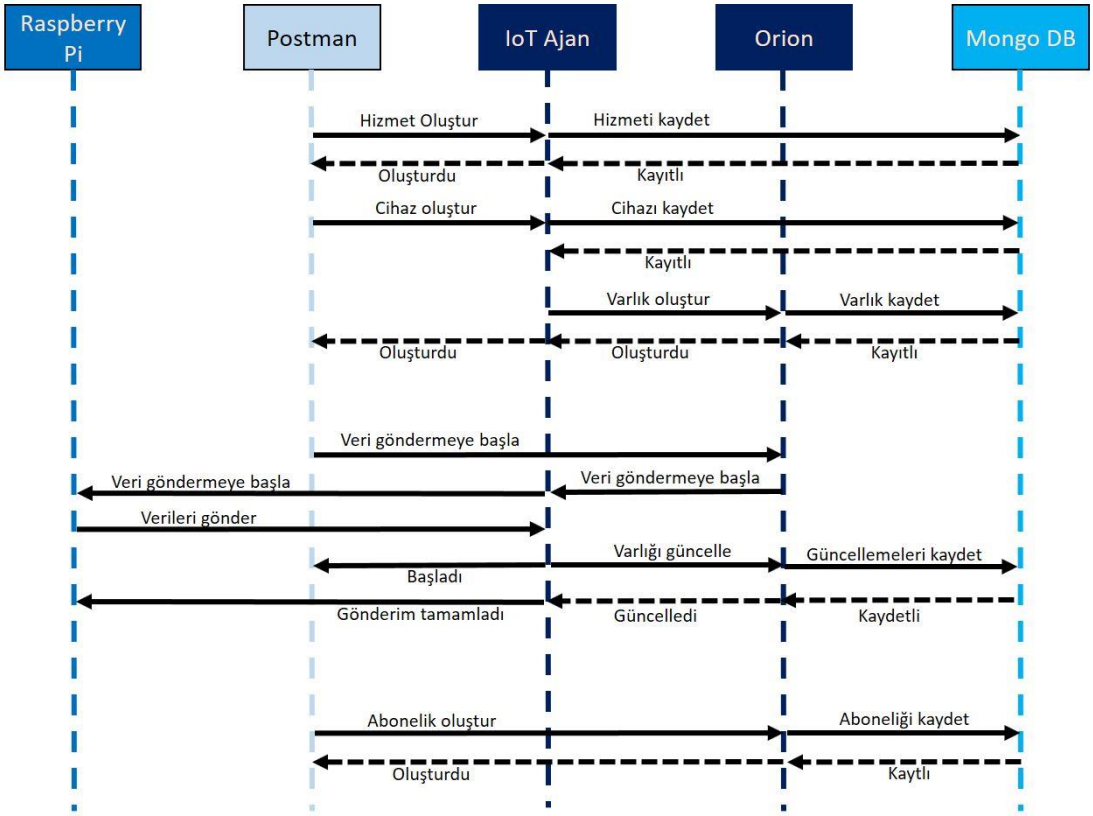
```
grafana:
   image: grafana/grafana
   hostname: grafana
   container_name: grafana
   depends_on:
       - mysql-db
   ports:
       - "3000:3000"
```

Bir tarayıcı açarak Grafana'ya erişmek için http://localhost:3000/ URL'sini ziyaret etmeniz gerekmektedir.

3.7. POSTMAN

Postman, kolayca etkileşime girebileceğimiz bir web arayüzü aracılığıyla basit API istekleri yaparak bir REST istemcisini test etmemizi sağlayan bir hizmettir. Farklı istek

türleri sunmaktadır (GET, POST, PUT, ...). Şekil 3.5'te, Postman ve diğer bileşenler arasındaki iletişimi göstermektedir.

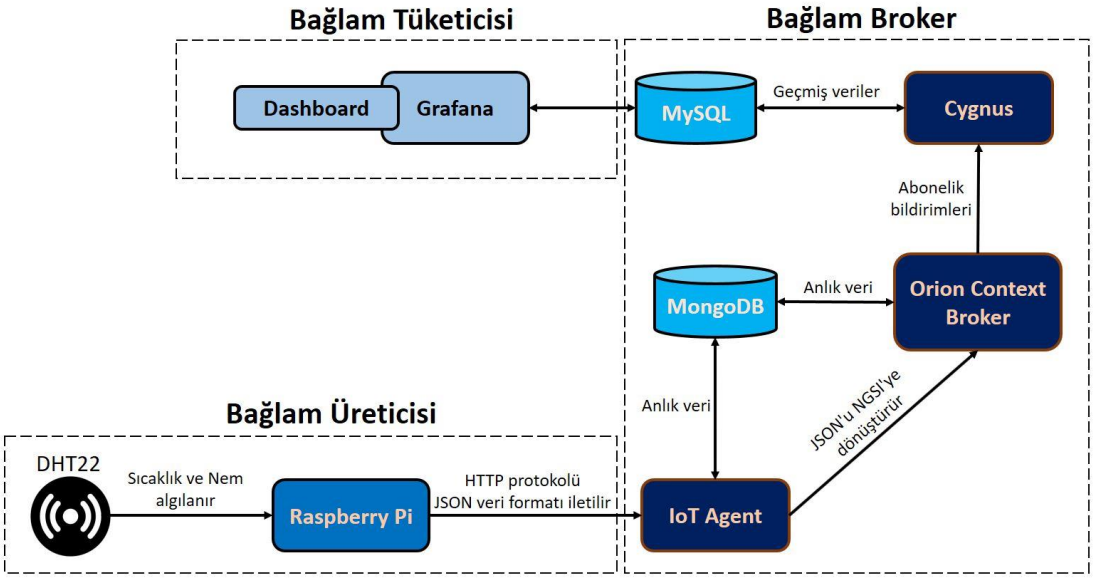


Şekil 3.5. Postman'ın platformdaki diğer bileşenlerle iletişimi.

Yukarıdaki diyagram, Postman bileşeninin platformun diğer bileşenlerle nasıl iletişim kurduğu gösterilmektedir. Postman, IoT Ajanı'nda ve Orion Context Broker'da bilgi modelinde tanımlanan varlıkları oluşturur. Bu varlıklar, IoT platformunda dijital nesnelerin temsilleri olarak kullanılır. Varlıkları oluşturmadan önce, bu bileşen verileri ayırmak için bir hizmet oluşturur. Bir varlık IoT Ajanı'nda oluşturulduğunda, Ajan, bu varlığın temsili olan bir varlığı Orion Context Broker'da oluşturur ve cihazlardan gelen verileri hangi varlığa göndereceğini belirlemek için bu varlığa veri gönderir. Postman, ardından IoT Ajanına veri göndermeye başlamasını ister ve bu da Orion Context Broker'daki varlığı günceller. Böylelikle, bu değişiklik MongoDB veritabanına kaydedilir. Postman, daha sonra Orion Context Broker'da Cygnus aboneliğini oluşturur, böylece Orion daha sonra varlıkta yapılan değişiklikler hakkında Cygnus'u bilgilendirebilir.

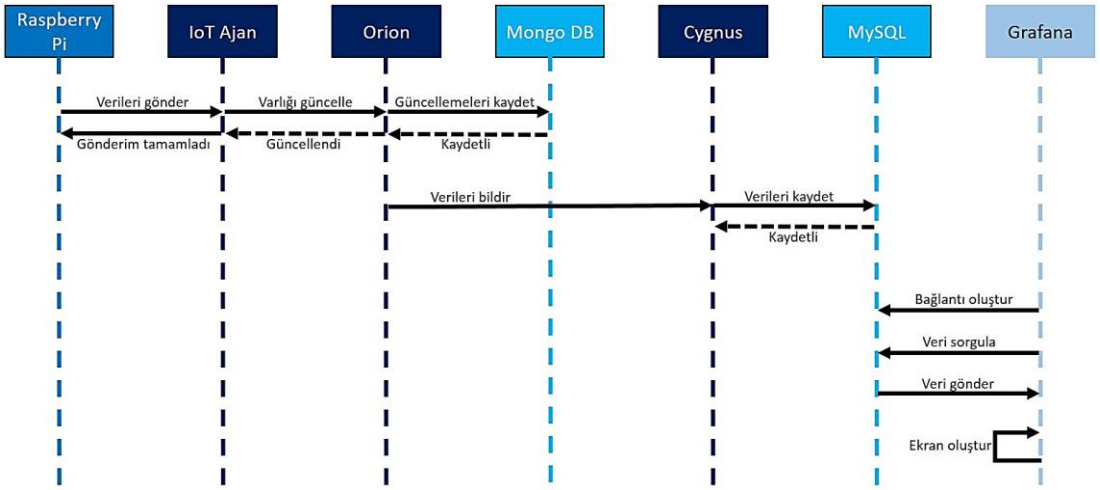
3.8. VERİ AKIŞI

Önerilen mimarideki veri akışı üç temel bölümden oluşmaktadır: Bunlar, bağlam üreticisi, bağlam broker ve bağlam tüketicisidir. Daha önce de belirtildiği gibi, Raspberry Pi bağlam üreticisi olarak görev yapmakta ve DHT22 sensörü tarafından ölçülen verileri platforma iletmektedir. FIWARE, bağlam broker ve geçmiş verileri depolamak için gereken tüm bileşenleri sağlar. Son olarak hem bağlam hem de geçmiş veriler bir görselleştirme bileşeni olan bir gösterge paneli tarafından tüketilmektedir.



