T.C.
NECMETTİN ERBAKAN
ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SCADA SİSTEMİ VE GÖRÜNTÜ İŞLEME TEKNİKLERİ KULLANARAK GERÇEK ZAMANLI AYDINLATMA SİSTEMİNİN TASARIMI

Murat ALTUNKAYA

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

02-2024
KONYA

TEZ KABUL VE ONAYI

Murat ALTUNKAYA tarafından hazırlanan "SCADA SİSTEMİ VE GÖRÜNTÜ İŞLEME TEKNİKLERİ KULLANARAK GERÇEK ZAMANLI AYDINLATMA SİSTEMİNİN TASARIMI" adlı tez çalışması 29/02/2024 tarihinde aşağıdaki jüri tarafından oy birliği ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri **İmza**

Başkan
Dr. Öğr. Üyesi Vahit TONGUR ……..

Danışman
Doç. Dr. Muhammed KARAALTUN ……..

Üye
Dr. Öğr. Üyesi Murat KARAKOYUN ……..

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …./…/20.. gün ve …….. sayılı kararıyla onaylanmıştır.

Prof. Dr. Şerife Yurdagül KUMCU
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Murat ALTUNKAYA

Tarih:

ÖZET

YÜKSEK LİSANS TEZİ

SCADA SİSTEMİ VE GÖRÜNTÜ İŞLEME TEKNİKLERİ KULLANARAK GERÇEK ZAMANLI AYDINLATMA SİSTEMİNİN TASARIMI

Murat ALTUNKAYA

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Muhammed KARAALTUN

2024, 96 Sayfa

Jüri
Danışmanın Doç. Dr. Muhammed KARAALTUN
Dr. Öğr. Üyesi Murat KARAKOYUN
Dr. Öğr. Üyesi Vahit TONGUR

SCADA terimi, İngilizce "Supervisory Control and Data Acquisition" kelimelerinin ilk harflerini temsil eder. SCADA sistemleri tek bir kontrol noktasından, otomasyon sistemlerinin fonksiyonlarını kontrol etmek ve izlemek için kullanılan sistemlerdir. SCADA sistemleri, doğal gaz, petrol, su dağıtım otomasyonları, hava kirliliği kontrolü ve benzeri alanlarda çok başarılı bir şekilde kullanılmaktadır. Bu tez çalışmasında, SCADA sistemi, şehirler arası yollar ile birlikte şehir içi parklar ve sokakların aydınlatma sistemlerinde kullanılmıştır. SCADA sistemi sokak, park ve şehirlerarası yollarında kullanılan aydınlatma armatürlerinin gereksiz zamanlarda çalışmasını engellemiş ve aydınlatma armatürlerinde meydana gelen arızaların tespitini kolaylaştırmıştır. Ayrıca parkların aydınlatma sistemlerinde insanların tespiti için gerçek zamanlı görüntü işleme teknikleri kullanılmıştır. Bu sistemi gerçekleştirmek için şehirler arası yollar ile birlikte şehir içinde bulunan sokaklar ve parklar için bir prototip hazırlanmıştır. Hazırlanan bu prototip donanım ve yazılım olmak üzere iki ana bileşenden oluşmaktadır. Donanım kısmında mikrodenetleyici olarak Arduino Mega2560 Pro Mini kullanılmıştır. Ayrıca TCA9548a Mux modülü, PCD8574 I2C modülü, LDR sensörü, LED ve kamera donanımları sırasıyla pinlerin çoğaltması, ledler ile Mux modülü arasında haberleşmenin sağlanması, ışık şiddetinin ölçülmesi, aydınlatma ve insan tespitinin yapılması için kullanılmıştır. Yazılım kısmında sistemin web tasarımı için ASP.Net MVC, mobil uygulaması için Flutter ve veri tabanı için ise MySQL kullanılmıştır. Tasarlanan sistem, test amaçlı web ve mobil uygulamalar üzerinden izlenmiş ve kontrolleri başarılı bir şekilde sağlanmıştır. Ayrıca tasarlanan sistemde insan ve arızaların tespiti, LDR sensörlerinin çalışması, tüm bilgilerin veri tabanına yazılması ve gerekli alarmların yönetimi başarılı bir şekilde gerçekleştirilmiştir.

Anahtar Kelimeler: Aydınlatma Armatürleri, Görüntü İşleme Teknikleri, Mobil Teknolojiler, SCADA, Web.

ABSTRACT

MS THESIS

DESIGN OF REAL-TIME LIGHTING SYSTEM USING SCADA SYSTEM AND IMAGE PROCESSING TECHNIQUES

Murat ALTUNKAYA

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE IN CUMPUTER ENGINEERING

Advisor: Asso. prof. Dr. Muhammed KARAALTUN

2024, 96 Pages

Jury
Advisor Doç. Dr. Muhammed KARAALTUN
Dr. Öğr. Üyesi Murat KARAKOYUN
Dr. Öğr. Üyesi Vahit TONGUR

The term SCADA stands for "Supervisory Control and Data Acquisition". SCADA systems are used to control and monitor the functions of automation systems from a single control point. These systems have been successfully used in areas such as natural gas, petroleum, water distribution automation, air pollution control, and similar fields. In this thesis study, a SCADA system has been utilized in the lighting system of urban parks and streets, along with intercity roads. The SCADA system has prevented unnecessary operation of lighting fixtures on streets, parks, and intercity roads, and facilitated the detection of faults in lighting fixtures. Additionally, real-time image processing techniques have been utilized for detecting people in park lighting systems. A prototype has been prepared for streets and parks in the city, along with intercity roads to implement this system. This prototype consists of two main components: hardware and software. Arduino Mega2560 Pro Mini microcontroller has been used as the hardware component. Furthermore, TCA9548a Mux module, PCD8574 I2C module, LDR sensor, LED, and camera equipment have been used for pin expansion, communication between LEDs and Mux module, measuring light intensity, and for lighting and human detection, respectively. For the software part, ASP.Net MVC, Flutter and MySQL have been used for web design, mobile application, and database, respectively. The designed system has been monitored through web and mobile applications for testing purposes, and its control has been achieved successfully. Additionally, human and fault detection, operation of LDR sensors, writing all information to the database, and management of necessary alarms have been successfully implemented in the designed system.

Keywords: Image Processing Techniques, Lighting Fixtures, Mobile Technologies, SCADA, Web.

ÖNSÖZ

Tez çalışmam sürecinde bana sağladığı değerli rehberlik, cesaretlendirme ve öğreti dolu anlarla destek veren danışman hocam Sayın Doç. Dr. Muhammed KARAALTUN' a teşekkür ederim. Onun bilgeliği, rehberliği ve öğretici yaklaşımı, bu çalışmanın kalitesini ve içeriğini zenginleştirdi.

Hayatım boyunca hep yanımda olan ve desteklerini benden hiç esirgemeyen aileme ve tez aşamasında bana her konuda destek olan ve sabrını benden esirgemeyen eşim Duygu ALTUNKAYA' ya sonsuz teşekkürlerimi ve sevgilerimi sunarım.

Murat ALTUNKAYA
KONYA-2024

İÇİNDEKİLER

KISALTMALAR

OpenCV: Open Source Computer Vision Library (Açık Kaynak Bilgisayar Görüsü Kütüphanesi)
SCADA: Supervisory Control and Data Acquisition (Gözetleyici Kontrol ve Veri Toplama Sistemi)
LED: Light-emitting diode (Işık Yayan Diyot)
LDR: Light Dependent Resistor (Işığa Duyarlı Direnç)
IR: Infrared Communication (Kızılötesi Haberleşme)
MQ2: Metal Oxide (Metal Oksit)
IoT: Internet of Things (Nesnelerin interneti)
HSR04: Ultrasonik Mesafe Sensörü
GPRS: General Packet Radio Service (Genel Paket Radyo Servisi)
Wifi: Wireless Fidelity (kablosuz internet)
TCP/ IP: Transmission Control Protocol / Internet Protokol
RS232: Recommended Standard 232
Esp8266: Enhanced Serial Peripheral Interface (Geliştirilmiş Seri Çevre Birimi Arayüzü 32 Model Numarasıdır)
ESP32: Enhanced Serial Peripheral Interface (Geliştirilmiş Seri Çevre Birimi Arayüzü 32 Model Numarasıdır)
DHT22: Sıcaklık ve Nem Sensörü
IOS: iPhone Operating System
PERCLOS: Percentage of Eye Closure (Göz Kapanma Yüzdesi)
WEB API: Application Programming Interface
PLC: Programmable Logic Controller (Programlanabilir Mantıksal Denetleyiciler)

ŞEKİLLER LİSTESİ

1. GİRİŞ

Günümüzde hızla büyüyen şehirler; artan enerji talepleri, sürdürülebilirlik ihtiyaçları ve yönetimi konularında daha akıllı sistemler için verimli ve sürdürülebilir çözümlere ihtiyaç duymaktadır. Bu bağlamda, SCADA (Supervisory Control and Data Acquisition) sistemleri, şehirlerin enerji yönetimi ve aydınlatma altyapısını daha verimli bir şekilde yönetmelerine olanak tanıyan daha etkili ve verimli bir teknoloji sunmaktadır.

SCADA sistemleri, birbiriyle bağlantılı ve genellikle coğrafi olarak uzak sistemleri tek bir kontrol noktasından izleyip yönetme yeteneği ile bilinir. Genellikle enerji, su, doğal gaz, fabrika otomasyonu gibi alanlarda kullanılan bu sistemler, uzaktan kontrol, veri toplama ve analiz yetenekleri içermektedir. Buna ek olarak, SCADA sistemleri, karmaşık sistemleri tek bir kontrol noktasından etkin bir şekilde yönetmeyi mümkün kılmaktadır.

Şehirlerin gece aydınlatması, sadece estetik bir önem taşımakla kalmayıp, aynı zamanda güvenlik, enerji tasarrufu ve çevresel sürdürülebilirlik gibi faktörlerle de ilişkilidir. Bu bağlamda, şehirlerde aydınlatma sistemleri önemli bir rol oynamaktadır. Geleneksel aydınlatma yönetiminden uzaklaşarak, SCADA tabanlı bir aydınlatma kontrol sistemi, şehir planlamacılarına daha fazla kontrol ve verimlilik sağlamaktadır. Bu sistemler, şehirleri daha akıllı ve sürdürülebilir hale getirerek enerji verimliliğini artırabilir ve bakım maliyetlerini azaltabilir. Sokaklar, parklar ve şehirlerarası yollar, SCADA tabanlı aydınlatma kontrol sistemleri sayesinde otomatik olarak yönetilebilir hale gelmektedir. Bu durum, enerji kaynaklarını daha etkili bir şekilde kullanmaya olanak tanırken şehirlerin güvenliği ve çevresel sürdürülebilirliği için önemli adımlar atılmasını sağlar. Artan nüfus ve teknolojik ilerlemelerle birlikte şehirlerin enerji ihtiyacı sürekli artığı için SCADA tabanlı aydınlatma kontrol sistemleri, bu enerji talebini daha verimli bir şekilde yönetebilmektedir. Ayrıca şehirlerin aydınlatma sistemlerini daha sürdürülebilir bir enerji politikası ile izlemesine olanak sağlamaktadır. Sokaklardaki ve parklardaki aydınlatma yönetimi ve şehirlerarası yollardaki trafik akışına göre aydınlatma sistemi, SCADA sistemleri ile optimize edilebilir. SCADA tabanlı aydınlatma kontrol sistemleri, şehir planlamacılarına çevresel etkiyi azaltma, enerji verimliliğini artırma ve şehirleri geleceğe taşıma konusunda güçlü bir araç sunar. Bu teknolojik çözümler, şehirlerin aydınlatma ihtiyaçlarına uygun bir şekilde tasarlanabilir.

Bu çerçevede, şehir planlamacıları ve teknoloji uzmanlarının, SCADA sistemlerini şehir aydınlatma sistemlerine entegre etmesi ile şehirlerin sürdürülebilirlik

hedeflerine ulaşmalarını destekleyen yeni nesil aydınlatma kontrol sistemleri ortaya çıkmaktadır. Böylece, gelecek nesiller için daha yaşanabilir, enerji dostu ve teknoloji odaklı şehir aydınlatma modellerinin oluşturulması mümkün olabilir.

1.1. Tezin Amacı

Dünyada gün geçtikçe artmakta olan enerji tüketimi, dünya üzerindeki enerji kaynaklarının sınırlı olmasından dolayı büyük bir sorun teşkil etmektedir. Fosil yakıtlar, kömür, petrol ve doğalgaz gibi geleneksel enerji kaynaklarının sınırlı olmasından dolayı, bu kaynakların tükenmesi büyük bir tehdit teşkil etmektedir. Ayrıca, bu kaynakların çıkarılması ve kullanılması, çevresel sorunlara ve iklim değişikliğine sebep olurken en önemlisi küresel ısınmayla birlikte doğaya zarar vermektedir. Aynı zamanda enerji üretimi ve tüketimi ile çevresel etkilere de yol açmaktadır. Bu etkilerin en belirginlerinden biri, küresel ısınmanın ortaya çıkması ve doğal kaynakların tahrip olmasıdır. Enerji tüketiminin artması, sera gazlarının atmosfere salınmasıyla sonuçlanır. Bu sera gazları, dünya üzerindeki sıcaklığı artırarak küresel ısınmaya neden olmaktadır. Küresel ısınma, dünya üzerindeki iklim sistemlerini ciddi şekilde etkilemektedir. Yükselen sıcaklıklar, deniz seviyelerinin yükselmesine, buzulların erimesine, kuraklık ve sel gibi aşırı hava olaylarına neden olmaktadır. Bu olumsuz etkiler, doğal ekosistemlere ve insan topluluklarına ciddi zararlar vermektedir. İklim değişikliği, sadece çevresel sorunlarla değil, aynı zamanda ekonomik ve sosyal etkilerle de ilişkilidir. Tarım, enerji, su kaynakları ve sağlık sistemleri gibi birçok sektörü etkilemektedir. Ayrıca, enerji üretimi ve tüketimi için gereken kaynaklar, ekonomik açıdan sürdürülebilir olmayan maliyetlere neden olabilir. Bu hem bireylerin hem de işletmelerin bütçesini derinden etkileyen bir durumdur. Ayrıca, enerji kaynaklarının sınırlı olması, enerji güvencesi konusunda endişeleri artırmaktadır. Bu nedenle enerji verimliliğini artırmak ve enerji tüketimini azaltmak hem çevresel hem de ekonomik açıdan büyük bir öneme sahiptir.

Aydınlatma, enerji tüketiminin önemli bir bileşeni olup, özellikle açık hava aydınlatma sistemleri, enerji israfının önemli bir kaynağıdır. Sokaklar, parklar, mesire alanları ve tüneller gibi açık hava alanlarında kullanılan aydınlatma armatürleri, gece boyunca sürekli veya istenmeyen zamanlarda yanmaktadır. Bu durum, enerji maliyetlerini artırmanın yanı sıra, çevresel sorunlara da neden olmaktadır. Armatürlerin istenmeyen zamanlarda yanması gereksiz enerji tüketimine yol açarken, yetersiz aydınlatma da suç artışlarına, kazalara, güvenlik sorunlarına neden olabilir.

Bu güvenlik zaafiyetlerinden bazıları:

Suç Artışı

Yetersiz aydınlatma, suç oranlarının artmasına neden olabilir. Kötü niyetli kişiler, karanlık ve gözetlenmesi zor alanlarda daha fazla cesaret bulabilmektedirler. Bu, hırsızlık, darp ve diğer suçların daha yaygın hale gelmesine yol açabilir.