Şekil 3.6. Önerilen mimarinin blok diyagramı.

Şekil 3.7'de, önerilen mimari içindeki bilgi akışını algıdan veri görselleştirmeye kadar bileşenler arasındaki iletişimi göstermektedir.



Şekil 3.7. Önerilen mimaride veri akışı.

Akış, Raspberry Pi'ye bağlı DHT22 sensörü tarafından ölçülen verileri yakalayarak başlar. Veriler, HTTP protokolü ve JSON veri formatı kullanılarak iletilmektedir. Bu veriler, cihazdan platforma yazılmasını yönetmekten sorumlu olan IoT Ajana gönderilmektedir.

IoT Ajanı, verileri NGSI formatına dönüştürerek ve Orion'a ileterek işlevini gerçekleştirilmektedir. Orion, tuttukları son durumu, bağlam bilgisi saklamak için MongoDB kullanılmaktadır. Orion'da Sıcaklık veya Nem öznitelikleri değiştirildiğinde, abonelik sistemi yeni değerleri Cygnus'a iletir ve bu, MySQL veri tabanında bu bilgilerin geçmişini kalıcı hale getirmektedir. Bu depolama, verilerin daha sonra Grafana, MySQL veritabanına bir bağlantı oluşturur ve ardından gösterge tablosunu güncellemek için sürekli olarak veritabanını sorgular. Pano, ilgili tarafların ihtiyaçlarını en iyi şekilde karşılayacak şekilde yapılandırılabilir. Bu çalışmada, ortam sıcaklığı ve bağıl nem bilgilerini görüntülemek için kullanılmaktadır.

BÖLÜM 4

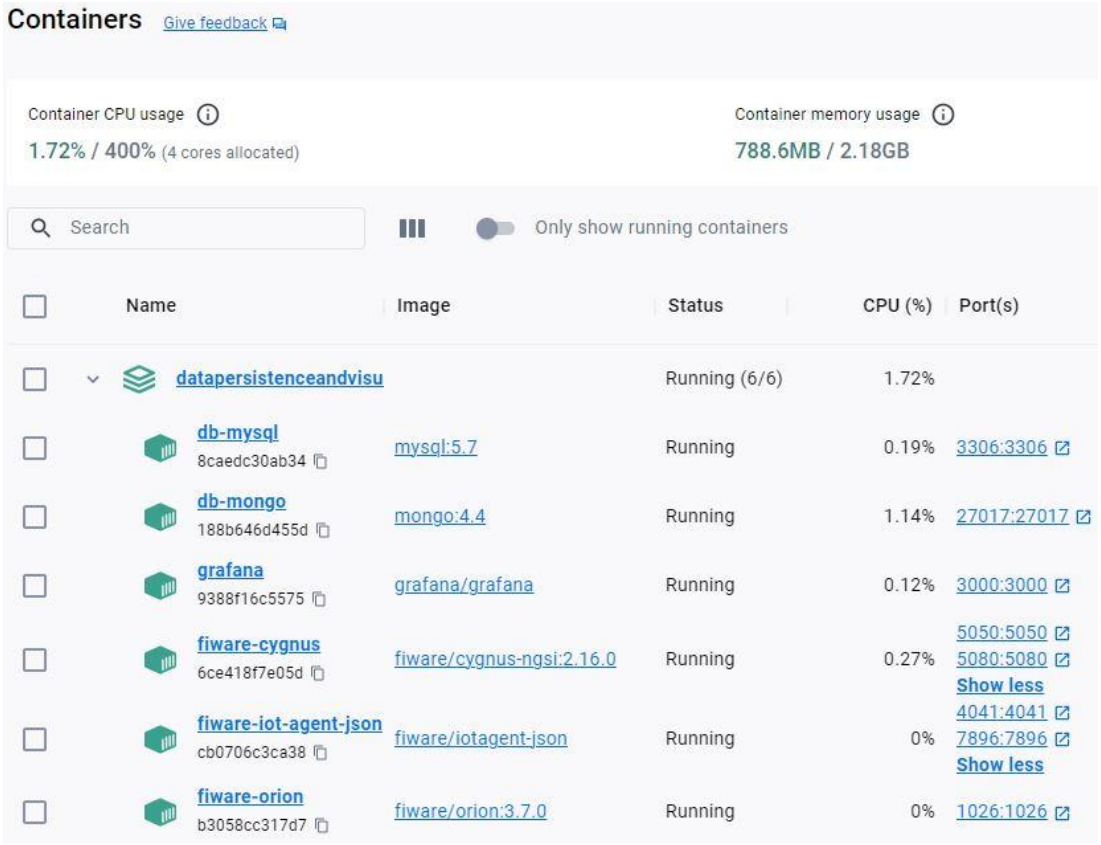
DENEYSEL ÇALIŞMALAR

Bu bölümde, platformun Raspberry Pi ile nasıl etkileşime geçtiğini anlamanıza yardımcı olmak için son derece basit bir kullanım durumu gösterilmektedir. Sistem konfigürasyonları ve geçmiş bağlam verilerin nasıl kalıcı hale getirileceği ve görselleştirileceği anlatılmaktadır. Bölüm 4.1'de, Docker konteynerlerinin başlatılması ve docker-compose içindeki konteynerlerin gösterilmesi anlatılmaktadır. Ardından bölüm 4.2'de, Raspberry Pi'ye uzaktan erişim sağlamak için nasıl bağlanılacağı anlatılmaktadır. Bölüm 4.3'te, Raspberry Pi'yi Orion Context Broker'a bağlamak için sistem yapılandırmalarını göstermektedir; bu bölüm, istek üzerine cihaz oluşturmak için bir hizmet oluşturarak ve ardından Orion Context Broker'da yeni bir cihaz oluşturarak sağlanmaktadır. Bölüm 4.4'te Grafana veri görselleştirme aracı ile FIWARE ara yazılım platformunda verilerin nasıl kalıcı hale getirileceği ve görselleştirileceği açıklanmaktadır.

4.1. DOCKER KONTEYNERLERİ BAŞLATMA

Bir terminal açın ve dosyanın bulunduğu klasöre gidin. Bundan sonra, tüm hizmetler aşağıda gösterildiği gibi "docker compose" komutu kullanılarak komut satırından başlatılmaktadır:

`docker compose up -d,` bu komut tüm konteynerleri oluşturur.



Şekil 4.1. Docker konteynerleri.

Temizlemek ve yeniden başlatmak için `docker compose down` komutu kullanılmakta ve ardından önceki komutu kullanılmaktadır.

`-d`, Docker'ın konteynerleri `detached` modda çalıştırması gerektiğini belirtilmektedir. Bu da konteynerlerin arka planda çalışacağı anlamına gelmektedir.

Docker uygulamasındaki her bir konteyneri kontrol edebilmektedir. Tüm konteynerleri listelemek ve durumlarını değerlendirmek için `docker compose ps -a` veya `docker ps -a` komutunu da kullanabilmektedir.

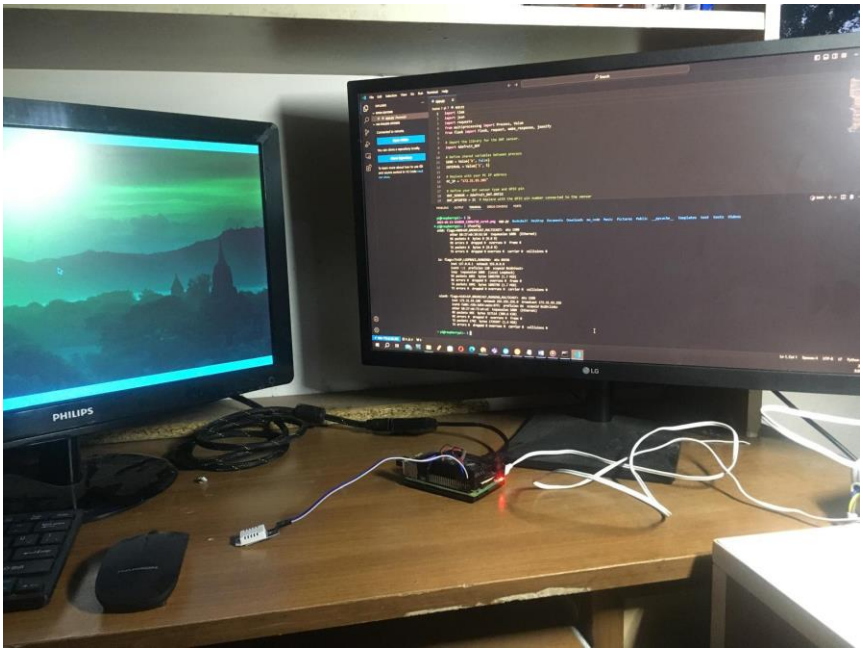
4.2. RASPBERRY Pİ KULLANIMI

Raspberry Pi'yi ve yerel bilgisayarı aynı ağa bağlamaktadır. Raspberry Pi'nin ve yerel bilgisayarın IP adresini bilmek de önemlidir. Bu çalışmada, yerel bilgisayar Windows

10 OS ve Raspberry Pi cihazları için optimize edilmiş Debian tabanlı olan Raspbian OS kullanılmıştır. Cihazlarda kullanılan işletim sistemine bağlı olarak `ifconfig` veya `ipconfig` komutları kullanılarak IP adresi bulabiliriz. Raspbian'ın özellikleri arasında çeşitli programlama dilleri için destek ve elektronik bileşenlerle etkileşim için GPIO erişimi bulunmaktadır.