Kazalar

Yetersiz aydınlatma, kaza ve yaralanma gibi durumların olasılığını artırabilmektedir. Özellikle yollarda ve kamu alanlarında, yetersiz aydınlatma, sürücülerin yaya geçitlerini veya tehlikeli engelleri görmelerini zorlaştırabilir. Bu durum trafik kazalarına, yayaların düşmesine veya diğer kazalara neden olabilmektedir.

Güvenlik Kameralarının Etkinliği

Güvenlik kameraları yetersiz aydınlatma altında düşük kalitede görüntüler kaydedebilir. Bu da güvenlik kameralarının etkinliğini azaltabilir. Çünkü suçları tespit etmek ve suçluları teşhis etmek zorlaşır. İyi aydınlatılmış alanlarda daha net görüntüler elde edilir ve güvenlik kameraları daha etkili hale gelmektedir.

Güvenlik Personeli İçin Tehlike

Yetersiz aydınlatma, güvenlik görevlileri ve diğer güvenlik personelinin işlerini zorlaştırabilir. Kötü niyetli kişilerin yaklaşması veya tehditlerin erken fark edilmesi daha zor hale gelebilir. Bu da güvenlik personelinin güvende olmalarını tehlikeye atabilir.

Aydınlatma sistemlerinde geleneksel olarak kullanılan floresan lambalar ve tasarruflu ampuller gibi aydınlatma teknolojileri, enerji tüketimini artıran diğer faktörlerdir. Bu tür geleneksel aydınlatma yöntemleri, enerjiyi etkin bir şekilde kullanamaz ve çoğu zaman bakım gerektirir.

Aydınlatma sistemlerinin kontrolsüz kullanımı enerji israfını artırmaktadır. Sokak aydınlatma sistemleri genellikle elektrik dağıtım şirketlerinin sorumluluğundadır. Parklar ve mesire alanları ise belediyelerin kontrolünde bulunur. Ancak, bu iki kuruluş için aydınlatma sistemlerinde oluşan eksikliklerin veya sorunların düzenli olarak izlenmesi veya kontrol edilmesi için genellikle bir sistem bulunmamaktadır. İnsanlar, aydınlatma arızalarını şikâyet etmek zorunda kalır. Ancak bu şikayetler bazen gereksiz veya yanıltıcı olabilir. Ayrıca bakım personelinin bu şikayetlere yanıt vermesi aydınlatmadan sorumlu kurum ve kuruşlar için zaman, maddiyat ve ulaşım maliyetlerini dolayısıyla enerji maliyetlerini yükseltir.

Enerji tüketimini azaltmak, enerji maliyetlerini düşürmek ve çevresel etkileri en aza indirmek amacıyla aydınlatma sistemlerinin etkin bir şekilde yönetilmesine yönelik bir strateji geliştirmek gerekmektedir. Bu amaç doğrultusunda, enerji tasarrufunu

artırmak ve enerji tüketimini optimize etmek için gelişmiş aydınlatma teknolojilerinin kullanılması, enerji yönetimi sistemlerinin uygulanması ve bakım süreçlerinin iyileştirilmesi gibi çeşitli stratejiler ve uygulamalar incelenmiştir.

Bu tez ile, enerji israfını azaltma ve enerji verimliliğini artırma konularında farkındalığı artırmak, sürdürülebilir enerji kullanımının teşvik edilmesine önemli bir katkı sağlayacaktır.

1.2. Tezin Önemi

Enerji tasarrufu yapabilmek için kullandığımız enerjinin türüne göre gereksiz kullanımları azaltarak tasarruf sağlanabilir. Enerji tasarrufu sağlayarak hem doğaya hem de ekonomiye katkıda bulunabiliriz. Aydınlatma alanında kullandığımız sistemleri, tasarruf edecek şekilde tasarlamak enerji tasarrufu sağlayacaktır. Bu çalışmada, aydınlatma sistemlerinden olan sokak, park, mesire alanı ve tünellerde kullanılan armatürlerden gerekli olanların en uygun şekilde ve zamanlarda çalışmasını sağlayacak bir tasarımın gerçekleştirilmesi, dolayısıyla sürekli yanmaması gereken armatürlerin istenilen zamanlarda yanmasının sağlanması ve enerji tüketim maliyetlerinin en aza indirilmesi amaçlanmıştır.

Sokak aydınlatmalarındaki armatürlerin kontrolü ve bakımı elektrik dağıtım şirketlerinin, park ve mesire alanlarındaki armatürlerinin kontrolü ve bakımı ise bağlı bulunduğu belediyelerin sorumluluğundadır. Fakat bu iki kurum da armatürlerin arızalarının giderilmesini ve genel bakımlarının yapılmasını şikâyet üzerine veya gelişi güzel bir şekilde yapmaktadır. Bu durum, şirket açısından iş, insan ve ekonomik yükün artmasına neden olurken arızaların sebep olduğu kötü çevre koşulları nedeniyle insanların refah seviyesinin düşmesine neden olmaktadır. Bu kuruluşların; enerji tasarrufu sağlayabilmesi, insan yükünü azaltabilmesi ve insanlar için güvenli çevre koşulları oluşturabilmesi için insan faktörünü ortadan kaldırarak tam otomatik veya yarı otomatik çalışabilen mekanizmaları kullanması gerekmektedir.

Bu tez çalışmasında, armatürlerin uzaktan izlenmesi ve kontrol edilebilmesi için bir SCADA sistemi tasarlanması amaçlanmıştır. Bu SCADA sisteminin, sadece merkezi kontrol odasından izlenmesi ve kontrol edilmesi yerine web ve mobil uygulamalar ile erişilebilecek ve kontrol edilebilecek bir sistem haline getirilmesi amaçlanmıştır. Ayrıca, bu çalışmada armatürlerde kullanılan geleneksel yöntemler yerine daha az enerji tüketen, dayanıklı, uzun ömürlü ve küçük LED kullanılması planlanmıştır.

2. KAYNAK ARAŞTIRMASI

Bu bölümde, akıllı şehir aydınlatma sistemleri ile ilgili daha önce yapılan çalışmalar incelenmiştir. “Internet of Things (IOT) Tabanlı Akıllı Yol Aydınlatma Sistemi” adlı çalışmada enerji tasarrufu sağlamak için IOT tabanlı akıllı sokak aydınlatma sistemi oluşturmuşlardır. Oluşturdukları sistemde sokak lambalarının uzaktan yönetimini sağlayan TCP/IP bağlantısı kullanılarak sistemin Wifi üzerinden denetlenmesi sağlanmıştır. Ayrıca sistemde gerçek bir veri tabanı kullanılmış ve bu sisteme internet üzerinden ulaşan ve kullananlara web tabanlı kontrol ve gözetleme arayüzü sunan ve bir sunucuda çalışan aydınlatma yönetim yazılımı sunulmuştur (Perdahçi vd., 2019).

“Aydınlatma Otomasyon Teknikleri: Cendere Caddesi Örneği” adlı çalışmada yol aydınlatma sistemlerinin kablosuz haberleşme yöntemi ile daha etkin bir şekilde kontrolünün sağlanması amaçlanmıştır. Ayrıca, mevcut koşullara göre armatürlerin loşlaştırma oranına karar vererek armatürleri yönlendirebilen ve enerji tasarrufu sağlayan bir sistem hedeflenmiştir. Çalışmada, armatürler ve bunların bağlı olduğu aydınlatma panoları arasındaki kablosuz haberleşme için ZigBee sistemi kullanılmıştır. Bu çalışma ile İstanbul ili için geliştirilmiş Akıllı Şehir Kontrol Yazılımı (SCM) aracılığıyla yol aydınlatma sistemlerinin daha verimli bir şekilde kontrol edilmesi sağlanmıştır (Çelik vd., 2017).

Özseven ve Sağlam, “IoT Based Street Lighting and Computer Aided Control” adlı çalışmada geleneksel aydınlatma sistemlerindeki uzaktan kontrol, arıza takibi, ekstra iş gücü gibi dezavantajları gidermek için IOT tabanlı akıllı aydınlatma sistemi geliştirilmiştir. Hazırlanan sistemde hareket sensörü ile alınan verilerin yoğunluğuna göre sokak lambalarının ışık şiddeti ayarlanmıştır. Ayrıca, LED armatür sensörlerinden yararlanılarak arıza takibi ve yoğunluk gibi veriler elde edilmiş ve Esp8266 (Wi-fi modülü) ile bu verileri bulut sistemine aktarılmıştır. Mekândan bağımsız hale gelen bulut üzerindeki bu verilerin ThingSpeak platformu üzerinden kontrolleri sağlanmıştır (Özseven ve Sağlam, 2021).

“Kamu Binalarında Mevcut Aydınlatma Elemanlarının LED Aydınlatma Elemanlarına Dönüştürülmesi ile Elde Edilecek Elektrik Enerjisi Tasarrufunun Belirlenmesi” adlı çalışmada aydınlatmada enerji tasarrufu sağlamak için aydınlatma elemanlarının tükettiği enerji incelenmiştir. Günümüzde yaygın olarak kullanılan aydınlatma elemanlarından floresan ve tasarruflu lambaların alternatifi yerine kullanılan LED aydınlatma elemanı ile elektrik enerjisi tasarrufu elde edilebileceğini belirtmişlerdir.

Çalışmalarında elektrik enerjisi tüketiminden yıllık %40 oranında tasarruf edileceği iddia edilmiştir (Yılmaz ve Sungur, 2020).

"Internet of Things Based Parking Lot LED Lighting System" adlı çalışmada otopark aydınlatması için geleneksel aydınlatma ürünleri yerine LED aydınlatma kullanılarak enerji tüketimini azaltmaya yönelik bir sistem sunulmuştur. Bu LED sürücü sisteminin senkron bir şekilde kontrolünü sağlamak amacıyla Nesnelerin İnterneti sisteme dahil edilmiştir. Aydınlatmalardan gelen tüm veriler ESP8266 Wi-Fi modülü ile bulut sistemine aktarılmıştır. Tasarlanmış bir kontrol yazılımı ile buluttaki veriler işlenmiş ve her aydınlatmanın ışık seviyesi bulut üzerine yazılmıştır. Bu çalışma sonucunda verimli bir aydınlatma sistemi oluşturularak gereksiz aydınlatmaların önüne geçilmiş ve arıza durumunda gerekli onarımlar kolay hale gelmiştir (Aydemir, 2019).

"Akıllı Ev Uygulamaları için Yeni Nesil IoT Denetleyici ile Gerçek Zamanlı Uzaktan İzleme ve Kontrol Uygulaması" adlı çalışmada zaman ve enerji tasarrufu sağlayabilmek için IOT tabanlı uzaktan kontrollü bir sistem geliştirilmiştir. Bu sistemde uzaktan haberleşme için ESP32 modülü ile ortamın sıcaklık ve nem bilgisini ölçmek için DHT22 modülü kullanılmıştır. Sistemi kontrol etmek ve verileri uzaktan izleyebilmek için Blynk platformu ile IOS/Android arayüzü tasarlanmıştır. Veri Tabanı olarak ise Blynk IOS/Android geliştiricinin sağladığı bulut sistemi kullanılmıştır. Önerilen akıllı ısıtma sistemi sayesinde oda sıcaklığı ve mobil cihazdan alınan konum bilgisiyle bir odanın ısıtma sisteminin otomatik olarak çalışması ve kapanması sağlanmıştır (Taştan, 2019).

"IoT Tabanlı Platform ile Gerçek Zamanlı İç Ortam Hava Kalitesi İzleme Sistemi" adlı çalışmada iç ortamda hava kalitesinin insan sağlığı için uygunluğunu kontrol edecek IoT tabanlı iç ortam hava kalitesi izleme sistemi gerçekleştirilmiştir. Bu çalışmada Raspberry Pi 3 kontrol kartı, DHT11 sıcaklık ve nem sensörü, MQ2 gaz sensörü ve LDR sensörünü kullanmışlar ve kontrol kartı sayesinde işlenen verileri veri tabanına kaydedilmiştir. Geliştirilen sistemi kontrol edebilmek için de bir web arayüzü tasarlanmış ve anlık olarak bilgilerin kullanıcılara aktarılması sağlanmıştır (Üçgün vd., 2020).

"An Easy to Deploy Street Light Control System Based on Wireless Communication and LED Technology" adlı çalışmada akıllı sokak aydınlatma sistemi geliştirilmiştir. Kablosuz iletişim teknolojilerine dayanan bu sistem ile geleneksel kablolu sistemlerin maliyetlerinin en aza indirmek amaçlanmıştır. Bu çalışmada haberleşmede wifi ve GPRS kullanılmış ayrıca geliştirilen sistem üzerinden veriler veri tabanına kaydedilmiştir (Elejoste vd., 2013).

“Smart Street Light Using Intensity Controller” adlı çalışmada sokak aydınlatmalarının güç tüketimini azaltmak için bir sistem önerilmiştir. Bu sistem LDR (ışığa bağımlı direnç), IR (kızılötesi sensör), pil ve LED kullanılarak tasarlanmıştır. Güç tüketimini azaltmak için lambalarının parlaklığı, oluşturulan bu sistem tarafından yayalar, bisikletliler ve arabalar gibi algılanan nesnelerin hareketinin hızına göre kontrol edilebilmektedir (Abdullah vd., 2018).

“Energy efficient smart Street light” adlı çalışmada şehirde yaşayan gelecek nesillerin refah içinde yaşayabilmesi için gerekli olan akıllı şehir yönetim planı oluşturmak ve enerji tasarrufu sağlayabilmek için bir akıllı sokak lambası tasarımı önerilmiştir. Önerilen bu IOT sisteminde ESP8266 Wi-Fi modülü, ışığa bağlı olan LDR sensörü ve HSR04 ultrasonik sensöründen yararlanılmıştır. Bu sistem ile ortamın ışığının yoğunluğuna ve bir araç veya bir kişinin hareketine göre göre sokak lambasının parlaklığının %60 ile %100 oranında ayarlanması sağlanmıştır (Kodali ve Yerroju, 2018).

“Design and implementation of smart solar LED street light” adlı çalışmada, akıllı sokak aydınlatma sistemi oluşturularak kırsal alanlar ile şehirlerde enerji tasarrufu sağlamak amaçlanmıştır. Sistemde, LED armatür, LED sürücü, PV paneli, şarj denetleyici ışık sensörü, hareket sensörü, Arduino bileşenlerinden yararlanılmıştır. Önerilen bu sistem, gündüz saatlerinde otomatik olarak kapanacak ve yalnızca gece ve şiddetli yağmur veya kötü hava koşullarında çalışacak şekilde programlanmıştır (Bhairi vd., 2018).

“Smart Street Light Management System with Automatic Brightness Adjustment Using Bolt IoT Platform” adlı çalışmada IOT tabanlı sokak lambası kontrol sistemiyle elektrik tüketimini azaltmayı ve insan gücünü en aza indirmeyi amaçladıkları bir sistem tasarlanmıştır. LED, LDR ve IR sensörlerinden yararlanılarak geliştirilen bu sistem ile mevcut sistemler karşılaştırıldığında daha iyi bir performans elde edildiği gösterilmiştir (Sorif vd., 2021).