Kullanıcılar Raspberry Pi'ye bu şekilde erişebilmektedir:

1. Harici bir monitör ve bir Yüksek Tanımlı Multimedya Arayüzü (HDMI) kablosu aracılığıyla.
2. Raspberry Pi'nin IP adresini bildiğinizde, Visual Studio Code'u kullanarak Güvenli Kabuk (SSH) aracılığıyla Raspberry Pi'ye bağlanabilmektedir.



Şekil 4.2. Raspberry Pi'ye bağlanma.

Raspberry Pi'ye girdikten sonra, bir terminal penceresi açmalı ve `raspberry` klasöründe bulunan python programlama dili kodlarına sahip `app.py` dosyasını Raspberry Pi'deki bir klasöre kopyalamaktadır. Bu dosyayı kullanmadan önce `pip install flask` komutunu kullanarak flask yüklemesi gerekmektedir. Bundan sonra aşağıdaki komutu kullanarak Raspberry Pi üzerinde sunucuyu başlatabilmektedir:

```
flask run --host=0.0.0.0
```

`--host=0.0.0.0` bayrağı, Flask web sunucusunun ağ üzerinde görünür olmasını ve dolayısıyla IoT Ajanı tarafından görülebilmesini sağlamaktadır.

4.3. SİSTEMİN YAPILANDIRILMASI

Öncelikle docker-compose konteynerlerini başlatmaktadır ve ardından Raspberry Pi'ye uzaktan erişmesi gerekmektedir. Raspberry Pi'yi Orion Context Broker'a bağlamak için IoT Ajanı JSON bir aracı hizmet olarak kullanılmaktadır. Bu tam olarak gerekli değildir, ancak IoT Ajanları, farklı cihaz protokollerini Orion Context Broker'da kullanılan NGSI-V2 protokolüne çevirebilen cihazlar ve Orion Context Broker arasında bir iletişim köprüsü olarak kullanıldığından tavsiye edilmektedir.

4.3.1. Hizmet Oluşturma

Postman'dan bir POST isteği gönderilir, bu istek IoT Ajan'da aşağıdaki istekle cihazlar oluşturabilmek için bir hizmet oluşturmaktadır:

```
{
    "services": [
        {
            "apikey": "4jggokgpepnvsb2uv4s40d59ov",
            "cbroker": "http://orion:1026",
            "entity_type": "Device",
            "resource": "/iot/json"
        }
    ]
}
```

Bu gövde, DHT22 türündeki varlıkların `apikey` değerine sahip bir api anahtarı kullanması gerektiğini belirtilmektedir. Bu API anahtarı, platforma veri gönderen cihazların kimliğini doğrulamak için kullanılmaktadır. `cbroker` değeri, Orion Context Broker'ın IP adresini temsil etmektedir.

4.3.2. Cihaz Oluşturma

Bir cihazı kaydetmek için IoT Ajan JSON'a bir POST isteği göndermesi gerekmektedir. Bir cihaz IoT Ajanında oluşturulduğunda, Ajan, bu cihazı temsili olan bir varlığı Orion Context Broker'da oluşturur ve cihazlardan gelen verileri hangi varlığa göndereceğini bilmek için bu varlığa veri gönderilmektedir.

Aşağıdaki istekle Orion Context Broker'da yeni bir varlık oluşturmaktadır:

```
{
    "devices": [
        {
            "device_id": "device_001",
            "entity_name": "urn:ngsi-ld:Device:001",
            "entity_type": "Device",
            "transport": "HTTP",
            "endpoint": "http://172.33.93.109:5000/Device/001",
            "timezone": "UTC+3",
            "attributes": [
                {
                    "object_id": "t",
                    "name": "temperature",
                    "type": "Number"
                },
                {
                    "object_id": "rh",
                    "name": "relativeHumidity",
                    "type": "Number"
                }
            ],
            "commands": [
                {
                    "name": "switch",
                    "type": "command"
                },
                {
                    "name": "interval",
                    "type": "command"
                }
            ]
        }
    ]
}
```

1. "device_id": "device_001" - IoT Ajan JSON 'unda bu cihaz kimliğini kaydeder.
2. "entity_name": "urn:ngsi-ld:Device:001" - Orion Context Broker 'da bu kimliğe sahip bir varlık kaydeder. Ayrıca, bu varlığı IoT Ajanı JSON ile eşleştirmek için Orion Context Broker içinde bir imza oluşturur.
3. "entity_type": "Device" - Varlığın türünü belirtir.
4. "transport": "HTTP" - HTTP aktarım protokolünün kullanılacağını belirtir.
5. "endpoint": "http://172.33.93.109:5000/device/001" - Cihazın 172.33.93.109 IP adresinde ve 5000 numaralı port bulunduğunu belirtir. IP adresi Raspberry Pi'nin IP adresidir ve port Flask web sunucusu için varsayılan porttur. Yalnızca bir cihazı belirttiğimiz için device/001 rotası kullanılır.
6. "timezone": "UTC+3" - Orion Context Broker'da kullanılan zaman damgasının bu zaman dilimine karşılık gelmesi gerektiğini belirtir.
7. "attributes" listesi - Bu cihazın gönderebileceği nitelikleri belirtir.
8. “commands" listesi - Bu cihazın alabildiği komutları gösterir.

4.4. GEÇMİŞ VERİLERİ KALICI HALE GETİRME VE GÖRSELLEŞTİRME

Bu çalışmada, Orion Context Broker'dan harici bir veri tabanına, bu durumda MySQL'e veri depolamak için Cygnus kullanılmaktadır. Çünkü ORION'un kendisi bağlam bilgilerini yalnızca gerçek zamanlı olarak sakladığından, geçmiş verileri saklamaz, bu nedenle Cygnus'un bu bilgileri alabilmesi ve geçmiş bilgilere sahip olabilmeniz için MySQL veri tabanında saklanmaktadır. Bu çalışmada Grafana bileşeni verileri görselleştirmek için kullanılmaktadır.

4.4.1. Veri tabanındaki Kalıcı Veriler

Geçmiş verileri kalıcı hale getirmek için Cygnus bileşeni kullanılmaktadır. Bu bileşen Orion Context Broker'dan bir bildirim almakta ve bunu docker-compose.yml dosyasında belirtilen veri tabanına kaydetmektedir. Bu durumda MySQL veri tabanı kullanılmaktadır. Bu mekanizmayı kullanmak için Orion Context Broker'da bir abonelik/kayıt oluşturmamız gerekmektedir.

4.4.1.1. Bağlam Değişikliklerinin İmzalanması

Aşağıdaki istekle Orion Context Broker'da bir Cygnus aboneliği oluşturulmaktadır:

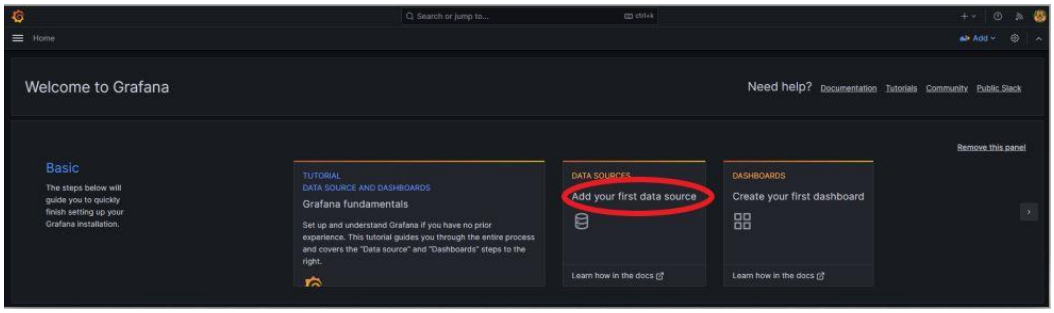
```
{
    "description": "SUbscription to Cygnus",
    "subject": {
        "entities": [
            {
                "id": "urn:ngsi-ld:Device:001",
                "type": "Device"
            }
        ],
        "condition": {
            "attrs": [
                "temperature"
            ]
        }
    },
    "notification": {
        "http": {
            "url": "http://cygnus:5050/notify"
        },
        "attrs": [
            "temperature",
            "relativeHumidity"
        ]
    },
    "throttling": 5
}
```

Postman'dan bir POST isteği gönderilir, bu istek, Orion Context Broker'da bir Cygnus aboneliği oluşturulmaktadır. Bu abonelik, sıcaklık değeri her 5 saniyede bir değiştiğinde Cygnus'u bilgilendirmektedir.