“SCADA Sistemi: Şehir İçi ve Şehirlerarası Yolların Aydınlatma Sisteminin Kontrolü ve Otomasyonu” adlı çalışmada şehir içi ve şehirlerarası yolların aydınlatma sisteminin izlenmesi ve kontrolü için bir SCADA sistemi önerilmiştir. Irak-Kerkük şehri için hazırlanan bu SCADA uygulaması, kullanıcı yönetimi, grafik yetenekleri, veri tabanı yönetimi, alarm yönetimi ve rapor üretim yönetimi gibi özellikler içermektedir. Aydınlatma direklerinden gelen bilgilerin görsel olarak kullanıcılara yansıtılması ile hataların hızlı ve verimli bir şekilde giderilmesi sağlanmıştır (İbrahim vd., 2022).

“Bir Montaj Parçasının Derin Öğrenme ve Görüntü İşleme ile Tespiti” adlı çalışmada firmaların üretim hatlarında kullanılacak, yurtdışına gönderilen yanlış parça durumlarını azaltarak çıkan ek maliyeti düşürecek, işçinin kodu ezbere bilmemesinden kaynaklanan malzeme arama süresini azaltarak, üretimin verimliliğini arttıracak bir sistem önerilmiştir. Bu çalışmada hatalı durumları önleyebilmek için harici bir kamera aracılığıyla montajlama işlemi sırasında kaç tane montaj işleminin yapıldığını ve montaj sırasında hangi parçaların kullanıldığını tespit etmek için görüntü işleme ve derin öğrenme algoritmaları ve nesneleri tespit etmek için OpenCV kütüphanesi kullanılmıştır. Çalışmanın sonucunda %84 oranında başarılı sonuçlar elde edildiği belirtilmiştir (Çağıl ve Yıldırım, 2020).

“Görüntü İşleme Teknikleriyle Yüz Algılama Sistemi Geliştirme” adlı çalışmada güvenlik sistemlerinde kullanılan yüz tanıma sistemi uygulaması geliştirilmiştir. Bu çalışmada, kamera ile alınan görüntülerden insanların yüz bölgeleri OpenCV kütüphanesi ile tespit edilmiş ve bu yüz fotoğrafları veri tabanına kaydedilmiştir. Yüz profilleri ile veri tabanında bulunan görüntüler arasında benzerlik oranının olup olmadığı belirlenmiştir. Çalışmada yaklaşık %79 oranında bir başarı elde edilmiştir (Eldem vd., 2017.

“Görüntü-Metre ile Görüntü İşleme Tabanlı Mesafe Ölçümü” adlı çalışmada endüstriyel amaçlar için kullanılabilecek bir ölçüm cihazı olan görüntü metrenin geliştirilmesi için teorik alt yapının oluşturulması amaçlanmıştır. Çalışmada yazılım ve donanım birleşenleriyle teorik temellerin işletilmesini hedeflemişlerdir. Geliştirilen elektro mekanik bir sistem ile 1-1000m için yapılan değerlendirmeler neticesinde %0,2’ nin altında başarılı bir oran elde edilmiştir (Yanık ve Turan, 2023).

“Real Time Pedestrian Alert System For Vehicles” adlı çalışmada yoldaki yayaları tespit ederek çarpışmaları önleyen gerçek zamanlı bir yaya uyarı sistemi geliştirilmiştir. Çalışmada derin öğrenme ve transfer öğrenme modelleri kullanılmıştır. Ayrıca, önceden eğitilmiş olan evrişimli modeller ile yayaların ve yolların algılanması sağlanmıştır. Önerilen sistem, çarpışmalardan kaçınmak için kullanıldığında %83.33'lük bir doğruluk oranı sağlamaktadır (Işık vd., 2020).

“Gerçek Zamanlı Sürücü Yorgunluk Tespit Sistemi” adlı çalışmada sürücülerin yorgunluğundan ve uykusuzluğundan kaynaklanan trafik kazalarının önlenmesi amaçlanmıştır. Geliştirilen sistemde, kameralar ile sürücüler anlık olarak izlenerek göz hareketlerine göre analiz yapılmış ve sürücünün uykusuz ve yorgun olduğu tespit edildiğinde sistemin Raspberry Pi 3 sistemi üzerinden alarm vermesi sağlanmıştır. Ayrıca

önceden belirtilen bir hesaba durum ile ilgili görüntülü ve yazılı olarak bilgi gönderimi yapılmıştır (Acar Vural vd., 2018).

"Görüntü İşleme Yöntemleriyle Araç Plakalarının Tanınarak Kapı Kontrolünün Gerçekleştirilmesi" adlı çalışmada güvenliğin yükseltilmesi ve insan gücünden tasarruf edilmesi hedeflenmiştir. Çalışmada giriş kapısında gelen arabanın plakasıyla kapının açılıp kapanmasını sağlayan bir sistem geliştirilmiştir. Geliştirilen sistemin yazılımı ile okunan arabanın plakasını veri tabanından kontrol ederek plakanın veri tabanında olması durumunda kapının açıldığı, olmaması durumunda alarm verdiği belirtmiştir. Çalışmada plakaların %98 bir oranla doğru okunduğu ve %88,1 oran ile başarılı bir sonuç elde edildiği belirtilmiştir (Kürşat Çevik ve Çakır, 2010).

"Gerçek Zamanlı Yüz Tanıma Tabanlı Personel Kontrol ve Takip Sistemi Tasarımı" adlı çalışmada personellerin birbirlerinin yerine kart basmalarının önüne geçilmesi ve personellerin aynı alana parmak izi okutması ile oluşabilecek salgın hastalık zamanlarında sağlık sıkıntısının önüne geçilmesi hedeflenmiştir. Çalışmada kullanıcıların firma çalışanlarının giriş ve çıkışlarını takip edilmesi amacıyla yüz takip sistemi geliştirilmiştir. Geliştirilen sistemde kullanıcıya ait bilgileri ekranda gösterip aynı zamanda veri tabanında saklamışlardır. Geliştirdikleri sistemde çalışanlara ait giriş, çıkış ve mesai saatleri otomatik hesaplanmış ve takip etmeleri sağlanmıştır (Mamak vd., 2020).

3. MATERYAL VE YÖNTEM

Bu çalışmada kullanılan materyal ve yöntemler bu bölümün alt başlıklarında detaylı olarak verilmiştir.

3.1. SCADA Sistemi

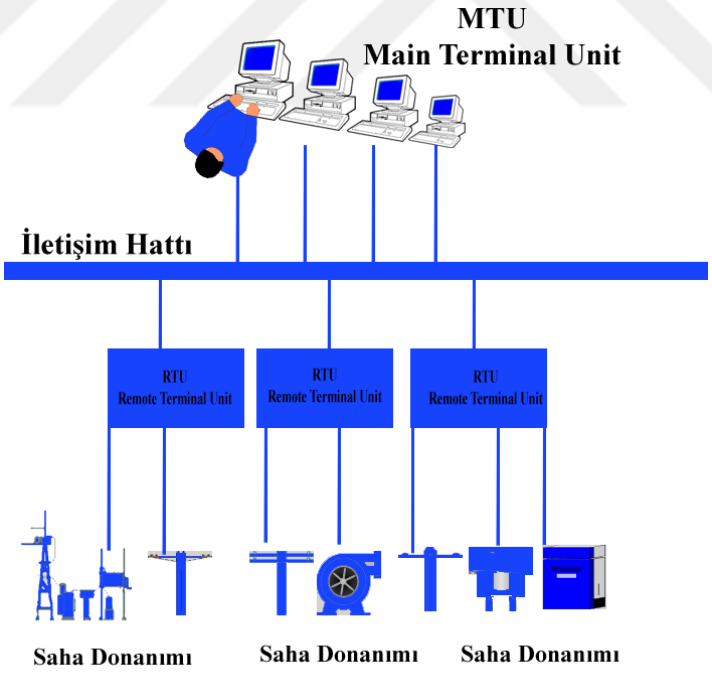
SCADA sistemi, ilk olarak 1950'li yıllarda işlemcilerle birlikte endüstriyel tesislerin kontrolünde kullanılmaya başlanılmıştır. Bu yeni teknolojinin ve denetim kontrolünün başlıca kullanım alanlarından bazıları; kamu hizmetleri, üretim, petrol ve doğal gaz boru hatlarının izlenmesi ve kontrolüdür . 1960'lı yıllarda ise telemetri denilen kavram ortaya çıkmıştır. Telemetri; bir sistemin kablolu veya kablosuz olarak uzaktan izlenmesi veya kontrol edilmesidir. SCADA terimi ise 1970'lerde otomatikleştirilmiş süreçlerin daha büyük oranda izlenmesi ve kontrolü için kullanılan PLC'ler ve mikro işlemcilere sahip sistemleri açıklayabilmek için kullanılmaya başlanmıştır. SCADA sistemi 1980'ler ve 1990'larda, bilgisayar sistemlerinin küçülmesi, Yerel Alan Ağı (LAN) ve HMI yazılımları sayesinde gelişmeye devam etmiştir. SCADA sistemi, geliştiricisinin dışındaki bağlantılara izin vermemekteydi. Bu gelişmelerden sonra SCADA sistemleri, "dağıtılmış SCADA sistemleri" olarak adlandırılan ikinci nesil sistemlere evrilmiştir. Bu teknolojiler sayesinde SCADA sistemi daha fonksiyonlu endüstriyel sistemlerde kullanılmıştır. SCADA sistemi, 2000'li yıllarda ise geliştiriciye özel olmayan iletişim protokolleri ile açık sistem mimarilerini kullanmaya başlamıştır. Gün geçtikçe geliştirilen SCADA sistemi, ağ bağlantılı bir sistem olarak isimlendirilmiştir. Hızla gelişen bilgisayar teknolojileri ile birlikte SCADA sistemi geliştiricileri tarafından temel oluşturan Yapılandırılmış Sorgu Dili (SQL) veri tabanları da kullanılmaya başlanmıştır. Günümüzde, SCADA sistemleri gün geçtikçe gelişen ve değişen teknolojiye ayak uydurmuş ve eski SCADA sistemlerine göre büyük gelişim sağlamıştır. SQL ve web tabanlı uygulamalar gibi modern bilgisayar teknoloji standartlarının özümsenmesiyle SCADA sistemi, gerçek zamanlı işletme bilgilerine dünyanın herhangi bir yerinden ulaşılmasına olanak sağlamıştır. Gelişmiş SCADA sistemi sayesinde kullanıcıların sistem verilerine kolay bir şekilde erişmesiyle, sahada toplanan verilere ve sistem analizlerine dayalı olarak işleyen gelişmiş tesis işlemleri kolaylaşmıştır. Günümüzde en yeni ve güncel altyapısıyla SCADA sistemini kullanan işletmelerde, saha ile etkileşime geçmek çok kolaydır. Uzak mesafelerden bile, fabrikayla kolaylıkla iletişime geçilebilir. SCADA, SQL veri tabanı modelini benimsediğinden,

geçmişte elde edilen veriler hafızaya kaydedilebilir ve tesis süreçlerini iyileştirmek ve bazı endüstriler için zorunlu kayıt tutma uygulamalarında kullanılabilir (“Scada Nedir ?”, 2022).

3.1.1. SCADA Sisteminin Yapısı

SCADA sisteminin yapısını, saha donanımı, merkezi ana birim MTU’lar, uzaktan izleme ve denetleme ünitesi RTU’lar ve iletişim hatları oluşturmaktadır. Şekil 3.1’de SCADA sisteminin yapısı gösterilmektedir.

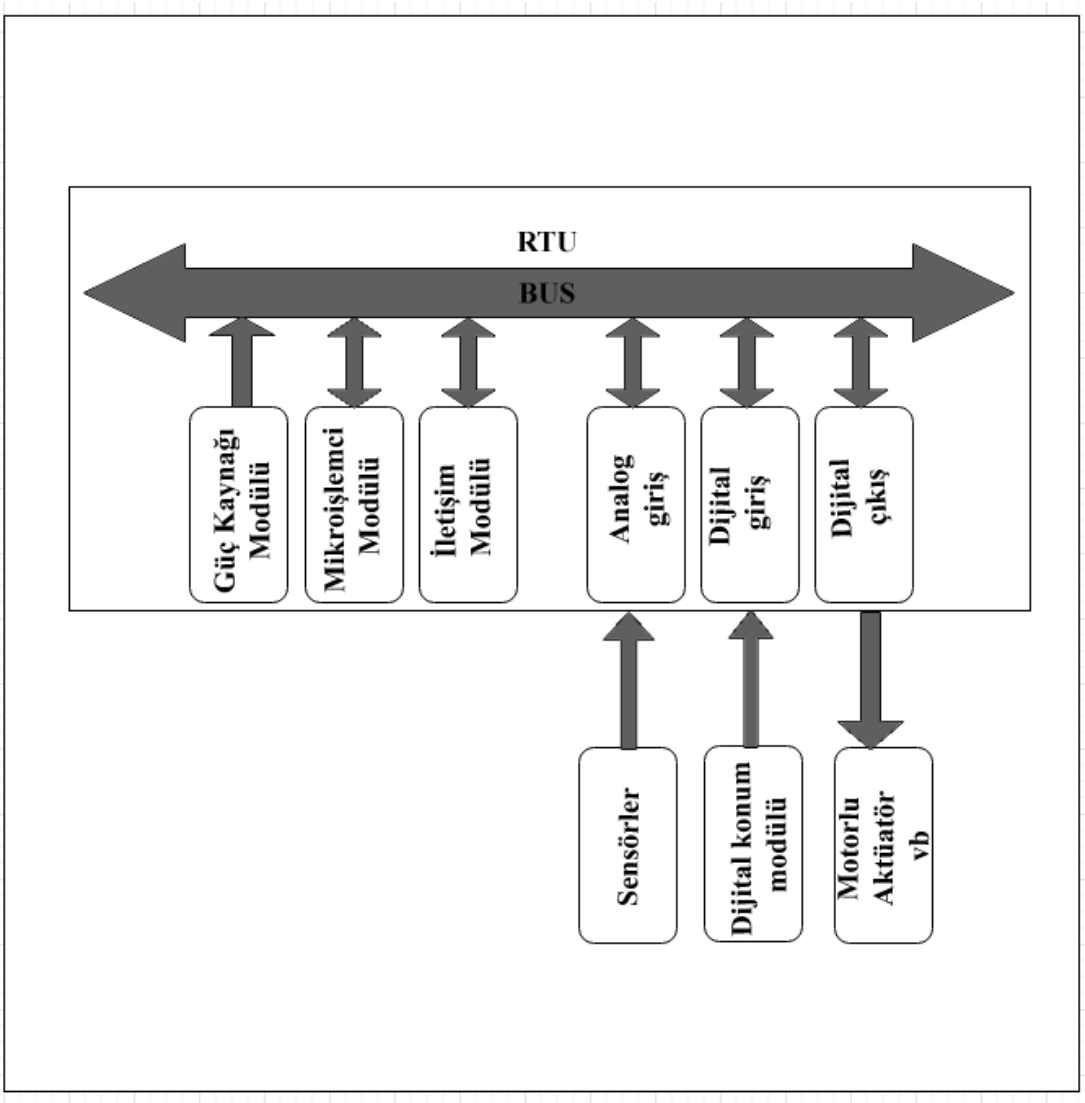
- Uç birim (RTU: Remote Terminal Unit): Veri toplama ve denetleme uç birimlerini gerçekleştiren ünitedir.
- İletişim sistemi (Communication System): Bir noktadan farklı bir noktaya çapraz bir şekilde bilgilerin veya bildirilerin gönderilmesini sağlayan sistemlerdir.
- Kontrol Merkezi Sistemi (MTU – Master Terminal Unit): Yaygın bir coğrafi sahaya dağılmış üretim tesislerinin bilgisayarlar ile uzaktan denetlenebileceği, gözetlenebileceği ve idare edilebileceği bir mekân olarak ifade edilebilir.



Şekil 3.1. SCADA sisteminin yapısı

3.1.1.1. Uzak Terminal Birimi (RTU: Remote Terminal Unit)

Uzak terminal birimi, ait olduğu merkezin sisteminin nesnelerine ait verileri derleyerek biriktirmekte, lazım olduğunda verileri belirtilmiş haberleşme yoluyla kontrol merkezine iletmektedir. Gelen bu verileri gerektiği şekilde uygulayan bir SCADA teçhizat birimidir (Karaçor ve Keleş, 2007). Uzak terminal birimleri, konumlandıkları noktalarda ölçme ve kontrol işlemlerini işleyen birimdir. Şekil 3.2'de, tipik bir RTU diyagramı gösterilmektedir.



Şekil 3.2. Tipik bir RTU diyagram gösterimi (Gündoğdu ve Şahin, 2008)

SCADA sistemleri dahilinde, yerel ölçüm ve denetleme noktaları teşkil eden RTU'lar birbirleriyle iletişim sağlayan farklı aygıtların kontrolünü sağlarlar. Vazifesi, uzak konumdaki süreç aygıtlarını denetlemek, bunlardan gelen bilgileri biriktirmek ve bu

bilgileri SCADA sistemine iletmektir. RTU'un farklı bir vazifesi de ölçülen değerlerin belirlenen limitler arasında kalıp kalmadığını denetleyerek, alarm durumlarını merkeze bildirmektir.

3.1.1.2. Programlanabilir Mantık Kontrolörü (PLC: Programmable Logic Controller)

Endüstriyel süreçlerin otomasyonu için programlanabilen ve özelleştirilebilen PLC'ler, tesis içindeki yerel kontrol görevlerini üstlenmektedir. PLC'ler, saha cihazları ve motorlar gibi cihazlarla iletişim kurmakta ve bu cihazları kontrol etmektedir. PLC hakkında detaylı bilgi, SCADA sistemlerinin temel işlevleri altında bulunan PLC kısmında anlatılmıştır.

3.1.1.3. İletişim Ağı (Communication System)

İletişim, bir noktadan farklı bir noktaya müşterek bir şekilde bilgi veya malumat iletilmesi şeklinde ifade edilebilir (Okay vd., 2019).

SCADA sistemin çalışabilmesi için haberleşme olmazsa olmazdır. Haberleşmede, bilginin toplanması ve denetlenmesinin yanında, iletişim hızının da sistem üzerinde önemli bir etkisi bulunmaktadır.

- **İletişim Ağı:** SCADA sisteminin hız verimini etkileyen kilit nokta haberleşme ağıdır. Denetimi yapılan tesis sistemlerinin, farklı endüstri düzeyinde karşılıklı olarak bağlanan birimler arasındaki bilgi taşınması ve düzenlenmesini kapsayan bütün olaylar iletişim ağları üzerinden yapılmaktadır. Bu tarz sistemlerde iletişim değeri oldukça yüksektir (Okay vd., 2019). Bu tarz bağlantılar, fiziksel bağlantı yönteminin yanı sıra ağ parçalarının coğrafi yerlerine göre yerel (LAN: Local Area Network) veya geniş alan ağları (WAN: Wide Area Network) şeklinde ifade edilmektedirler.
- **LAN:** SCADA sisteme tek bir noktadan yani fabrikanın içinden haberleşme sağlanıyorsa, bu tarz haberleşme bağlantılarına yerel ağ bağlantısı denilmektedir. Bu tür ağlar lokal olduğundan dolayı ufak kapsamlı işlevlerde kullanılır (Gökalp, 2021).
- **WAN:** WAN farklı noktalardaki ve daha geniş mesafelerdeki sistemler arasında bağlantı sağlamaktadır.
- **SPI:** Serial Peripheral Interface full duplex modda çalışan senkron bir seri veri bağlantısı standardıdır.

- **UART:** Evrensel asenkron alıcı / verici anlamına gelir ve iki cihaz arasında seri veri alışverişi için bir protokol veya kurallar kümesini tanımlamaktadır (“Universal asynchronous receiver/transmitter (UART)”, 2006).
- **I2C:**I2C kısa mesafeli seri veri aktarımına izin vermek için tasarlanmış veri yolu protokolüdür (Gasperi ve Hurbain, 2010).
- **CAN Bus:** CAN BUS hattının görevi, elektronik kontrol üniteleri arasındaki iletişimi (veri alışverişini) sağlamaktır. Otomobillerde ve diğer teknolojik araçlarda, merkezi bir hal almıştır (de Andrade vd., 2018).
- **İletişim Protokolleri**

  Araçlar arasındaki bağlantılar kablolar vasıtası ile sağlanmaktadır. Birçok uygulamada uygulanabilecek standart bir arabirim vardır. İletişimde geçerli iki mühim arabirim RS-232 ve RS-485’tir (Okay vd., 2019).
    - RS-232 iki konum arasında iletişim için kullanılır.
    - RS-485 iki veya daha çok konum arasında iletişim için kullanılır. RS-232 ‘e göre daha uzun mesafelerde kullanır.

  RS-232 ve RS-485 arabirimler, gözetleme ve denetleme mekanizmalarında kullanılmaktadırlar.

  - **İletişim Bağlantıları**

  Merkez ile Uzak Terminal Üniteleri arasında bulunan ve Uzak Terminal Ünitelerinin kendi aralarında bulunan haberleşme için kullanılacak aygıtlar, oluşturulacak ağ ’a göre değişkenlik gösterebilir. Aşağıda bazı örnekler verilmiştir:
    - Enerji taşıma hatları
    - Kiralanmış PTT telefon hatları
    - Kablolu TV hatları
    - Radyo frekansında iletişim
    - Fiber optik

3.1.1.4. Kontrol Merkezi Sistemi (MTU – Master Terminal Unit)

Ana terminal üniteleri, SCADA sisteminin merkezi veya bilgisayarı şeklinde adlandırılmaktadır. MTU’da, Teknikerler, idareciler ve bakım personelleri faal olarak sahada yer alan sistemleri bir merkezden bilgisayarlar ile kontrol eder ve izleyebilirler (Çelikpençe, 2016).

Ana Terminal Üniteleri, SCADA sisteminde geniş coğrafi alana yayılmış RTU'ların düzenli hizmet vermesini, RTU'lardan gelen verilerin değerlendirilmesini, sistem personeline gösterilmesini ve ilaveten sistemden gelen komutları RTU'lara yayınlayarak merkezin kontrol işlevini sağlamaktadır. MTU'nun görevleri aşağıda özetlenmiştir:

- RTU'dan gelen bilgilerin toplamak
- Elde edilen bilgileri yazılımla beraber işledikten sonra monitörlere ya da yazıcıya iletmek
- SCADA'da denetlenecek aygıtlara denetleme komutunu iletmek
- Öbür bilgisayar mekanizmaları ile haberleşme sağlamak

3.1.2. SCADA Sistemindeki Veriler ve Amaçları

SCADA sistemleriyle aşağıda örnek olarak verilen bilgiler elde edilmektedir.

3.1.2.1. Sistem Bilgileri

Üretim tesisindeki işlemlerin ne kadar zaman aldığı, işlemler sırasında tesisin ne kadar durduğu, durmaların nerelerden kaynaklandığı gibi bilgileri içermektedir.

3.1.2.2. Üretim Verimliliği Bilgileri

Üretim tesisinde üretilen ürünün toplam miktarı, üretimin toplam durma süresi ve nedenleri gibi bilgilerdir.

3.1.2.3. Üretim Maliyetleri Bilgileri

Üretim tesisinde üretilecek ürün, kullanılacak mamul ve yarı mamul miktarları, güç tüketimi, üretim aşamalarında meydana çıkan maliyetler gibi bilgilerdir.

3.1.2.4. Bakım Bilgiler

Üretim tesisinde, üretim hatlarındaki arıza sebepleri, arıza süreleri ve tamir süreleri gibi bilgilerdir.

3.1.2.5. Çalışanların Kontrolü

Üretim tesisindeki üretim hatlarında çalışan işçilerin tespit edilmesi gibi bilgilerdir.

3.1.2.6. Üretilen Mamullerin Kodlanması-Geçmişe Dönük Veri Edinme

Üretim tesisinde üretilen mamullerin hangi hatta yapıldığı ve bu hat ile ilgili bilgiler ile üretilen ürünlerin ilişkilenmesi gibi bilgilere ulaşılabilir.

3.1.2.7. Alarm

Üretim hatlarındaki süreçler hakkındaki verilerin her an gözlenmesi gerekmekte ve bu veriler istenmeyen durumlarda, alarm durumu SCADA sistemi aracılığıyla yetkililere bildirilmektedir. SCADA sistemi, alarmların çeşitli ayarların gerçekleştirildiği ve teknisyeni bilgilendirmek üzerine oluşturulan bir sisteme sahiptir (Karaçor ve Keleş, 2007). Alarmlar katalog halinde olacağı gibi sıralamanın önemine göre sınıflanarak diyagram biçiminde de gösterilebilir.

3.1.2.8. İstatistiksel Bilgiler

Üretim tesisinde üretilen ürünlerin toplam sayısı, kusurlu ürünlerin sayısı ve kusurlarının neler olduğuyla ilgili bilgilerdir. SCADA sistemi, üretim tesislerinde değişik vardiyalarda üretilen ürün verilerini, üretim sürecinin belirli değişkenlerini istenildiğinde veya periyodik olarak raporlar (Karaçor ve Keleş, 2007). Raporlanan bu verileri, üretim tesisinin isteklerine göre değişik şekillerde hazırlamak mümkündür.

SCADA sistemi, üretim tesislerindeki kontrol cihazları ve sensörlerden toplanan tüm bilgileri veri tabanında tutmaktadır. Toplanan bu veriler işletmenin istediği şekilde düzenlenip işletmeye sunulur. Bu veriler aşağıdaki şekillerde kullanılabilirler:

- SCADA sistemdeki üretim verilerinin grafik animasyon olarak gösterilmesi
- Elde edilen verilerin, belirlenen değerler dışına çıkması durumunda alarmların oluşturulması
- Üretim tesisinden toplanan verilerin, veri tabanına kaydedildikten sonra geriye dönük ve istatistiksel olarak kullanılabilmesi
- Raporlama işleminin olması
- Üretim tesisinden gelen verilerin gerçek zamanlı izlenilebilmesi ve değişikliklerin değişme zamanlarıyla kaydedilmesi
- SCADA sistemine reçetelerin yetkilendirmeye göre ayarlanabilmesi

3.1.3. SCADA Yazılımında Umulanlar

- Çabuk ve basit uygulama dizaynı
- Kullanıcı ara yüzü

- Diyagram gösterimi
- Alarm yönetimi
- RTU arayüzü
- Veri Tabanı
- Yetkilendirme
- Raporlama
- İstemci/server dağıtımlı süreç
- Ölçeklenebilirlik

3.1.4. SCADA Sisteminin Temel Elemanları

3.1.4.1. Veri Toplama Üniteleri

Kontrol üniteleri, endüstriyel sistemlerin önemli birimlerinden biridir. Kontrol üniteleri, kontrol odası düzeyinde farklı destek işletmelerin kontrol birimlerinden işletme ve idare seviyesine kadar tüm veri ve bilgileri üst düzey hızlarda işleyecek bir yapıdadır. SCADA sisteminde elde edilen bilgilerin değerlendirilmesi, monitörlerde gösterilmesi ya da diske eklenmesi için PC'ye aktarılması gereklidir (MEGEP, 2007).

3.1.4.2. Programlanabilir Lojik Denetleyiciler (PLC)

İlk olarak otomasyon endüstrisinin ihtiyaçlarına yanıt vermek amacıyla geliştirilen PLC'lerin tarihi 1960'lara kadar gitmektedir. Bu cihazlar, endüstriyel röle panolarının yerini almaya başlamış ve daha esnek ve programlanabilir bir kontrol sağlamışlardır. İlk PLC'ler, genellikle endüstriyel röle mantığına dayanıyordu, ancak zamanla daha karmaşık ve işlevsel hale gelmişlerdir. Günümüzdeki PLC'ler, mikrodenetleyici teknolojisi ve gelişmiş yazılım ile donatılmıştır.

PLC endüstriyel otomasyon uygulamalarında kullanılan bir kontrol cihazıdır. Bu cihazlar, karmaşık süreçleri ve makine kontrolünü gerçekleştirmek için programlanabilmekte ve endüstriyel tesislerde yaygın olarak kullanılmaktadır. PLC sistemleri, giriş sinyallerini işler, çıkışlara komutlar gönderir ve genellikle otomasyon ve veri toplama işlevleri sunarlar. PLC'lerin uygulama alanları ve PLC yapısı aşağıda verilmiştir:

- **PLC'nin Uygulama Alanları**

  Üretim hattı kontrolü

  Pompa ve motor kontrolü

Aydınlatma sistemleri
Trafik sinyali kontrolü
Asansörler ve yürüyen merdivenler
Kimya endüstrisi işlemleri
Gıda işleme tesisleri
Enerji dağıtımı ve kontrolü

- **PLC'nin Yapısı**

Programları çalıştıran, mantıksal kararlar alan ve giriş sinyallerini işleyen Merkezi İşlem Birimi (CPU), PLC'nin beyni olarak isimlendirilir. CPU, işlemci hızı, bellek kapasitesi ve işlem gücü gibi önemli özelliklere sahiptir.

Giriş birimi, PLC'ye farklı algılayıcılardan ve cihazlardan gelen giriş sinyallerini kabul eder. Bu sinyaller genellikle dijital veya analog formda olabilir.

Çıkış birimi, PLC tarafından verilen komutları alır ve bağlı cihazları kontrol eder. Bu çıkışlar, motorlar, valfler, ışıklar gibi endüstriyel cihazları etkileyebilir.

Bellek elemanları, kullanıcı tarafından yazılan kontrol programını depolar. Bu program, giriş sinyallerini izleyen ve çıkışlara komutlar gönderen mantıksal bir kontrol işlemi tanımlar.

Programlama birimi, kullanıcıların PLC'yi programlamak için kullandığı araçları içerir. Bu, genellikle bir bilgisayar yazılımı veya özel bir programlama cihazı olabilir.

Genişletme birimi, PLC sistemini gerektiğinde daha fazla giriş ve çıkışla genişletme olanağı sağlar. Bu, sistemlerin ölçeğini büyütmek için kullanılır.

PLC'ler, endüstriyel otomasyon ve süreç kontrolünün vazgeçilmez bir parçasıdır. Endüstriyel tesislerin verimliliğini artırmak ve süreçleri optimize etmek için yaygın olarak kullanılırlar. PLC'ler, endüstri 4.0 ve akıllı üretim trendlerine uyum sağlamada önemli bir rol oynarlar.

SCADA sistemlerinde kullanılan PLC'ler, kumanda ile devreleri kontrol edebilecek sayıda giriş ve çıkış birimlerine sahiptir. Bunun yanı sıra, haberleşme arabirimleri ile kuşatılmış denetleme yapısına elverişli mekanizma uygulaması altında işleyen endüstriyel bir aygıttır (Karataş vd., 2018). PLC'ler, toplanan veri ile malumatları

SCADA'ya aktarırken diğer taraftan yazılıma uygun olarak lojik kontrol denetimi sağlamaktadır.

3.1.4.3. Veri Toplama (Data Acquisition – DAQ) Modülleri

SCADA sisteminin temeli, iyi bir kontrolün sağlanması ve verinin toplanmasına dayanmaktadır (Karataş ve Erçetin, 2018).