4.4.2. Verilerin Görselleştirilmesi

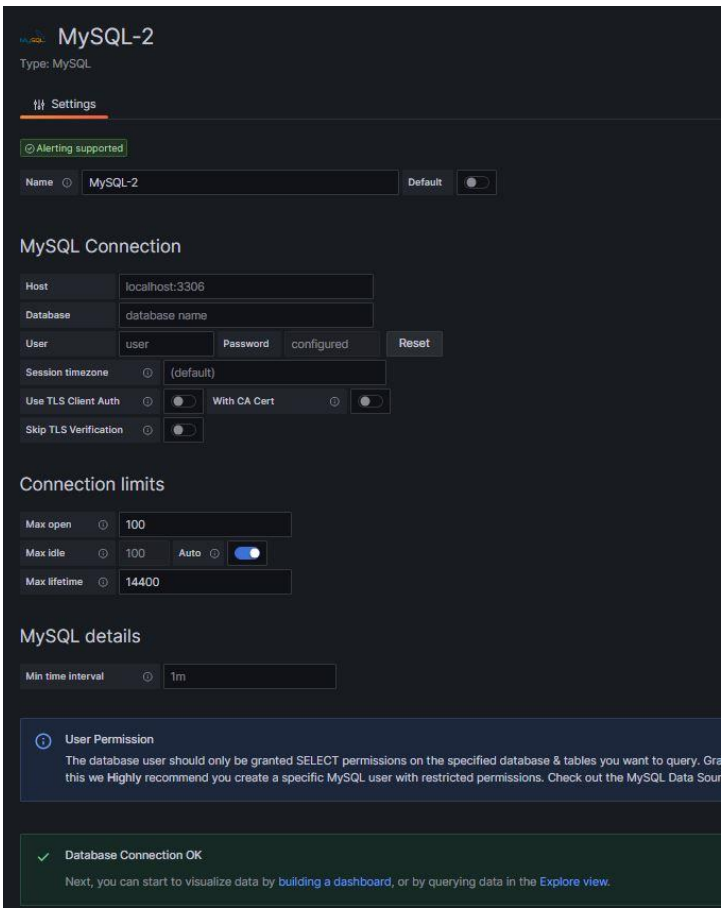
Grafana'ya erişmek için bir tarayıcı açmanız gerekmekte ve `http://localhost:3000/` URL'sine gidilmesi gerekmektedir. Şu şekilde erişilebilen bir oturum açma sayfası açmaktadır: `username=admin` ve `password=admin`. Sizden uygun gördüğünüz şekilde girebileceğiniz yeni bir parola istenmektedir. Yeni bir parola oluşturduktan sonra ana ekran görüntülenmektedir. Herhangi bir gösterge tablosunu yapılandırmadan önce bir veri kaynağı eklenmesi

gerekmektedir. Bunu yapmak için **Add your first data source** üzerine tıklanması gerekmektedir.



Şekil 4.3. Veri kaynağı eklemek.

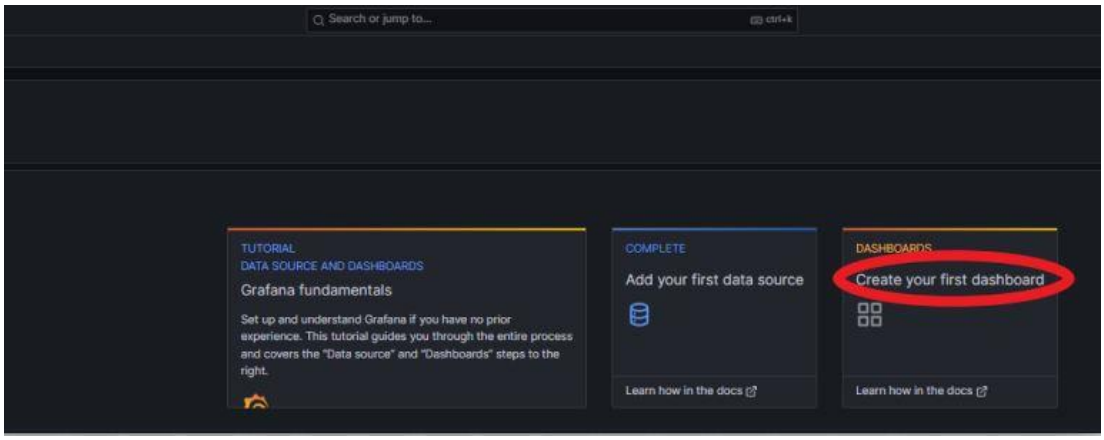
Arama çubuğunda veri kaynağı olarak MySQL'i seçin. Aşağıdaki bilgilerin doldurulması gerekmektedir:



Şekil 4.4. Grafana veri kaynağı kurulumu sekmesi.

save and test üzerine tıklanmakta ve her şey uygun şekilde çalıştığı gözlemlendiğinde, Veri Kaynağı güncellendi bilgisi ve yeşil bir bayrakla birlikte bir pop-up işareti görüntülenmektedir. Database Connection OK mesajı sayfanın altında görülmesi gerekmektedir.

Ana ekrana geri dönün ve Create your first dashboard üzerine tıklanır ve aşağıdaki gibi sayfa açılması gerekmektedir.



Şekil 4.5. Kontrol paneli oluştur.

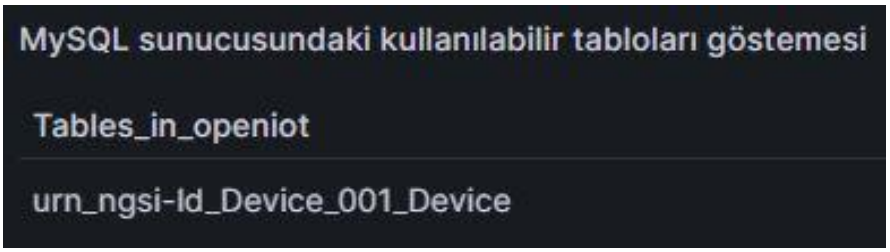
Yeni bir gösterge tablosu ekle seçeneğine tıkladığınızda, doldurmanız ve bir gösterge tablosu oluşturmanız için bir form içeren bir ekran yüklenmektedir. Sorgu seçeneğini Builder'dan Code değiştirilmektedir.

4.4.2.1. Grafana'da MySQL Veri tabanından Veri Okuma

Ağ üzerinde bir docker konteyneri çalıştığında, çalışan veri tabanı hakkında bilgi edinmek mümkündür. Kullanılabilir veritabanlarının listesini görüntülemek için SHOW DATABASES komutu çalıştırılır ve sonuç aşağıdaki gibi gözükmektedir:

Şekil 4.6. MySQL sunucusundaki kullanılabilir veri tabanları.

Cygnus'un Orion Context Broker aboneliğinin bir sonucu olarak, `openiot` adında yeni bir şema oluşturulmaktadır. Şemanın adı `fiware-service` başlığına karşılık gelmektedir. Bu nedenle `openiot` IoT cihazlarının geçmiş bağlamını içermektedir. Çalışan veri tabanından bilgi alarak `SHOW tables FROM openiot` komutu çalıştırılır ve sonuç aşağıdaki gibidir:



Şekil 4.7. MySQL sunucusundaki kullanılabilir tablosu.

4.4.2.2. Grafana'da MySQL Sunucusunun Geçmişi Hakkında Bilgi

Geçmiş değerlerini filtrelemek için aşağıda gösterildiği gibi bir select komutu çalıştırılmaktadır:

```
SELECT * from `urn_ngsi-ld_Device_001_Device` LIMIT 10;
```

Sonuç aşağıdaki gibidir:

MySQL sunucusunun geçmişi hakkında bilgi etmek

recvTimeTs	recvTime	fiwareServicePath	entityId	entityType	attrName	attrType	attrValue	attrMd
1704561056354	2024-01-06 17:10:56.354	/	urn:ngsi-ld:Device:001	Device	temperature	Number	19.700000763	[{"name":"Ti
1704561056354	2024-01-06 17:10:56.354	/	urn:ngsi-ld:Device:001	Device	relativeHumidity	Number	55.099998474	[{"name":"Ti
1704561062087	2024-01-06 17:11:02.87	/	urn:ngsi-ld:Device:001	Device	temperature	Number	19.700000763	[{"name":"Ti
1704561062087	2024-01-06 17:11:02.87	/	urn:ngsi-ld:Device:001	Device	relativeHumidity	Number	55.099998474	[{"name":"Ti
1705327975682	2024-01-15 14:12:55.682	/	urn:ngsi-ld:Device:001	Device	temperature	Number	17.299999237	[{"name":"Ti
1705327975682	2024-01-15 14:12:55.682	/	urn:ngsi-ld:Device:001	Device	relativeHumidity	Number	46.799999237	[{"name":"Ti
1705327982944	2024-01-15 14:13:02.944	/	urn:ngsi-ld:Device:001	Device	temperature	Number	17.299999237	[{"name":"Ti
1705327982944	2024-01-15 14:13:02.944	/	urn:ngsi-ld:Device:001	Device	relativeHumidity	Number	46.799999237	[{"name":"Ti

Şekil 4.8. MySQL sunucucunun geçmiş bağlam verileri.

MySQL'in sorgu syntax'ını kullanarak uygun alanları ve değerleri filtreleyebiliriz. Ortam sıcaklığının geçmiş değerlerini filtrelemek için aşağıda gösterildiği gibi bir SELECT komutu çalıştırılmaktadır.

```
SELECT recvtime, attrvalue from `urn_ngsi-ld_Device_001_Device` WHERE
attrName="temperature" LIMIT 10;
```

Sonuç aşağıdaki gibidir:

Sıcaklık

recvtime	attrvalue
2024-01-06 16:34:02.348	19.100000381
2024-01-06 16:34:14.646	19
2024-01-06 16:34:22.893	19.100000381
2024-01-06 16:34:30.146	19.100000381
2024-01-06 16:34:36.924	19.100000381
2024-01-06 16:34:42.496	19.100000381
2024-01-06 16:34:48.169	19.100000381
2024-01-06 16:34:56.628	19.100000381
2024-01-06 16:35:02.208	19.100000381
2024-01-06 16:35:07.879	19.100000381

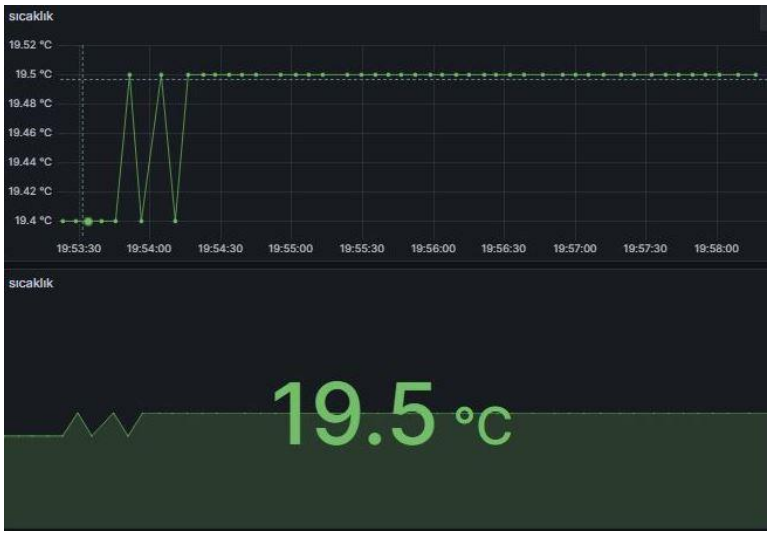
Şekil 4.9. Ölçülen ortam sıcaklığının geçmiş bağlam verileri.

Bu çalışmada Grafana ayarları kullanılarak iki grafik içeren bir panel oluşturulmaktadır. Bunlar biri sıcaklık ve diğeri bağıl nem içindir. Bu bilgileri içeren veri tabanını sorgulamak için, aşağıdaki sorgular kullanılmaktadır.

Sıcaklık için:

```
SELECT
  UNIX_TIMESTAMP(recvTime) as "time",
  CAST(attrValue as decimal(5,2)) as "value"
FROM `urn_ngsi-ld_Device_001_Device`
WHERE attrName = "temperature"
ORDER BY "time"
```

Sonuç aşağıdaki gibidir:



Şekil 4.10. Ortam sıcaklığı ölçüm bilgisi.

Bağıl nem için:

```
SELECT
  UNIX_TIMESTAMP(recvTime) as "time",
  CAST(attrValue as decimal(5,2)) as "value"
FROM `urn_ngsi-ld_Device_001_Device`
WHERE attrName = "relatveHumidity"
ORDER BY "time"
```

Sonuç aşağıdaki gibidir:

Şekil 4.11. Bağıl nem ölçüm bilgileri.

Şekil 4.12. Ortam sıcaklığı ve bağıl nem ölçümlerinden elde edilen bilgiler.

BÖLÜM 5

SONUÇ VE GELECEK ÇALIŞMALAR

Bu bölümde, IoT kavramının pratik uygulamasına yapılan ana katkıların bir özeti sunulmaktadır. Özellikle geçmiş bağlam verilerini kalıcı hale getirmek ve görselleştirmek için IoT ara yazılım platformunu kullanma konusu üzerinde durulmaktadır. Ayrıca, gelecekteki araştırmalar için potansiyel alanları vurgulamaktadır.

Bu çalışmanın amacı, FIWARE platformuna dayalı bir sistem mimarisi geliştirmek, veri kalıcılığı ve görselleştirme için en uygun bileşenleri belirlemek ve seçmek, ayrıca sensörden gelen verileri toplamak ve platforma göndermek için uygun düşük maliyetli cihazı seçmektir. FIWARE, belirli bir uygulamaya veya işleve bağlı olmaması nedeniyle çevik bir şekilde benimsenebilen ve yeniden kullanılabilen birçok karmaşık bileşen veya genel etkinleştirici mimarisi içermektedir. Bileşenleri uygulamanın ihtiyaçlarına göre uyarlayarak birçok uygulama yelpazesine uyarlanabilmektedir.

5.1. ARAŞTIRMA KATKILARI

Bu araştırma, IoT kavramının pratik kullanımına katkıda bulunmaktadır. Özellikle de IoT ara yazılım platformunun geçmiş bağlam verilerini kalıcı hale getirme ve görselleştirme amacıyla kullanımı konusunda katkı sunmaktadır.

1. Araştırmacılara bir kullanım senaryosu sağlar ve FIWARE platformu ile Raspberry Pi'nin nasıl bağlanacağını, sensörler tarafından ölçülen verilerin platforma nasıl gönderileceğini ve ölçümlerin daha sonraki analiz için bir veri tabanında nasıl kalıcı hale getirebileceğini göstermektedir. Raspberry Pi, IoT projelerinde pratik, uygun maliyetli çözümler için kullanılabilecek düşük maliyetli, düşük tüketim bir cihazdır. Raspberry Pi, bir IoT ara yazılım

platformuna veri göndererek bilgilerin gerçek zamanlı olarak toplanmasına ve işlenmesine önemli bir katkı sağlayabilmektedir. IoT ara yazılım platformları, IoT kavramı basitleştirilmesinde ve verimli bir şekilde uygulanmasında çok önemli bir rol oynamaktadır. IoT çözümlerini geliştirmek, dağıtmak ve yönetmek için gereken çabayı önemli ölçüde azaltan çeşitli araçlar ve hizmetler sunmaktadır.

2. Bir diğer önemli katkı ise bu çalışmada kullanılan teknolojilerin açık kaynak kodlu ve düşük maliyetli olması ve hem akademik çalışmalar hem de büyük şirket projeleri için kullanılabilmesidir. Açık kaynak teknolojileri IoT'nin geliştirilmesi ve ilerletilmesinde önemli bir rol oynamaktadır. IoT'ye katkıları, esneklik, şeffaflık, iş birliği ve yenilikçilik gibi faydalar sağlayan bir dizi alanda görülebilmektedir. Düşük maliyetli teknolojiler, finansal engelleri azaltarak ve daha erişilebilir çözümler sunarak IoT'nin yaygınlaşmasında, popülerleşmesinde ve demokratikleşmesinde kilit bir rol oynamaktadır. IoT çözümlerinin uygulanmasını çeşitli sektörler ve senaryolar için daha erişilebilir hale getirerek sınırlı kaynaklarla yenilikçi projelerin oluşturulmasını sağlamaktadır.

3. Deneyler, önerilen sistemin kullanılmasının, geçmiş bağlam verilerinin açık kaynaklı bir ara yazılım platformunda tutulmasını ve bilgilerin açık kaynaklı bir görselleştirme aracında kolayca ve şeffaf bir şekilde görselleştirilmesini mümkün kılarak uygulama çabasını azalttığını göstermiştir. Geliştiriciler ve yöneticiler, IoT projelerinin uygulanmasında Grafana gibi açık kaynaklı bir veri görselleştirme aracı kullanarak, sağlam ve verimli bir izleme arayüzü oluşturmak için gereken çabayı azaltabilir, zamandan ve kaynaklardan tasarruf edebilirler. Verileri net ve sezgisel bir şekilde görselleştirme yeteneği, IoT uygulamalarının başarısına önemli ölçüde katkıda bulunmaktadır.

4. Bu tez çalışmasında, akıllı şehir uygulamaları geliştirmek için bazı IoT ara yazılım platformlarını sunmaktadır. Ayrıca, sunulan platformlar gereksinimler (Birlikte çalışabilirlik, Cihaz keşfi ve yönetimi, Dinamik adaptasyon, Bağlam bilimi, Ölçeklenebilirlik, Büyük hacimli verilerin işlenmesi, Güvenlik, Veri

yönetimi ve Geliştirme araçları) analiz edilmiştir. FIWARE tüm gereksinimleri karşılayan tek platformdu ve bu çalışmada verileri kalıcı hale getirmek ve görselleştirmek için kullanılmaktadır.