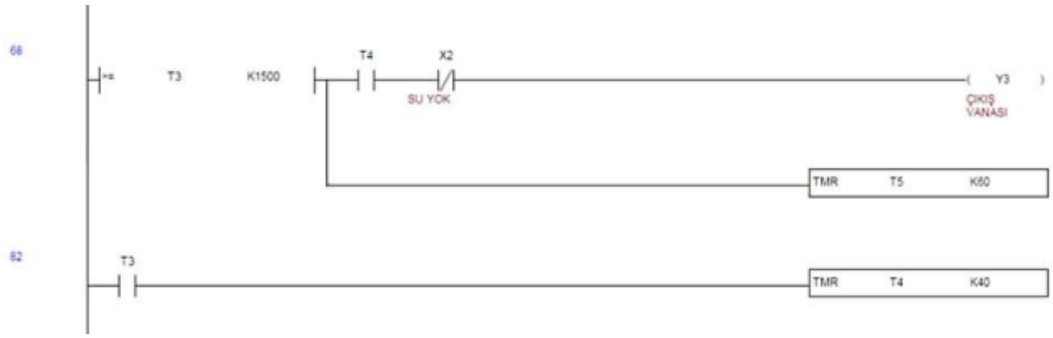
3.1.4.4. Sensörler ve Algılayıcılar

Üretim tesislerinde kullanılan SCADA sistemlerinin en alt noktasını oluştururlar. Sensörler ve algılayıcılar, fiziksel elemanlar olup yerel denetleyicilerdir. SCADA sistemlerinde gereksinimlere uygun olarak sensör ve algılayıcılar eklenmektedir.

3.1.4.5. Yazılım

Yazılım, elektronik cihazların istenilen işi yapmasına olanak veren uygulamalara verilen isimdir (Karataş ve Erçetin, 2018). Cihazların haberleşmesi ve sistemin görevlerini yerine getirmesi için kullanılan ortak makine dili komutlarıdır.

PLC'ler, mantık olarak Arduino, PIC ve diğer mikro denetleyicilere benzemektedirler. Kumanda ve kontrol elemanları olan PLC'ler, farklı firmalar tarafından üretildikleri için faklı diller ile yazılabilmektedirler. PLC'lerin programlanması yapılırken zamanlama, sayma, taşıma, mantık işlemleri, bellek fonksiyonlarına ihtiyaç duyulmaktadır. Bu ve bunun gibi birçok fonksiyon PLC'lerin içerisinde kendi hafızasına bağlı olarak yer almaktadır. PLC yazılımları sahadaki kontrolleri sağlamak için sıkça kullanılmaktadır (Kılıç ve Özdemir, 2010). Şekil 3.3'te, PLC programlama görüntüsü gösterilmektedir.



Şekil 3.3. PLC programlama görüntüsü (Gencer vd., 2020)

RTU, endüstriyel alanda kullanılan iletişim protokolüdür. SCADA sistemleri dahilinde yerel ölçüm ve denetleme noktalarını teşkil eden RTU'lar, birbirlerine iletişim sağlayan farklı aygıtların kontrol ve kumandasını ederler. Vazifesi uzak konumdaki süreç aygıtlarını denetlemek, bunlardan gelen bilgileri biriktirmek ve bu bilgileri SCADA sistemine iletmektir. RTU'un farklı bir vazifesi de ölçülen değerlerin belirlenen limitler arasında kalıp kalmadığını denetleyerek, ters bir durumda alarm durumunu merkeze bildirmektir. RTU'lar sahadaki kontrolleri sağlamak için kullanılmaktadır.

3.1.4.6. Merkezi Kontrol Odası

Yaygın bir coğrafi sahaya dağılmış üretim tesislerinin bilgisayarlar ile uzaktan denetlenebileceği, gözetlenebileceği ve idare edilebileceği bir mekân olarak ifade edilebilir. Üretim tesisinin merkezi bir noktasına merkezi kontrol odasının yerleştirilmesi ve orada kurulması gerekir (Okay vd., 2019). Merkezi kontrol odasında; ağ, SCADA paket uygulamaları, terminaller, yazıcı ve alarm için siren vb. yardımcı donanımlara ihtiyaç vardır.

Arıza durumlarında, arızanın nerede olduğu kontrol odasındaki ekranlarda gösterilmektedir. Merkezi kontrol odasının kritikliğinden dolayı, statik elektriğe karşı yalıtılmış olması gerekmektedir.

3.1.4.7. Kontrol Panoları

Sistemdeki programlanan kontrol kartlarını içeren panolardır. Aynı zamanda, kontrol panoları, tesisin mimik diyagramına ve sistemde oluşabilecek bir sorunun sinyal lambaları ile gösteren şemalara sahiptir. Kontrol sisteminde, kontrol birimleri ve alçak gerilim aygıtlarının yerleştirilmesi bu panolarda yapılmaktadır. Şekil 3.4'te, kontrol panosu gösterilmektedir.

Şekil 3.4. Kontrol panosu görüntüsü ("Yüksek Gerilim Kontrol & Scada & Mimik Panosu", 2022)

3.1.4.8. SCADA Sistem Terminalleri

SCADA sistemi, kullanıcı ara yüzünde farklı personellere kullanma ve erişme olanağı vermektedir. Bu uç birimler, teknikerlerin sistemi gözetlemesini ve denetlemesini sağlamaktadır. Sistemin denetimi, kullanıcı adı ve şifre ile olmaktadır.

3.1.4.9. Bilgisayar Monitörleri

Tesisteki mimik diyagramının devamlı izlenmesi gerekir. İzlenme işleminin yapılması için bilgisayar monitörlerine ihtiyaç duyulmaktadır. Bu monitörlerin canlı renkleri desteklemesi ve yüksek çözünürlükte olması gerekmektedir.

3.1.4.10. Yazıcılar

Üretim tesisinde bulunan SCADA sistemindeki olay ve sorunların durumlarını rapor halinde yazdırma olanağı sağlamaktadır.

3.1.4.11. Kesintisiz Güç Kaynağı

Üretim tesisinin ürettiği ürüne bağlı olarak tesisin elektrik kesinti durumlarında ne kadar elektriğe ihtiyaç duyacağının belirlenmesi gerekmektedir. Ayrıca, bu gerekliliklerin yanı sıra sistemde bulunan bilgisayarlar ve çevredeki teçhizatların kesintisiz akım sağlayacak bir AC veya DC güç kaynağı bulundurulması gerekmektedir.

3.1.4.12. Veri Tabanı

SCADA sistemleri, endüstriyel süreçleri izlemek, kontrol etmek ve verileri toplamak için kullanılan kritik sistemlerdir. Bu sistemlerin temel bir bileşeni, toplanan verileri depolamak ve erişmek için kullanılan veri tabanlarıdır. SCADA sistemlerinde kullanılan veri tabanları, genellikle endüstriyel kontrol uygulamalarına özgü ihtiyaçları karşılamak üzere tasarlanmıştır. SCADA sistemlerinde yaygın olarak kullanılan bazı veri tabanları aşağıda verilmiştir:

- MySQL, açık kaynaklı bir veri tabanı yönetim sistemidir ve SCADA uygulamaları için ideal bir seçenektir. Hafif yapısı ve geniş topluluğu sayesinde, endüstriyel kontrol sistemlerinde kullanılmak üzere özelleştirilebilir ve uygun maliyetli çözümler sunabilir (“MySQL Documentation”, 2024).
- Oracle Database, büyük ölçekli endüstriyel uygulamalarda güvenilir performans sağlayan bir veri tabanıdır. Yüksek veri güvenliği ve işlem kapasitesi, SCADA sistemlerinde geniş çaplı veri yönetimi gereksinimlerini karşılamak için tercih edilir. 1970’li yıllarda, Oracle’ın büyük katkılarıyla SQL IBM’de geliştirilmiştir (“Veritabanı Nedir?”, 2023).
- PostgreSQL, açık kaynaklı bir nesnesel-relasyonel veritabanı yönetim sistemidir. SCADA sistemlerinde kullanılabilirliği, geniş veri tipleri desteği ve genişletilebilir mimarisi, endüstriyel kontrol uygulamalarında veri tabanı olarak tercih edilmesini sağlar. Aynı zamanda veri bütünlüğünü korur ve pek çok platform ile uyumludur. Ayrıca, SQL standartları ile uyumludur (“PostgreSQL Nedir? Ne İşe Yarar?”, 2023).
- MongoDB, belge tabanlı ve NoSQL bir veritabanıdır. JSON benzeri belgeleri depolar ve bu, özellikle büyük ve dağınık veri setleriyle çalışan uygulamalarda tercih edilir. Özellikle, hız gerektiren ve geleneksel ilişkisel veri tabanlarının hantal ve yavaş kaldığı yapılarda kullanılmaktadır. SCADA sistemlerinde, çeşitli sensör verilerini depolamak ve esnek bir veri modeline ihtiyaç duyulan durumlarda kullanılabilir (“MongoDB”, 2023).
- SQLite, hafif bir gömülü veritabanıdır ve tek bir dosya içinde çalışabilir. Genellikle küçük ve orta ölçekli projelerde kullanılır. SCADA sistemlerinde, daha küçük ölçekli uygulamalarda veya yerel depolama ihtiyacı olan durumlarda tercih edilebilir.

- Azure Cosmos DB, çok modlu ve küresel ölçekte dağıtılmış bir veritabanı servisidir. Yapılandırılabilir konsistensi destekler ve birden fazla API'yi destekler. SCADA sistemlerinde ölçeklenebilir ve yüksek performanslı veritabanı ihtiyacını karşılamak için kullanılabilir (“Azure Cosmos DB – MS Azure Turkey”, 2017).
- DynamoDB, Amazon Web Services (AWS) tarafından sağlanan hızlı ve esnek bir NoSQL veritabanıdır. Özellikle bulut tabanlı SCADA uygulamalarında ve IoT projelerinde tercih edilebilir (“Amazon DynamoDB’nin Özellikleri”, 2023).

SCADA sistemi, saha cihazlarından gelen verileri toplar, bu verileri işler, analiz eder ve gerektiğinde operatörlere veya diğer otomasyon sistemlerine yanıt verir. Operatörler, SCADA arayüzü üzerinden süreçleri izler ve kontrol eder. Ayrıca, sistem, tanımlanmış eşik değerlerin aşılması durumunda otomatik olarak alarm ve bildirimler gönderebilir.

Sonuç olarak, SCADA sistemi, endüstriyel tesislerdeki süreçlerin etkili bir şekilde izlenmesi ve kontrol edilmesi için kritik bir rol oynar. Bu sistem, endüstriyel otomasyonun temel taşlarından biridir ve verimliliği artırırken güvenlik sağlar. SCADA sistemi, farklı endüstrilerde ve tesislerde geniş bir uygulama yelpazesi sunar ve sürekli olarak geliştirilmeye devam eder.

3.2. Python Yazılımı

Python'un hikayesi, Guido van Rossum'un Hollanda'da bir programlama dili geliştirme projesine katıldığı 1980'lerin sonlarına kadar uzanmaktadır. Guido, 1989 yılında ABC adlı bir dilin geliştirilmesine katılmış ve bu deneyim, Python'u tasarlarken ona ilham vermiştir. Python'un ilk sürümü olan Python 0.9.0 piyasaya 1991 yılında sürülmüştür.

3.2.1. Python Yazılımının Temel Özellikleri

- Basit ve Okunabilir Söz Dizimi: Python'un en bilinen özelliklerinden biri, okunabilir bir söz dizimine sahip olmasıdır. Bu; kodun yazılması, anlaşılması ve bakımının yapılmasını kolaylaştırır.
- Yüksek Düzeyde Bir Dil: Python, yüksek düzeyde bir dildir. Bu özellik, karmaşık işlemleri daha az kodla gerçekleştirmenizi sağlar.
- Çok Amaçlı: Python; web geliştirme, veri analizi, yapay zeka, bilimsel hesaplama, oyun geliştirme ve daha birçok farklı alanlar için kullanılabilir.

- Geniş Kütüphane ve Çerçeve Desteği: Python, birçok standart ve üçüncü taraf kütüphane içerir. Bu kütüphaneler, geniş bir işlevselliğe sahiptirler ve özelleştirilebilirler.
- İşletim Sistemleriyle Uyumlu: Python; Windows, MacOS ve Linux gibi farklı işletim sistemlerinde sorunsuz bir şekilde çalışabilir.
- Topluluk Desteği: Python, büyük ve aktif bir kullanıcı topluluğuna sahiptir. Bu topluluk; sorunların çözümü, kütüphanelerin geliştirilmesi ve dilin sürekli iyileştirilmesi için katkı sağlar.

3.2.2. Python Yazılımının Kullanıldığı Alanlar

- Web Geliştirme: Python, Django ve Flask gibi çerçevelerle web uygulamaları geliştirmek için kullanılır.
- Veri Bilimi ve Analitik: Python, veri analizi ve makine öğrenimi için popüler bir seçenektir. Kütüphaneler arasında; NumPy, Pandas ve SciPy gibi önemli araçlar bulunur.
- Yapay Zeka ve Makine Öğrenimi: Python; TensorFlow, Keras ve PyTorch gibi kütüphanelerle yapay zeka ve makine öğrenimi projelerinde sıklıkla kullanılır.
- Bilimsel Hesaplama: Bilim alanında çalışanlar, Python'u matematiksel modellemeler ve simülasyonlar için tercih ederler. SciPy ve Matplotlib gibi kütüphaneler, bu tür projeleri destekler.
- Oyun Geliştirme: Python, Pygame ve Unity gibi oyun geliştirme çerçeveleriyle oyunların oluşturulmasında kullanılır.

3.3. Flutter

Flutter, Google tarafından geliştirilen açık kaynaklı bir mobil uygulama geliştirme çerçevesidir. Modern, taşınabilir ve hızlı uygulamalar oluşturmak için kullanılır. Flutter'ın ne olduğu, temel özellikleri ve kullanım alanları aşağıda açıklanmıştır.

Açık kaynaklı bir proje olan Flutter 2017 yılında Google tarafından geliştirilmiştir. Özellikle, mobil uygulama geliştirme projelerinde kullanılır. Flutter, tek bir kod tabanı ile hem iOS hem de Android platformlarına yönelik uygulamalar geliştirmenizi sağlar. Dart programlama dili kullanılarak geliştirilmiş olan Flutter, hızlı ve sorunsuz performans sağlar.

3.3.1. Flutter Temel Özellikleri

- **Hız ve Performans:** Flutter, derlenmiş bir dil olan Dart ile geliştirildiği için yüksek hız ve performans sunar. Aynı zamanda, "sıcak yeniden yükleme" özelliği sayesinde, geliştiriciler hızla değişiklikleri gözlemleyebilir ve uygulamalarını hızlıca güncelleyebilirler.
- **Mobil, Web ve Masaüstü Uygulamalar:** Flutter, sadece mobil uygulama geliştirmek için değil, aynı zamanda web ve masaüstü platformlarını geliştirmek için de kullanılabilir. Bu özellik, geliştiricilere uygulamalarını farklı platformlara genişletme esnekliği sunar.
- **Birleşik Kod Tabanı:** Flutter, tek bir kod tabanı ile birden fazla platform için uygulama geliştirmenizi sağlar. Bu, geliştirme sürecini daha etkili hale getirir ve bakımı kolaylaştırır.
- **Görsel ve Anlatısal Öğeler:** Flutter, özel widget'lar ve görsel öğeler sağlar. Bu özellik, uygulamaların çekici ve özgün olmasını sağlar. Ayrıca, dilin bağımsız olmasından dolayı, kullanıcı arabirimi tasarımı için her iki platformda da aynı görünüm elde edilebilir.
- **Geniş Kütüphane Desteği:** Flutter, zengin bir kütüphane koleksiyonuna sahiptir. Bu; veri tabanları, harita hizmetleri, animasyonlar ve daha fazlası için kullanabileceğiniz birçok aracı içerir.

3.3.2. Flutter Kullanım Alanları

- **Mobil Uygulamalar:** Flutter, iOS ve Android platformları için taşınabilir ve hızlı uygulamalar geliştirmek isteyen geliştiriciler için ideal bir seçenektir. Özellikle işletmeler, markalar ve girişimciler tarafından tercih edilir.
- **Web Uygulamaları:** Flutter, web uygulamaları geliştirmek isteyenler için de kullanılabilir. Bu sayede, web tabanlı projeler hızlıca hayata geçirilebilir.
- **Masaüstü Uygulamalar:** Flutter; Windows, MacOS ve Linux platformlarına yönelik masaüstü uygulamalar geliştirmek için kullanılabilir. Bu, özellikle yazılım geliştiricileri ve şirket içi uygulama gereksinimlerini karşılamak isteyen işletmeler için önemlidir.
- **Oyun Geliştirme:** Flutter, oyun geliştirme konusunda da kullanılabilir. Oyun geliştiricileri, Flutter'ı özellikle hızlı prototip oluşturmak ve oyunlarını farklı platformlara taşımak için kullanırlar.

3.3.3. Flutter’da Kullanılan Komponentler

- **Stateless Widget (Durağan Widget):** Stateless widget'lar, uygulamadaki görünümlerin veya bileşenlerin değişmeyen (durağan) bölümlerini temsil eder. Bu widget'lar genellikle uygulama arayüzünün statik kısımlarını oluşturmak için kullanılır. Örneğin, bir metin veya düğme, stateless widget olarak oluşturulabilir. Bu widget'lar genellikle bir build() metodu içerir ve widget'ın nasıl görüneceğini tanımlar. Ancak, bu widget'lar durumlarını saklayamazlar.
- **Stateful Widget (Değişken Widget):** Stateful widget'lar, uygulamadaki dinamik veya değişen bileşenleri temsil eder. Özellikle kullanıcı etkileşimi sonucunda değişen bir sayacı veya giriş alanlarını yönetmek için stateful widget'lar kullanılır. Bu widget'lar bir state (durum) sınıfı içerir ve bu durum sınıfı, widget'ın değişen verilerini saklar. Stateful widget'lar, kullanıcı etkileşimleri veya uygulama durumu değişiklikleri ile yeniden çizilmeleri gereken yapılara sahiptir.
- **Scaffold Widget (Şema Widget):** Scaffold, temel bir yapı sağlar ve genellikle uygulamanın ana çerçevesini oluşturmak için kullanılır. App Bar, Drawer (çekmece), Bottom Navigation gibi öğeleri içerebilir. Scaffold, genellikle uygulamanın temel şemasını oluşturur.
- **AppBar Widget (Üstbilgi Widget):** AppBar, uygulamanın üst kısmındaki çubuk olarak görünen ve genellikle uygulama başlığı, gezinme menüsü veya diğer önemli kontrolleri içeren bir widget'tır. Uygulamanın genel düzenini oluşturur.
- **Container Widget (Kapsayıcı Widget):** Container widget'lar, diğer widget'ları içine alan ve düzenleyen bir yapı sağlar. Genellikle boyutlandırma, kenar boşlukları ve arka plan rengi gibi özelliklerle birlikte kullanılır. Özellikle, düzenleme gereksinimleri olan kullanıcı arayüzü bileşenlerini oluştururken kullanışlıdır.
- **Row ve Column Widget'lar:** Row ve Column widget'lar, sırasıyla yatay ve dikey düzenler oluşturmak için kullanılır. Birden fazla widget'ı sıralamak veya sıralamada yerleştirmek için kullanışlıdır. Özellikle birden fazla widget'ı yan yana veya alt alta düzenlemek istenildiğinde kullanılır.
- **ListView Widget (Liste Görünümü):** ListView, liste şeklinde sıralanan verileri görüntülemek için kullanılır. Genellikle, uzun liste öğelerini kaydırarak görüntülemek istediğinizde kullanışlıdır. ListView, çeşitli liste öğelerini içerebilir ve bu liste öğeleri otomatik olarak kaydırılabilir.

- **Card Widget (Kart Widget):** Card, bilgi veya içeriği düzenlemek için kullanılan bir widget'tır. Özellikle veri veya bilgi parçalarını gruplamak ve görsel olarak ayırt etmek için kullanılır.
- **Text Widget (Metin Widget):** Metin widget'ı, metin içeriğini görüntülemek için kullanılır. Özellikle başlık, açıklama veya etiketler için kullanılır.
- **Image Widget (Resim Widget):** Resim widget'ı, görüntü veya görsel içerik görüntülemek için kullanılır.
- **BottomNavigationBar Widget (Alt Navigasyon Çubuğu):** BottomNavigationBar, uygulamanın alt kısmında görünen gezinme çubuğunu temsil eder. Kullanıcıların farklı sayfalar veya sekmeler arasında geçiş yapmalarını sağlar.
- **TextField Widget (Metin Girişi Widget):** TextField, kullanıcının metin girmesini veya düzenlemesini sağlayan bir widget'tır. Özellikle form alanları, arama kutuları ve giriş alanları için kullanılır.
- **IconButton Widget (Simge Düğme Widget):** IconButton, ikonlar veya simgelerle temsil edilen basit düğmeleri oluşturmak için kullanılır. Özellikle, işlevsellik sağlamak için kullanılır.
- **Expanded Widget (Genişletilmiş Widget):** Expanded, bir sütunu veya satırı genişletmek için kullanılır. Özellikle, widget'ların mevcut alanı doldurmasını veya boşluğu doldurmasını sağlar.
- **Drawer Widget (Çekmece Widget):** Drawer, uygulamanın kenarından kaydırılarak açılan bir menüyü temsil eder. Genellikle, gezinme seçeneklerini veya diğer uygulama işlevlerini içerir.
- **AlertDialog Widget (Uyarı Penceresi Widget):** AlertDialog; kullanıcıya bilgi, onay veya uyarı mesajlarını iletmek için kullanılır. Özellikle, kullanıcı iletişimi için kullanılır.
- **TabBar Widget (Sekme Çubuğu Widget):** TabBar, birden çok sekme arasında gezinmeyi kolaylaştıran bir widget'tır. Özellikle, çoklu içerik gösteren sayfaları düzenlemek için kullanılır.
- **ListView.builder Widget:** Bu widget, büyük veri listelerini verimli bir şekilde oluşturmaya olanak sağlar. Veri kaynağından öğeleri dinamik olarak oluşturulmasını sağlar.

- **GridView Widget:** GridView, düzenli bir ızgaranın oluşturulmasını sağlar. Öğeleri satır ve sütunlarla düzenlemek için kullanılır.
- **Stack Widget:** Stack, diğer widget'ları üst üste veya yan yana konumlandırmak için kullanılır. Widget'ları istiflemek için kullanışlıdır.
- **CardView Widget:** CardView, malzeme tasarım yönergelerine uygun olarak yuvarlatılmış köşeleri olan kartlar oluşturulmasına olanak tanır.
- **RichText Widget:** RichText, farklı metin stilleri ve bağlantılar içeren metinler oluşturulmasını sağlar. Bu, metin biçimlendirmesi için kullanışlıdır.
- **AnimatedContainer Widget:** AnimatedContainer; konteynerlerin boyutunu, rengini veya diğer özelliklerini animasyonlarla değiştirmeye olanak tanır.
- **InkWell Widget:** InkWell, dokunma efektleri eklemek için kullanılır. Özellikle dokunma veya tıklama işlemleri için kullanışlıdır.
- **Spacer Widget:** Spacer, boşlukları düzenlemek ve widget'ları esnek bir şekilde sıralamak için kullanılır.
- **ClipRRect Widget:** ClipRRect, widget'ın kenarlarını yuvarlatmak veya kırpma işlemleri uygulamak için kullanılır.
- **Hero Widget:** Hero, sayfalar arasında geçişlerde görüntüleri veya widget'ları süzme efektleriyle eşleştirmeyi sağlar.
- **FutureBuilder Widget:** FutureBuilder, gelecekteki bir değeri içeren bir Future'dan sonuçları görüntülemek için kullanılır. Özellikle asenkron işlemleri gerçekleştirmek için kullanışlıdır.
- **PageView Widget:** PageView, sayfalar arasında kaydırma veya dönme işlemleri oluşturmaya olanak tanır.
- **TabBarView Widget:** TabBarView, sekme çubuğu ile ilişkilendirilmiş bir sayfa içeriğini görüntülemenizi sağlar.
- **ClipOval Widget:** ClipOval, içeriği oval bir şekilde kırpmaya olanak tanır.
- **Stepper Widget:** Stepper, çok adımlı işlemlere rehberlik etmek için kullanılır

(“Flutter documentation”, 2024).

3.4. Aydınlatma Sistemleri

Uluslararası Fotometri Komisyonu (International Commission on Photometry – CIP), ışık ve aydınlatma alanında standartlar oluşturmak için 1900 yılında kurulan bir komisyondur. Komisyonun ismi, 1913 senesinde yeniden düzenlenerek Uluslararası

Aydınlatma Komisyonu (International Commission on Lighting – CIE) olarak güncellenmiştir. Standartlaşma organizasyonu olan CIE, aydınlatmayı şu şekilde tanımlar: "Ortam, çevre ve küçük veya daha geniş alanlara ışık sağlayarak bunların fark edilmesini sağlamaktır."

3.4.1. Aydınlatma Sisteminin Kullanımı

Aydınlatma sistemlerini iç ve dış aydınlatma olmak üzere iki gruba ayırabiliriz:

İç aydınlatma: Apartman, ev, otel vb. gibi yerlerin içinde görsel rahatlık sağlanması için olan yapay aydınlatma türüdür.

Dış aydınlatma: Sokak, cadde, park vb. gibi yerlerde görsel rahatlık sağlanması için olan yapay aydınlatma türüdür.

Aydınlatma sistemleri günlük hayatımızın her alanında büyük öneme sahiptir. Aydınlatmanın yetersiz olduğu veya olmadığı yerlerde bazı sorunlara neden olmaktadır:

- Doğal aydınlatmanın yetersiz olduğu veya akşam saatlerinde yeterince aydınlatılmayan alanlarda, asayiş sorunları oluşur. Bundan dolayı, insanlar kendilerini güvende hissedemezler.
- Aydınlatmanın olmadığı hiçbir yerde insan görme yetisini kullanamaz veya yaşantısını sağlamakta oldukça güçlük çeker.
- Aydınlatmanın yetersiz olduğu sokak, cadde, park vb. yerlerde, insanlar arasında şiddetin yoğun olduğu gözlenmektedir.
- Aydınlatmanın olmaması, bireylerin yer, yön, istikamet bulmalarında zorluk çekmelerine neden olmaktadır.
- Tünellerde aydınlatmanın olmaması, yetersiz olması veya çok fazla olması, kaza riskinin artmasına ve maddi hasarlı veya ölümlü kazalara sebebiyet vermektedir.

3.4.1.1. Işık Kaynakları

Aydınlatma, ışık kaynağına göre 3’e ayrılmaktadır.

3.4.1.2. Doğal Aydınlatma

Ana gücünü güneşten alan güneş ışıklarının dünyaya gelmesi ile görsel rahatlık olanağı tedarik eden aydınlatma diye ifade edilmektedir.

3.4.1.3. Yapay Aydınlatma

Ana gücünü insanların yaptığı kaynaklardan alan görsel rahatlık olanağı tedarik eden aydınlatma diye ifade edilmektedir.

3.4.1.4. Bütünleşik Aydınlatma

Görsel rahatlığın oluşması için doğal ve yapay aydınlatmadan istifade ederek oluşan aydınlatma olarak açıklanabilir.

3.5. ASP .NET MVC

ASP.NET MVC, web uygulamaları geliştirmek için kullanılan güçlü bir çerçevedir. Bu model, görünüm ve denetleyiciyi (Controller) ayırarak uygulama geliştirmeyi kolaylaştırır. ASP.NET MVC, .NET platformunun bir parçasıdır ve web uygulamalarını hızlı ve verimli bir şekilde oluşturmaya yardımcı olur. Bu tezde, ASP.NET MVC'nin temel kavramlarına, avantajlarına ve nasıl kullanılacağına dair kapsamlı bir bakış sunulmuştur.

ASP.NET MVC, Microsoft tarafından geliştirilen ve web uygulamalarını oluşturmak için kullanılan bir modelleme çerçevesidir. MVC, Model-View-Controller'ın kısaltmasıdır ve web uygulamalarının tasarım ve geliştirme sürecini düzenlemeye yardımcı olur. Bu yöntem temel olarak; uygulamayı, Model, View ve Controller olarak adlandırılan üç ana bölüme ayırır.

3.5.1. Model

Model, uygulamanın veri yapısını temsil eder. Bu, veri tabanı işlemleri ve uygulama mantığının kalbidir. Model, uygulamanın verilerini çeker, işler ve saklar.
Uygulamanın veri yapısını temsil eden Modelleri oluşturur. Entity Framework veya benzeri bir ORM (Object-Relational Mapping) aracı kullanarak veri tabanı işlemlerini basitleştirir.

3.5.2. View

Görünüm, uygulamanın kullanıcı arayüzünü temsil eder. HTML, CSS ve JavaScript ile oluşturulan sayfaları içerir. Görünümler, kullanıcıya verileri sunar ve kullanıcı etkileşimini yönetir.

Görünümler ve kullanıcı arayüzleri, bu bölümde tasarlanır. Razor görünüm motoru, HTML, C# ve özel etiketler kullanarak görünümleri oluşturmanıza olanak tanır.

3.5.3. Controller

Denetleyici, Model ve View arasındaki bağlantıdır. İstekleri alır, Modeli günceller ve sonuçları View'e iletir. Denetleyici, kullanıcı isteklerini işler ve uygulamanın akışını kontrol eder.

Sınıflar ve bu sınıflar içindeki kullanıcı isteklerini işleyen Action metotları, bu bölümde tanımlanır. Action metotları, Modeli günceller ve sonuçları Görünüme (View) iletir.

3.6. HTML

HTML (Hypertext Markup Language), internetin temel yapı taşı olan bir işaretleme dilidir. HTML, 1991 yılında Tim Berners-Lee tarafından oluşturulmuş ve World Wide Web'in temelini atmıştır. İlk sürümü olan HTML 1.0, metin ve bağlantıları düzenlemek için sınırlı bir dizi etiket içermiştir. Daha sonra HTML 2.0, HTML 3.2, HTML 4.01 gibi sürümler gelmiştir. Ancak, HTML'nin en büyük dönüşümü, HTML 5.0 ile gerçekleşmiştir.

HTML 5.0, 2014 yılında W3C (World Wide Web Consortium) tarafından yayımlanmıştır. Bu sürüme; multimedya öğelerini doğrudan desteklenmesi, geliştirilmiş form işleme yetenekleri ve artırılmış tarayıcı uyumluluğu gibi yeni özellikler eklenmiştir. HTML 5.0, modern web uygulamaları geliştirmek için güçlü bir temel sunar (Berners-Lee ve Connelly, 1995).