5.2. ARAŞTIRMA SINIRLAMALARI

1. Raspberry Pi bir FIWARE ara yazılım platformuna nasıl bağlanacağına ve Grafana'da bilgilerin nasıl görüntüleneceğine odaklanılmaktadır. Ayrıca, Orion Context Broker'dan gelen bağlam bilgilerini harici veri tabanlarında tutmanın bir yolunu sunmaktadır. Bu yöntem FIWARE Cygnus bileşenini ve Orion Context Broker tarafından sağlanan abonelik mekanizmasını kullanmaktadır. Bu bileşeni ve mekanizmayı kullanarak, geçmiş bağlam bilgilerini oluşturabilir ve SQL veri tabanını gerektiği gibi okunması mümkündür.

2. FIWARE tabanlı bir sistem mimarisi geliştirilerek, Raspberry Pi kullanılarak bir sensörden ölçülen verilerin ara yazılım platformuna gönderilmesi, bağlam güncellendiğinde bir veri tabanındaki durum değişikliklerinin geçmiş veriler kalıcı hale getirilmesi ve görselleştirilmesi için en uygun bileşenlerin belirlenip seçilmesi amaçlanmaktadır. Ayrıca, bu sistem için uygun ve düşük maliyetli cihazı ve HTTP iletişim protokolü kullanılarak uygulanmaktadır.

5.3. GELECEK ÇALIŞMALAR

IoT kavramlarının ve mevcut teknolojilerin kullanımı, gelecekteki çalışmalar için aşağıdakiler gibi çeşitli fırsatlar sunmaktadır:

1. Diğer teknolojilerin kullanımının sistemin performansını ve ölçeklenebilirliğini etkileyip etkilemeyeceğini görmek için diğer ara yazılımlarla testler gerçekleştirmek.

2. MQTT ve CoAP gibi diğer bulut iletişim protokolleriyle testler gerçekleştirmek ve ayrıca UltraLight (UL) gibi başka bir veri formatının kullanımını test etmektir.

3. Endüstri 4.0 bağlamında IoT kavramını ve semantik modelleri uygulayarak endüstriyel sistemleri entegre etmek. IoT kavramının uygulamalarından biri de endüstri alanındadır ve Endüstri 4.0 olarak adlandırılan yeni bir endüstriyel devrime öncülük etmektedir. Bu alandaki gelecek çalışmalar, Endüstri 4.0 bağlamında, sistemin unsurlarını temsil etmek için ontolojileri ve entegrasyon aracı olarak hizmet etmek için IoT ara yazılımını kullanarak bileşenleri otomatik bir şekilde entegre edebilmektedir [130].

4. IoT kavramı, teknolojilerin kullanımında giderek gelişen hassas tarım alanında uygulanmak. IoT, hassas sulamayı otomatikleştirmek için doğal bir seçimdir, ancak uygunluğu henüz en iyi şekilde kanıtlanmamıştır. Geleneksel sulama yöntemleri, hassas sulama için gereken temel bilgileri tam olarak bilmediğinden su rezervlerini doğru şekilde kullanmamaktadır. IoT ara yazılım platformlarının kullanımı, yönetim yapısında temel bir rol oynayabilmektedir [131].

KAYNAKLAR

1. Atzori, L., A. Iera, and G. Morabito, *The internet of things: A survey*. Computer networks, 2010. **54**(15): p. 2787-2805.

2. Nayyar, A., A. Kumar, and A. Solanki, *DIGITAL CITIES ROADMAP: IoT-based architecture and Sustainable Buildings*. 2021.

3. Nand, S., *Environmental Energy Harvesting Techniques to Power Standalone IoT-Equipped Sensor and Its Application in 5G Communication.* Emerging Science Journal, 2021. **4**: p. 116-126.

4. Wan, J., et al., *Wearable IoT enabled real-time health monitoring system.* EURASIP Journal on Wireless Communications and Networking, 2018. **2018**(1): p. 1-10.

5. Chaudhary, S., et al. *CRAIoT: concept, review and application (s) of IoT*. in *2019 4th international conference on internet of things: Smart innovation and usages (IoT-SIU)*. 2019. IEEE.

6. Gou, H. and Y. Yoo. *A geographic location based reliable data transmission protocol for wireless sensor networks*. in *2012 IEEE 12th International Conference on Computer and Information Technology*. 2012. IEEE.

7. Qin, X., et al., *Making data visualization more efficient and effective: a survey.* The VLDB Journal, 2020. **29**(1): p. 93-117.

8. Leppänen, T., *Data visualization and monitoring with Grafana and Prometheus.* 2021.

9. Bidollahkhani, M., et al., *LoRaline: A critical message passing line of communication for anomaly mapping in IoV systems.* IEEE Access, 2023. **11**: p. 18107-18120.

10. Ahmad, K., et al., *Developing future human-centered smart cities: Critical analysis of smart city security, Data management, and Ethical challenges.* Computer Science Review, 2022. **43**: p. 100452.

11. Sookhak, M., et al., *Security and privacy of smart cities: a survey, research issues and challenges.* IEEE Communications Surveys & Tutorials, 2018. **21**(2): p. 1718-1743.

12. Teixeira, T., et al. *Service oriented middleware for the internet of things: A perspective*. in *European conference on a service-based internet*. 2011. Springer.

13. Bansal, S. and D. Kumar, *IoT ecosystem: A survey on devices, gateways, operating systems, middleware and communication.* International Journal of Wireless Information Networks, 2020. **27**(3): p. 340-364.

14. Bernstein, P.A., *Middleware: a model for distributed system services.* Communications of the ACM, 1996. **39**(2): p. 86-98.

15. Al-Fuqaha, A., et al., *Internet of things: A survey on enabling technologies, protocols, and applications.* IEEE communications surveys & tutorials, 2015. **17**(4): p. 2347-2376.

16. Bell, C., *Beginning sensor networks with Arduino and Raspberry Pi*. 2014: Apress.

17. Bauer, M. *Fiware: Standard-based open source components for cross-domain iot platforms*. in *2022 IEEE 8th World Forum on Internet of Things (WF-IoT)*. 2022. IEEE.

18. Ahmed, E., et al., *The role of big data analytics in Internet of Things.* Computer Networks, 2017. **129**: p. 459-471.

19. Banafa, A., *Three major challenges facing iot*. 2017.

20. Zhou, Y., et al. *Enabling query of frequently updated data from mobile sensing sources*. in *2014 IEEE 17th International Conference on Computational Science and Engineering*. 2014. IEEE.

21. Dakkak, O., S. Arif, and S.A. Nor, *A critical analysis of simulators in grid.* Jurnal Teknologi, 2015. **77**(4).

22. Dakkak, O., S. Arif, and S.A. Nor, *Resource allocation mechanisms in computational grid: A survey.* Asian Research Publishing Network (ARPN), 2015. **10**.

23. SalemAlzboon, M., et al. *Peer to peer resource discovery mechanisms in grid computing: a critical review*. in *The 4th International Conference on Internet Applications, Protocols and Services (NETAPPS2015)*. 2015.

24. Dakkak, O., S.A. Nor, and S. Arif. *Scheduling through backfilling technique for HPC applications in grid computing environment*. in *2016 IEEE Conference on Open Systems (ICOS)*. 2016. IEEE.

25. Dakkak, O., S.A. Nor, and S. Arif, *Proposed algorithm for scheduling in computational grid using backfilling and optimization techniques.* Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 2016. **8**(10): p. 133-138.

26. Dakkak, O., S.A. Nor, and S. Arif, *Scheduling Jobs through Gap Filling and Optimization Techniques in Computational Grid.* J. Comput. Sci., 2017. **13**(5): p. 105-113.

27. Rahman, H. and M.I. Hussain, *A comprehensive survey on semantic interoperability for Internet of Things: State-of-the-art and research challenges.* Transactions on Emerging Telecommunications Technologies, 2020. **31**(12): p. e3902.

28. Chen, M., S. Mao, and Y. Liu, *Big data: A survey.* Mobile networks and applications, 2014. **19**: p. 171-209.

29. Tracey, D. and C. Sreenan. *A holistic architecture for the internet of things, sensing services and big data*. in *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*. 2013. IEEE.

30. Hindia, M.N., et al., *Cloud computing applications and platforms: A Survey.* 2014.

31. lahmood HAMEED, F. and O. DAKKAK. *Brain Tumor Detection and Classification Using Convolutional Neural Network (CNN)*. in *2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*. 2022. IEEE.

32. López-Riquelme, J., et al., *A software architecture based on FIWARE cloud for Precision Agriculture.* Agricultural water management, 2017. **183**: p. 123-135.

33. SALLEMINE, A., *An Open-Source Approach to Wireless Internet of Things (IoT): Smart Agriculture Proof-of Concept Using FIWARE*. 2019, PAUWES.