Web sayfalarının oluşturulmasında kullanılır. Ayrıca, metin, görseller, bağlantılar ve diğer medya öğelerini bir araya getirerek web tarayıcıları tarafından görüntülenmesini sağlar.

HTML, web sayfalarını tasarlamak ve içeriğini tanımlamak için kullanılan bir işaretleme dilidir. Metin belgesi biçiminde yazılır ve web tarayıcıları tarafından yorumlanır. HTML, kullanıcıların metni biçimlendirmesine, bağlantılar oluşturmasına, görsel öğeler eklemesine ve interaktif öğeler oluşturmasına olanak tanır. Temel amacı, web sayfalarının yapısını tanımlamaktır.

HTML, "hypertext" veya "hipermetin" kavramının temelini oluşturur. Hipermetin, belirli bir metin parçasının, diğer metinlere veya kaynaklara bağlantılar içermesi anlamına gelir. Bu nedenle, HTML ile oluşturulan belgeler, diğer web sayfalarına veya kaynaklara yönlendiren bağlantılar içerebilir.

HTML, web tarayıcıları tarafından yorumlanır ve görüntülenir. Bir web sayfası açıldığında, HTML belgesi tarayıcı tarafından okunur ve belgede tanımlanan öğeler ekranda gösterilir. HTML dosyaları genellikle .html uzantısına sahiptir ve web sunucuları tarafından istemcilere (kullanıcılara) gönderilir.

HTML, metin içeriğini bir dizi etiket veya işaretleme ile çevreler. Bu işaretleme, metin öğelerini tanımlamak ve biçimlendirmek için kullanılır. HTML'nin temel bileşenleri EK-1 de verilmiştir (Berners-Lee ve Connelly, 1995).

3.7. CSS

CSS, "Cascading Style Sheets" teriminin kısaltmasıdır. Bu, bir web sayfasının görünümünü ve düzenini tanımlayan bir stil dilidir. CSS; sayfanın metin biçimi, renkleri, boyutları, kenar boşlukları ve daha fazlasını kontrol etmek için kullanılır. Ayrıca, sayfanın farklı ekran boyutlarına veya cihazlara uyumlu olmasını sağlayan "duyarlı tasarım" (responsive design) oluşturmak için de kullanılır.

HTML (Hypertext Markup Language) ile birlikte kullanılan CSS, web geliştiricilerinin sayfalarını daha çekici, düzenli ve kullanıcı dostu hale getirmelerine olanak tanır. CSS'in style özellikleri EK-2'de verilmiştir (Bartlett, 2006).

3.8. Web API

Web API, yazılım uygulamalarının birbirleriyle iletişim kurmasını sağlayan arayüzlerdir. Bu, farklı platformlarda çalışan uygulamaların veri ve işlevselliği paylaşmasını kolaylaştırır. Web API, sunucu tarafında hizmet veren uygulamalar tarafından sağlanır ve genellikle HTTP (Hypertext Transfer Protocol) ile iletişim kurarlar.

Web API, genellikle XML veya JSON gibi veri formatları kullanarak veri alışverişinde bulunur. Bu, istemcilerin (genellikle web tarayıcıları veya mobil uygulamalar gibi) sunucu tarafındaki uygulamadan veri almasını ve göndermesini sağlar. Web API, diğer uygulamaların verilerini okuma ve işleme yeteneği sunar. Web API'lerin kökenleri, 20. yüzyılın sonlarına kadar gitmektedir. Ancak, modern Web API'lerinin tarihi, internetin evrimi ile daha fazla öne çıkmıştır. Web API'lerin tarihçesinin önemli dönemleri:

1990'lar: World Wide Web Consortium (W3C), HTTP protokolünü standartlaştırdı ve web üzerinde veri alışverişi için temel olan HTTP GET ve POST metotlarını tanıttı.

2000'ler: Bu dönemde REST (Representational State Transfer) gibi API tasarımı ilkeleri gelişti. REST, web API'lerinin temel tasarım felsefesini sağladı ve HTTP metotları ile kaynakların temsilini birleştirdi.

2000'lerin ortaları: Web 2.0 dönemi, API'leri daha da yaygınlaştırdı. Sosyal medya platformları, API'lerini geliştiricilere açarak üçüncü taraf uygulamaların gelişimini teşvik etti.

2010'lar: Bulut bilişim hizmetleri, API'lerini sunmaya odaklandı. Amazon Web Services (AWS), Microsoft Azure ve Google Cloud Platform gibi büyük bulut sağlayıcıları, API'lerini geliştiricilere sunarak hizmetlerini genişletti.

SOAP (Simple Object Access Protocol) ve REST (Representational State Transfer) iki farklı web hizmeti iletişim protokolüdür. RESTful ise REST'in prensiplerine uygun olarak tasarlanmış web hizmetlerini ifade eder. Bunlar, web tabanlı uygulamalarda veri iletişimi ve işlem yapma amacıyla kullanılan yaklaşımlardır.

3.8.1. SOAP (Simple Object Access Protocol)

SOAP, web hizmetleri için eski bir iletişim protokolüdür. Temel olarak, XML tabanlı bir mesajlaşma protokolüdür ve ağ üzerindeki uygulamaların iletişim kurmasını sağlar. SOAP, ağ üzerindeki uygulamalar arasında yapılandırılmış ve standartlaştırılmış veri alışverişini sağlar. Özellikle büyük işletmeler ve kurumsal sistemler için güvenilirlik ve güvenlik gereksinimlerini karşılamak isteyen projelerde yaygın olarak kullanılır. Özetle:

- **Karmaşık**: SOAP mesajları, XML tabanlı olduğu için genellikle daha karmaşıktır ve daha büyük boyutlara sahiptir.
- **Standart**: SOAP, WS-Security ve WS-ReliableMessaging gibi geniş bir standart yelpazesi tarafından desteklenir. Bu, güvenlik ve güvenilirlik gereksinimlerini karşılamak için kullanışlıdır.
- **Bağımsız Protokol**: SOAP, HTTP dışındaki çeşitli iletişim protokollerini destekler (SMTP ve TCP gibi).
- **Stateful**: SOAP durum tutma yeteneğine sahiptir, bu da bir oturumu sürdürmek için kullanılabilir.
- **Tanımlı İşlemler**: SOAP, RPC (Remote Procedure Call) aracılığıyla işlemleri çağırmak için kullanılır. İşlemler ve veri yapıları WSDL (Web Services Description Language) ile tanımlanır.

3.8.2. REST (Representational State Transfer)

REST, web hizmetleri ve API'ler için daha yeni bir yaklaşımdır ve HTTP protokolünü temel alır. REST, kaynakları (resources) temsil eden URL'ler üzerinden basit HTTP istemcileri ile iletişim kurmayı hedefler. Ağ üzerindeki uygulamalar arasında kaynakların temsilini ve etkileşimini kolaylaştırır. Özetle:

- **Stateless (Durumsuz):** REST, her isteği ve yanıtı bağımsız ve durumsuz olarak işler. Bu, her isteğin kendi bağlamını taşıdığı anlamına gelir.
- **Veri Temsil İşlemi**: REST, kaynakların temsilini kullanır. Her kaynak, tek bir benzersiz URL ile temsil edilir. Bu kaynaklara, HTTP metotları (GET, POST, PUT, DELETE) kullanılarak erişilir.
- **Ölçeklenebilirlik**: REST, web uygulamalarını ölçeklendirmek ve paylaşmak için uygun bir şekilde tasarlanmıştır. Bu, web uygulamalarının büyüdükçe ve yeni kaynaklar ekledikçe iyi çalışmasını sağlar.
- **Hafiflik ve Basitlik**: REST, genellikle JSON veya XML gibi hafif veri formatlarını kullanır ve iletişim daha basit ve anlaşılır hale gelir.
- **Uygulama Alanları**: REST, web hizmetlerinde, uygulama programlama arayüzlerinde (API'ler), sosyal medya uygulamalarında ve web tabanlı uygulamalarda yaygın olarak kullanılır.

3.8.3. RESTFul

RESTful, REST ilkelerine uygun olarak tasarlanmış web hizmetlerini ifade eder.

- **Kaynaklar**: Her kaynak tek bir URL ile temsil edilir.
- **HTTP Metotları**: HTTP metotları (GET, POST, PUT, DELETE) kullanılarak kaynaklara erişilir ve işlem yapılır.

  GET: Web sunucudaki kaynağın ne olduğunu öğrenmeyi sağlar.

  POST: Sunucuda yeni kaynak oluşturulabilmesi için kullanılır.

  PUT: Sunucuda bulunan bir kaynağı güncellemek kullanılır.

  DELETE: Sunucuda bulunan bir kaynağı kaldırmak için kullanılır.
- **Durumsuzluk**: Her istek bağımsızdır ve tüm gerekli bilgileri içerir.
- **Veri Temsil İşlemi**: Kaynaklar, JSON veya XML gibi hafif veri formatları kullanılarak temsil edilir.
- **Bağlantısızlık**: RESTful hizmetleri, istemciler ve sunucular arasındaki bağlantısızlık prensibini takip eder.

- **Paylaşılan Arayüz**: RESTful hizmetleri, veri ve işlem yapma için standart HTTP metotlarını ve belirli bir kaynak yapısını paylaşır.

3.8.4. Web API Kullanım Alanları

Web API'ler, birçok farklı kullanım alanında önemli bir rol oynar. Web API'lerinin bazı kullanım alanları aşağıda verilmiştir:

- **Sosyal Medya Entegrasyonu**: Sosyal medya platformları; Web API'lerini kullanarak üçüncü taraf uygulamaların, bu platformlarla etkileşime girmesine olanak tanır. Bu, kullanıcıların sosyal medya hesaplarını yönetmelerini, içeriklerini paylaşmalarını ve daha fazlasını sağlar.
- **Ödeme İşlemleri**: Finans kurumları ve ödeme hizmet sağlayıcıları, Web API'lerini kullanarak çevrimiçi ödeme işlemleri sağlar. Bu, e-ticaret siteleri ve uygulamaları için kritik bir özelliktir.
- **Haritalama ve Konum Hizmetleri**: Harita hizmetleri (Google Haritalar vb.); Web API'lerini kullanarak, haritalama ve konum verilerini uygulamalara sağlar. Bu hizmet; konum tabanlı uygulamalar, seyahat ve navigasyon uygulamaları için önemlidir.
- **Veri Analizi ve Yönetimi**: Veri analitik platformları ve büyük veri hizmet sağlayıcıları, Web API'lerini kullanarak veri alışverişini ve analizini kolaylaştırır. Bu, işletmelerin verilerini daha iyi anlamalarına ve kullanmalarına yardımcı olur.
- **Mobil Uygulamalar**: Mobil uygulamalar, Web API'lerini kullanarak sunucu ile ilgili işlemler gerçekleştirir ve veri alışverişi yapar. Bu, mobil uygulamaların güncel ve etkileşimli olmalarını sağlar.
- **IoT (Nesnelerin İnterneti)**: IoT cihazları, Web API'ler aracılığıyla veri toplar ve yönetir. Bu; akıllı ev cihazları, endüstriyel sensörler ve diğer IoT çözümleri için önemlidir.
- **Eğitim ve E-Öğrenme**: Eğitim platformları; öğrenci yönetimi, içerik dağıtımı ve sınav değerlendirmesi için Web API'leri kullanır (“ASP.NET documentation”, 2023)

3.9. MySQL

MySQL, açık kaynaklı bir ilişkisel veri tabanı yönetim sistemidir (RDBMS). Verilerin saklanması, yönetilmesi ve sorgulanmasını kolaylaştıran bir yazılımdır.

MySQL; kullanım kolaylığı, hızlı performans, güvenilirlik ve geniş bir topluluk tarafından desteklenmesi gibi özellikleri ile popülerlik kazanmıştır.

MySQL, ilk olarak 1994 yılında İsveçli geliştirici Michael Widenius tarafından geliştirilmeye başlanmıştır. 1995 yılında İsveçli şirket MySQL AB tarafından ticari bir ürün olarak sunulmuş ve aynı yılın temmuz ayında ilk resmi sürümü olan MySQL 1.0 piyasaya sürülmüştür. MySQL, daha sonra 2008 yılında Sun Microsystems tarafından satın alınmış ve 2010 yılında Oracle Corporation'a devredilmiştir. Ancak, bu değişiklikler açık kaynak topluluğunu endişelendirmiş ve bu topluluk, MySQL'in açık kaynak sürümünün gelişmeye devam etmesi için MariaDB adında bağımsız bir sürüm oluşturmuştur (“MySQL”, 2023), (“MySQL Documentation”, 2024).

3.9.1. MySQL Temel Özellikleri

Açık Kaynaklı: MySQL, GNU Genel Kamu Lisansı (GPL) altında ücretsiz olarak dağıtılır. Herkesin MySQL'i ücretsiz olarak kullanabileceği, inceleyebileceği ve geliştirebileceği anlamına gelir.

İlişkisel Veri tabanı Yönetim Sistemi: MySQL, verilerin tablolar aracılığıyla ilişkilendirildiği bir ilişkisel veri tabanı yönetim sistemidir. Verilerin yapısallaştırılmasını ve veriler arasındaki ilişkilerin tanımlanmasını kolaylaştırır.

Performans ve Hız: MySQL, yüksek performans ve hızlı sorgulama özellikleri sunar. Veri tabanı işlemleri hızlı bir şekilde gerçekleştirilir.

Çoklu Platform Desteği: MySQL; Windows, Linux, MacOS ve diğer birçok işletim sistemi üzerinde çalışabilir. Bu, farklı platformlarda kullanıcıların tercihlerine göre MySQL'i kullanabilmesini sağlar.

Veri Bütünlüğü: MySQL, ACID (Atomik, Tutarlı, İzole, Dayanıklı) özelliklerini destekler. Veri bütünlüğünü ve güvenilirliğini sağlamak için önemlidir.

Güvenlik: MySQL, verilerin güvenliğini sağlamak için kullanıcı yetkilendirmesi, şifreleme ve diğer güvenlik önlemlerini destekler.

Yüksek Kullanılabilirlik ve Dayanıklılık: MySQL, yüksek kullanılabilirlik gereksinimlerini karşılayacak şekilde yapılandırılabilir. Verilerin yedeklenmesi ve güvende tutulması için bir dizi seçenek sunar.

Geniş Topluluk Desteği: MySQL, büyük ve aktif bir açık kaynak topluluğu tarafından desteklenir. Sorunların çözülmesi, geliştirmeler yapılması ve bilgiye erişilmesi için değerli bir kaynak oluşturur.

Geniş Veri Türü Desteği: MySQL; metin, sayılar, tarih/zaman, resimler ve diğer veri türlerini destekler.

3.9.2. MySQL Yönetim Araçları

MySQL veri tabanını yönetmek ve izlemek için birçok araç bulunmaktadır.

- **phpMyAdmin**: Web tabanlı bir yönetim aracıdır ve veri tabanlarının tarayıcı üzerinden kolayca yönetilmesini sağlar.
- **MySQL Workbench**: Görsel bir arayüz sunan MySQL resmi yönetim aracıdır. Veri tabanı tasarımı, sorgu oluşturma, izleme ve performans analizi gibi işlemlerinin yapılmasını sağlar.
- **Command-Line Aracı**: MySQL komut satırı aracı, gelişmiş kullanıcılar için bir seçenektir. Burada SQL sorguları doğrudan çalıştırabilir ve veri tabanı yönetilebilir.