34. Avcı, İ., *Akıllı evlerde IoT teknolojileri ve siber güvenlik.* Avrupa Bilim ve Teknoloji Dergisi, 2022(34): p. 226-233.

35. Cerbulescu, C.C., M. Marian, and E. Ganea. *IoT Big Data Management For Improved Response Time*. in *2021 22nd International Carpathian Control Conference (ICCC)*. 2021. IEEE.

36. Fakhri, M.N., et al. *Modeling Middleware Platform for Supporting Active Communication Between IoT Devices*. in *2019 International Conference on Information Science and Communication Technology (ICISCT)*. 2019. IEEE.

37. Munoz-Arcentales, A., et al., *Data usage and access control in industrial data spaces: Implementation using FIWARE.* Sustainability, 2020. **12**(9): p. 3885.

38. Mehmood, Y., et al., *Internet-of-things-based smart cities: Recent advances and challenges.* IEEE Communications Magazine, 2017. **55**(9): p. 16-24.

39. Razzaque, M.A., et al., *Middleware for internet of things: a survey*. IEEE Internet of things journal, 2015. **3**(1): p. 70-95.

40. Wang, J., et al., *Big data analytics for intelligent manufacturing systems: A review*. Journal of Manufacturing Systems, 2022. **62**: p. 738-752.

41. Sharma, S. and R. Sharma. *An Innovative Solution for Data Persistence Problem in Reliable Data Transmission*. in *2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*. 2023. IEEE.

42. Samaila, M.G., et al., *Challenges of securing Internet of Things devices: A survey*. Security and Privacy, 2018. **1**(2): p. e20.

43. Yang, H.-C., S. Choi, and M.-S. Alouini, *Ultra-reliable low-latency transmission of small data over fading channels: A data-oriented analysis.* IEEE Communications Letters, 2019. **24**(3): p. 515-519.

44. Oussous, A., et al., *Big Data technologies: A survey*. Journal of King Saud University-Computer and Information Sciences, 2018. **30**(4): p. 431-448.

45. Lavalle, A., et al., *Improving sustainability of smart cities through visualization techniques for big data from IoT devices.* Sustainability, 2020. **12**(14): p. 5595.

46. Nourildean, S.W., M.D. Hassib, and Y. Mohammed, *Internet of things based wireless sensor network: a review.* Indones. J. Electr. Eng. Comput. Sci, 2022. **27**(1): p. 246-261.

47. Sudarmani, R., et al., *Machine to machine communication enabled internet of things: a review*. International Journal of Reconfigurable and Embedded Systems, 2022. **11**(2): p. 126.

48. Chander, B. and G. Kumaravelan, *Internet of things: foundation.* Principles of Internet of Things (IoT) Ecosystem: Insight Paradigm, 2020: p. 3-33.

49. de Amorim Silva, R., et al. *Aplicando internet das coisas na educação: Tecnologia, cenários e projeções*. in *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*. 2017.

50. Klein, A., F.B. Pacheco, and R.d.R. Righi, *Internet of things-based products/services: process and challenges on developing the business models.* JISTEM-Journal of Information Systems and Technology Management, 2017. **14**: p. 439-461.

51. Pliatsios, A., K. Kotis, and C. Goumopoulos, *A systematic review on semantic interoperability in the IoE-enabled smart cities.* Internet of Things, 2023. **22**: p. 100754.

52. Trappey, A.J., et al., *A review of essential standards and patent landscapes for the Internet of Things: A key enabler for Industry 4.0.* Advanced Engineering Informatics, 2017. **33**: p. 208-229.

53. Nath, K., *Evolution of the internet from web 1.0 to metaverse: The good, the bad and the ugly.* TechRxiv Powered by IEEE, scholar. archive. org, 2022: p. 1-12.

54. Han, C., et al., *A cross-layer communication module for the Internet of Things.* Computer Networks, 2013. **57**(3): p. 622-633.

55. Sunyaev, A. and A. Sunyaev, *Internet computing*. 2020: Springer.

56. Platform, K.A., *Internet.* Zurich, Switzerland: KNIME AG, 2020.

57. Kanmai, Z. *TCP/IP protocol security problems and defenses*. in *2020 International Conference on Intelligent Computing and Human-Computer Interaction (ICHCI)*. 2020. IEEE.

58. Tian, Y.-C. and J. Gao, *Network Addressing Architecture*, in *Network Analysis and Architecture*. 2023, Springer. p. 161-219.

59. Ashraf, Z., et al., *Challenges and Mitigation Strategies for Transition from IPv4 Network to Virtualized Next-Generation IPv6 Network.* Int. Arab J. Inf. Technol., 2023. **20**(1): p. 78-91.

60. Adaramola, M.F., et al., *Taxonomy of Internet Protocol Addressing Standards for Next Generation Internet Services.* 2024.

61. Bayılmış, C., et al., *A survey on communication protocols and performance evaluations for Internet of Things.* Digital Communications and Networks, 2022. **8**(6): p. 1094-1104.

62. Behal, A., J.K. Sandhu, and G. Gupta. *Comparing HTTP And COAP For IoT Low-power and Lossy Networks Using The Cooja Simulator*. in *2023 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*. 2023. IEEE.

63. Gerez, A.H., et al. *Energy and processing demand analysis of tls protocol in internet of things applications*. in *2018 IEEE International Workshop on Signal Processing Systems (SiPS)*. 2018. IEEE.

64. Hou, C.-D., et al., *SeaHTTP: A resource-oriented protocol to extend rest style for Web of Things.* Journal of Computer Science and Technology, 2014. **29**(2): p. 205-215.

65. Naik, G.P. and A.U. Bapat, *A brief comparative analysis on application layer protocols of internet of things: MQTT, CoAP, AMQP and HTTP*. Int. J. Comput. Sci. Mob. Comput, 2020. **9**(9): p. 135-141.

66. Yassein, M.B., et al. *Internet of Things: Survey and open issues of MQTT protocol*. in *2017 international conference on engineering & MIS (ICEMIS)*. 2017. IEEE.

67. Azzedin, F. and T. Alhazmi, *Secure data distribution architecture in IoT using MQTT.* Applied Sciences, 2023. **13**(4): p. 2515.

68. Dakkak, O., et al. *Improving QoS for non-trivial applications in grid computing*. in *Emerging Trends in Intelligent Computing and Informatics: Data Science, Intelligent Information Systems and Smart Computing 4*. 2020. Springer.

69. Toldinas, J., et al. *MQTT quality of service versus energy consumption*. in *2019 23rd International Conference Electronics*. 2019. IEEE.

70. Silva, D., et al., *A performance analysis of internet of things networking protocols: Evaluating MQTT, CoAP, OPC UA.* Applied Sciences, 2021. **11**(11): p. 4879.

71. Raza, S., et al., *SecureSense: End-to-end secure communication architecture for the cloud-connected Internet of Things.* Future Generation Computer Systems, 2017. **77**: p. 40-51.

72. Al-Masri, E., et al., *Investigating messaging protocols for the Internet of Things (IoT).* IEEE Access, 2020. **8**: p. 94880-94911.

73. Shelby, Z. and C. Bormann, *6LoWPAN: The Wireless Embedded Internet John Wiley & Sons*. 2009, Inc.

74. ALASALI, T. and O. DAKKAK, *Exploring the landscape of sdn-based ddos defense: A holistic examination of detection and mitigation approaches, research gaps and promising avenues for future exploration.* International Journal of Advanced Natural Sciences and Engineering Researches, 2023. **7**(4): p. 327-349.

75. Abosata, N., et al., *Internet of things for system integrity: A comprehensive survey on security, attacks and countermeasures for industrial applications.* Sensors, 2021. **21**(11): p. 3654.

76. Raeisi-Varzaneh, M., et al., *Internet of Things: Security, Issues, Threats, and Assessment of Different Cryptographic Technologies.* Journal of Communications, 2024. **19**(2).

77. VARZANEH, M.R., A. HABBAL, and O. DAKKAK, *FIREWALLS AND INTERNET OF THINGS SECURITY: A SURVEY*. Current Trends in Computing, 2024. **1**(1): p. 22-43.

78. Dakkak, O., et al., *Towards accommodating deadline driven jobs on high performance computing platforms in grid computing environment.* Journal of Computational Science, 2021. **54**: p. 101439.

79. Farahzadi, A., et al., *Middleware technologies for cloud of things: a survey*. Digital Communications and Networks, 2018. **4**(3): p. 176-188.

80. Pires, P.F., et al., *Plataformas para a internet das coisas.* Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2015.

81. Lea, R. and M. Blackstock. *City hub: A cloud-based iot platform for smart cities*. in *2014 IEEE 6th international conference on cloud computing technology and science*. 2014. IEEE.

82. Wan, J., et al. *M2M communications for smart city: An event-based architecture*. in *2012 IEEE 12th International Conference on Computer and Information Technology*. 2012. IEEE.

83. Araujo, V., et al., *Performance evaluation of FIWARE: A cloud-based IoT platform for smart cities.* Journal of Parallel and Distributed Computing, 2019. **132**: p. 250-261.

84. Salinas, K., et al. *Cityhub: A library for urban data integration*. in *2022 35th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. 2022. IEEE.

85. Wondratschek, F., V. Gangadharan, and N. Wilson, *How might a City Hub at Stuttgart Central Station reduce Emissions and improve the Pre-last Mile Delivery by Rail Freight?* ScienceOpen Preprints, 2024.

86. Winn, J., *Open data and the academy: An evaluation of CKAN for research data management.* 2013.

87. Blackstock, M. and R. Lea. *IoT mashups with the WoTKit*. in *2012 3rd IEEE International Conference on the Internet of Things*. 2012. IEEE.

88. Tachmazidis, I., et al. *Hypercat RDF: semantic enrichment for IoT*. in *Semantic Technology: 6th Joint International Conference, JIST 2016, Singapore, Singapore, November 2-4, 2016, Revised Selected Papers 6*. 2016. Springer.

89. Filipponi, L., et al. *Smart city: An event driven architecture for monitoring public spaces with heterogeneous sensors*. in *2010 Fourth International Conference on Sensor Technologies and Applications*. 2010. IEEE.

90. Dakkak, O., et al. *From grids to clouds: Recap on challenges and solutions*. in *AIP Conference Proceedings*. 2018. AIP Publishing.

91. Vicente Cebrián, V.M., *Interconexión de dispositivos IoT (Internet of Things) con plataforma Sofia2*. 2017, Universitat Politècnica de València.

92. Sajat, M.S., et al., *A critical review on energy-efficient medium access control for wireless and mobile sensor networks.* Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 2016. **8**(10): p. 89-94.

93. D'Elia, A., et al., *Enabling interoperability in the internet of things: A osgi semantic information broker implementation.* International Journal on Semantic Web and Information Systems (IJSWIS), 2017. **13**(1): p. 148-168.

94. Pantsar-Syväniemi, S., J. Kuusijärvi, and E. Ovaska. *Context-awareness micro-architecture for smart spaces*. in *International Conference on Grid and Pervasive Computing*. 2011. Springer.

95. Pan, J., S. Paul, and R. Jain, *A survey of the research on future internet architectures.* IEEE Communications Magazine, 2011. **49**(7): p. 26-36.

96. Shenker, S., *Fundamental Design Issues for the Future Internet, in IEEE Journal on Selected Areas in Communications.* Vol.(13), 2006: p. 1176-1188.

97. Tuominen, L., *Future Internet in the European Union-Case FI-WARE*. 2013.

98. Cirillo, F., et al., *A standard-based open source IoT platform: FIWARE.* IEEE Internet of Things Magazine, 2019. **2**(3): p. 12-18.

99. Mell, P. and T. Grance, *The NIST definition of cloud computing.* 2011.

100. Vaquero, L.M. and L. Rodero-Merino, *Finding your way in the fog: Towards a comprehensive definition of fog computing.* ACM SIGCOMM computer communication Review, 2014. **44**(5): p. 27-32.

101. Heikkilä, V., et al., *Report on FIWARE Platform.* 2020.

102. Andriani, P., et al. *Fiware generic enablers as building blocks of a marketplace for energy*. in *eChallenges e-2015 Conference*. 2015. IEEE.

103. Celesti, A., et al., *How to develop IoT cloud e-health systems based on FIWARE: a lesson learnt.* Journal of Sensor and Actuator Networks, 2019. **8**(1): p. 7.

104. Ahmad, A., *Model-based testing for IoT systems: methods and tools*. 2018, Université Bourgogne Franche-Comté.

105. de Oliveira, Á., M. Campolargo, and M. Martins. *Constructing human smart cities*. in *Smart Cities, Green Technologies, and Intelligent Transport Systems: 4th International Conference, SMARTGREENS 2015, and 1st International Conference VEHITS 2015, Lisbon, Portugal, May 20-22, 2015, Revised Selected Papers 4*. 2015. Springer.

106. Ucuz, D. *Comparison of the IoT platform vendors, microsoft Azure, Amazon web services, and Google cloud, from users' perspectives*. in *2020 8th international symposium on digital forensics and security (ISDFS)*. 2020. IEEE.

107. Swamy, S.N. and S.R. Kota, *An empirical study on system level aspects of Internet of Things (IoT).* IEEE Access, 2020. **8**: p. 188082-188134.

108. Lopes, D.A.R., *Universal Internet of Things System Powered by FIWARE*. 2018, ISCTE-Instituto Universitario de Lisboa (Portugal).

109. Pozo, A., Á. Alonso, and J. Salvachúa, *Evaluation of an iot application-scoped access control model over a publish/subscribe architecture based on fiware.* Sensors, 2020. **20**(15): p. 4341.

110. Llopis, J.A., et al. *MI-FIWARE: A web component development method for FIWARE using microservices*. in *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. 2021. IEEE.

111. Dantas, L.C.C., *Development of support tools for the use of IoT and context FIWARE components*. 2017, Universidade Federal do Rio Grande do Norte.

112. Ribeiro, D.M., *Visualização de dados na Internet.* 2009.

113. Salituro, E., *Learn Grafana 7.0: A beginner's guide to getting well versed in analytics, interactive dashboards, and monitoring*. 2020: Packt Publishing Ltd.

114. Mehdi, A., et al. *"Unleashing the Potential of Grafana: A Comprehensive Study on Real-Time Monitoring and Visualization"*. in *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*. 2023.

115. WŁOSTOWSKA, S., et al., *Comparison of SQL, NoSQL and TSDB database systems for smart buildings and smart metering applications.* Przeglad Elektrotechniczny, 2023. **2023**(11).

116. Reis, D., et al., *Developing docker and docker-compose specifications: A developers' survey*. Ieee Access, 2021. **10**: p. 2318-2329.

117. Ibrahim, M.H., M. Sayagh, and A.E. Hassan, *A study of how Docker Compose is used to compose multi-component systems.* Empirical Software Engineering, 2021. **26**: p. 1-27.

118. Zhang, J., X. Lu, and D.K. Panda. *Performance characterization of hypervisor- and container-based virtualization for HPC on SR-IOV enabled InfiniBand clusters*. in *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 2016. IEEE.

119. Morabito, R. *Power consumption of virtualization technologies: an empirical investigation*. in *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*. 2015. IEEE.

120. Arce, P., et al., *FIWARE based low-cost wireless acoustic sensor network for monitoring and classification of urban soundscape.* Computer Networks, 2021. **196**: p. 108199.

121. Oliveira, R. and C.A. Kamienski. *IoT Redirector: um redirecionador para gerenciamento da heterogeneidade de dados em aplicações IoT*. in *Anais do VI Workshop de Computação Urbana*. 2022. SBC.

122. Fernandes, J., et al., *Social Sensing and Human in the Loop Profiling during Pandemics: the Vitoria application.* arXiv preprint arXiv:2207.01920, 2022.

123. Vaglica, G., F. Bono, and G. Renaldi, *A JRC FIWARE Testbed for SMART Building and Infrastructures.* Publications Office of the European Union: Luxembourg, 2020.

124. Storek, T., et al. *Application of the open-source cloud platform FIWARE for future building energy management systems*. in *Journal of Physics: Conference Series*. 2019. IOP Publishing.

125. Vítor, G., et al., *A scalable approach for smart city data platform: Support of real-time processing and data sharing.* Computer Networks, 2022. **213**: p. 109027.

126. Győrödi, C.A., et al., *Performance impact of optimization methods on MySQL document-based and relational databases.* Applied Sciences, 2021. **11**(15): p. 6794.

127. Zmaranda, D.R., et al., *An analysis of the performance and configuration features of MySQL document store and elasticsearch as an alternative backend in a data replication solution.* Applied Sciences, 2021. **11**(24): p. 11590.

128. Kirešová, S., et al. *Grafana as a Visualization Tool for Measurements*. in *2023 IEEE 5th International Conference on Modern Electrical and Energy System (MEES)*. 2023. IEEE.

129. Győrödi, C., et al. *A comparative study: MongoDB vs. MySQL*. in *2015 13th International Conference on Engineering of Modern Electric Systems (EMES)*. 2015. IEEE.

130. INTEROPERABILITY, I., *Guest editorial semantic technologies in automation systems.* IEEE Transactions on Industrial Informatics, 2017. **13**(6): p. 3335.

131. Kamienski, C. and M. Visoli, *Swamp: uma plataforma para irrigação de precisão baseada na Internet das Coisas.* 2018.

ÖZGEÇMİŞ

Augusto GOMES JÚNIOR; akademik kariyerine Bissau, Gine-Bissau'da başladı. Türkiye'de Lisans eğitimini 2018-2019 yıllarında Çanakkale'de Çanakkale Onsekiz Mart Üniversitesi Bilgisayar Mühendisliği bölümünde mezun oldu. 2021 yılında lisansüstü çalışmalarına devam etmek üzere Karabük'e taşındı ve Karabük Üniversitesi'nde Bilgisayar Mühendisliği Yüksek Lisans programına kaydoldu.