3.9.3. MySQL Kullanım Alanları

MySQL birçok farklı kullanım alanına sahiptir. Bunlardan bazıları aşağıda verilmiştir:

- **Web Uygulamaları**: MySQL, web tabanlı uygulamaların veri depolama ihtiyaçlarını karşılar. Çoğu CMS (İçerik Yönetim Sistemi), blog platformu ve e-ticaret sitesi tarafından kullanır.
- **Veri Analitiği**: MySQL, büyük veri analitiği ve iş zekâsı uygulamalarında kullanılır.
- **Mobil Uygulamalar**: MySQL, mobil uygulamaların veri depolama gereksinimlerini karşılar.
- **Endüstriyel Otomasyon**: Sanayi ve otomasyon sistemleri, üretim ve veri yönetimi için MySQL'i kullanabilir.

3.10. PCD8574 I2C Modülü

PCD8574 I2C (Inter-Integrated Circuit), bir giriş/çıkış (GPIO) genişletme modülü olarak kullanılan bir arabirimdir. Modül, bir mikrodenetleyici veya bir bilgisayar tarafından kontrol edilen diğer cihazlarla iletişim kurmak için kullanılır.

I2C İletişimi

PCD8574 modülü, I2C iletişim protokolünü kullanarak mikrodenetleyicilere veya diğer cihazlara bağlanır. I2C, seri bir iletişim protokolüdür ve birden çok cihazın aynı veri yolu üzerinden iletişim kurmasını sağlar.

Genel Amaçlı Giriş/Çıkış (GPIO) Genişletme

PCD8574, mikrodenetleyicinin sınırlı sayıda GPIO pinini genişletmek için kullanılır. Modül, daha fazla dijital giriş ve çıkış pinine sahip olunmasını sağlar.

Paralel Giriş/Çıkış Kontrolü

Modül üzerindeki paralel giriş/çıkış pinleri, mikrodenetleyici veya başka bir cihaz tarafından kontrol edilebilir. Bu pinler, dijital sinyalleri okuma ve yazma yeteneği sağlar.

I2C Adres Seçimi

PCD8574 modülü, birden fazla cihazı aynı I2C veri yoluna bağlanılması için genellikle ayarlanabilir bir I2C adresine sahiptir. Bu, farklı PCD8574 modüllerini aynı veri yolu üzerinde kurulabilmesini sağlar.

Çıkış Akımı Sınırlamaları

Modülün çıkış pinlerinin sınırlı bir akım taşıma kapasitesi vardır. Bu nedenle, yüksek akım gerektiren uygulamalarda bir güç sürücü devresi veya röle kullanılması gerekebilir.

Çeşitli Kullanım Alanları

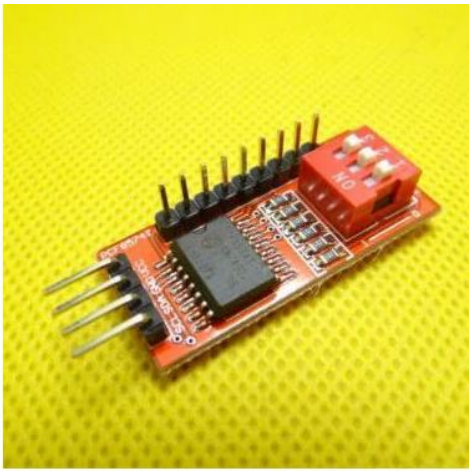
PCD8574 I2C modülü; mikrodenetleyici tabanlı projeler, otomasyon sistemleri, uzaktan kumanda uygulamaları ve diğer elektronik projeler için kullanışlıdır. Genellikle düşük hızlı veri iletimi gerektiren uygulamalarda tercih edilir.

Yazılım Desteği

PCD8574 I2C modülüyle uyumlu mikrodenetleyiciler için kütüphaneler ve sürücüler bulunmaktadır. Bu kütüphaneler ve sürücüler, modülün daha kolay ve hızlı bir şekilde kullanılmasını sağlar.

Devre Bağlantısı

PCD8574 modülü, mikrodenetleyici veya cihaz ile bağlantı kurabilmek için uygun bir devre bağlantısına ihtiyaç duyar. Bu bağlantılar genellikle SDA (Serial Data), SCL (Serial Clock), VCC (besleme gerilimi) ve GND (toprak) gibi pinleri içerir. Şekil 3.5'de, PCD8574 I2C modülü gösterilmektedir ("I2C", 2023),("Remote 8-bit I/O expander for I2C-bus with interrupt", 2013).



Şekil 3.5. PCD8574 I2C modülü

3.11. TCA9548A Mux Modülü

TCA9548A, Texas Instruments tarafından üretilen bir I2C çoklayıcı (multiplexer) entegresidir. "Mux" olarak da kısaltılan çoklayıcılar, birden çok I2C cihazını aynı I2C veri yoluna (bus) bağlamak için kullanılır. TCA9548A özellikle projelerde birden fazla I2C cihazını kontrol veya takip etmek istenildiğinde işe yarar. TCA9548A çoklayıcı (mux) modülü; sensörler, ekranlar, depolama cihazları ve daha birçok I2C (Inter-Integrated Circuit) iletişimine dayalı bileşen içerir. Özellikle sınırlı sayıda I2C adresi ve veri yolu (bus) bulunan projelerde, TCA9548A Mux (Çoklayıcı) modülü karmaşıklığı azalttığı için büyük bir öneme sahiptir. Bunun sebebi de birden çok I2C cihazına tek bir I2C veri yolundan erişilmesini sağlayan bir çözüm sunmasıdır.

3.11.1. TCA9548A Mux Modülünün Temel Özellikleri

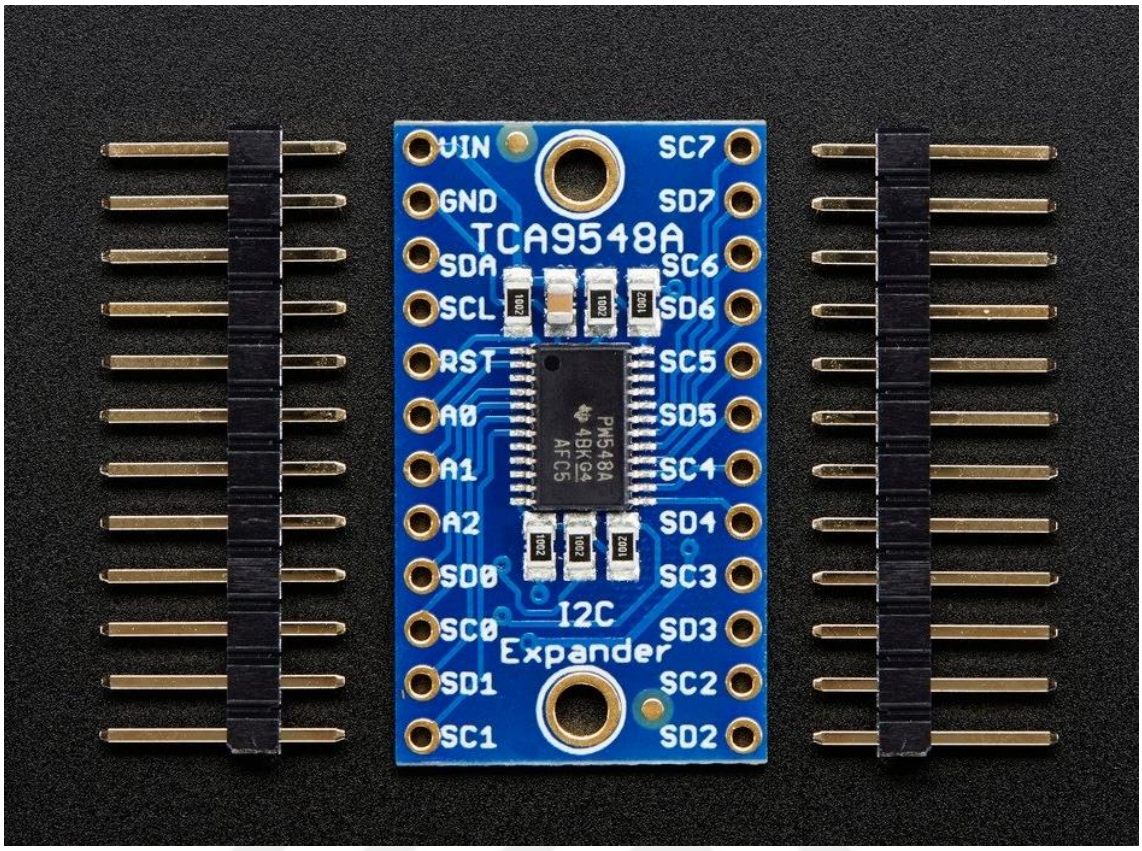
- **8 Kanal**: TCA9548A, 8 ayrı kanal sağlar. Her kanal, farklı bir I2C cihazına bağlanabilir.
- **I2C Kontrol**: Modül, I2C protokolünü kullanarak kontrol edilir ve yapılandırılır. Bu, bir mikrodenetleyici veya başka bir cihaz üzerinden TCA9548A'yı programlama kolaylığı sağlar.
- **Bağımsız Kanal Kontrolü**: Her kanal, bağımsız olarak etkinleştirilebilir veya devre dışı bırakılabilir. Sadece belirli cihazlara erişmek istenildiğinde enerji tasarrufu yapılmasını sağlar.

- **Daisy-Chaining**: Birden çok TCA9548A modülünü daisy-chain (zincirleme) yöntemiyle bağlayarak daha fazla kanal eklemek mümkündür.
- **Geniş Uygulama Alanları**: TCA9548A; sensör ağları, ekranlar, bellek cihazları ve diğer I2C cihazlarının yönetimini kolaylaştırır. Bu, IoT (Nesnelerin İnterneti) projelerinden robotik uygulamalara kadar geniş bir yelpazede kullanılmasını sağlar.
- **Yüksek Hızlı İletişim**: TCA9548A, yüksek hızlı I2C iletişimini destekler, bu da hızlı veri aktarımı sağlar.

3.11.2. TCA9548A Mux Modülü Kullanım Alanları

- **Sensör Ağları**: Birden çok sensörün aynı mikrodenetleyiciye bağlandığı projelerde, TCA9548A sensörler arasındaki iletişimi kolaylaştırır.
- **Ekran Kontrolü**: LCD ekranlar, OLED ekranlar ve diğer görüntü cihazları, TCA9548A ile tek bir mikrodenetleyici tarafından kontrol edilebilir.
- **Bellek Cihazları**: EEPROM'lar ve diğer bellek cihazları, TCA9548A ile aynı I2C veri yolunda birden çok projeye entegre edilebilir.
- **Endüstriyel Otomasyon**: Endüstriyel kontrol sistemleri, TCA9548A'yı farklı cihazlara erişmek için kullanabilir (“TCA9548A Low-Voltage 8-Channel I2C Switch with Reset”, 2019).

Şekil 3.6’da, TCA9548A Mux Modülü gösterilmektedir.



Şekil 3.6. TCA9548A Mux Modülü (“Adafruit TCA9548A 1-to-8 I2C Multiplexer Breakout”, 2015)

3.12. Ardunio Mega2560 Pro Mini Modülü

Arduino Mega 2560 Pro Mini, Arduino Mega 2560 R3'ün kompakt ve mini bir versiyonudur. Bu kart, ATMega2560 mikrodenetleyiciye dayanır ve çok sayıda dijital giriş/çıkış pini (GPIO), analog giriş pinleri, seri iletişim (UART), I2C ve SPI arabirimleri gibi birçok özellik sunar.

Arduino Mega 2560 Pro Mini, minyatür boyutu ve güçlü yetenekleri sayesinde büyük projeler için ideal bir seçenektir.

Mikrodenetleyici, büyük ve karmaşık projeleri işlemek için yeterli işlem kapasitesine sahip ve güçlü performans desteği sağlar (“Arduino Mega2560 Pro Mini Geliştirme Kartı”, 2023).

3.12.1. Ardunio Mega2560 pro mini Modülünün Temel Özellikleri

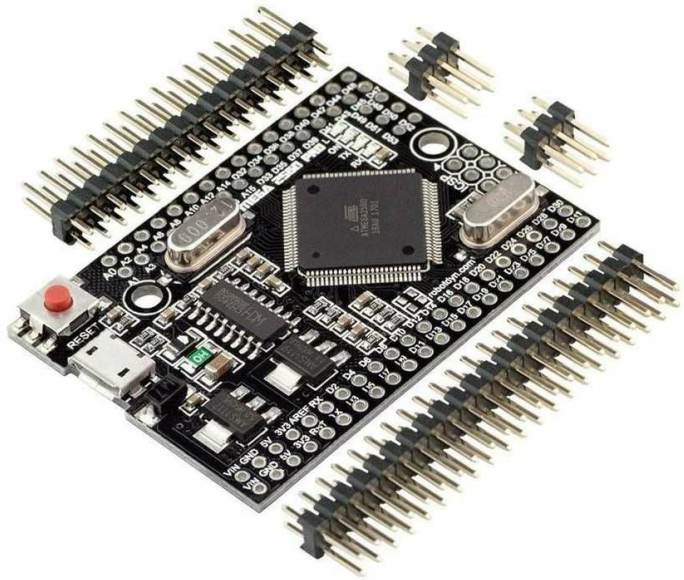
- **Mikrodenetleyici**: Arduino Mega 2560 Pro Mini, ATMega2560 mikrodenetleyiciye sahiptir. Bu mikrodenetleyici, 256 KB flash bellek, 8 KB SRAM ve 4 KB EEPROM ile güçlü bir işlem kapasitesine sahiptir.

- **Çok Sayıda Pin**: Kart, 54 dijital giriş/çıkış pini (14 tanesi PWM çıkışı olarak kullanılabilir) ve 16 analog giriş pini sunar.
- **Geniş Hafıza**: Büyük projeler için yeterli hafıza sunar ve programların saklanması ve çalıştırılması için geniş bir alan sağlar.
- **Seri İletişim**: Arduino Mega 2560 Pro Mini, dört adet UART (seri) iletişim bağlantısına sahiptir. Bu sayede, farklı cihazlarla seri iletişim kurulmasına olanak tanır.
- **USB Arabirimi**: Kartın üzerinde bir USB bağlantı noktası bulunur, bu sayede programlar kolayca yüklenebilir.
- **Harici Besleme**: Kart hem USB bağlantısı hem de harici bir besleme kaynağı aracılığıyla beslenebilir.
- **I2C ve SPI**: I2C ve SPI gibi iletişim arabirimlerini destekler, bu da sensörler, ekranlar ve diğer cihazlarla iletişim kurulmasını kolaylaştırır.
- **Geniş Kullanım Alanı**: Arduino Mega 2560 Pro Mini; endüstriyel otomasyon, robotik, 3D yazıcılar, büyük ölçekli IoT projeleri ve daha birçok uygulama için uygundur.

3.12.2. Ardunio Mega2560 Pro Mini Modülünün Kullanım Alanları

- **3D Yazıcılar**: Büyük ölçekli 3D yazıcıları ve CNC makineleri kontrol etmek için idealdir.
- **Endüstriyel Otomasyon**: Endüstriyel kontrol sistemleri geliştirmek için kullanılabilir.
- **Büyük IoT Projeleri**: Büyük ölçekli IoT projeleri için güçlü bir temel oluşturur.
- **Robotik**: Robot kontrolü ve otonom araçlar için uygundur.
- **Eğitim**: Eğitim amaçlı projeler için kullanılır ve öğrencilere mikrodenetleyici programlamayı öğretmek için yaygın bir seçenektir.

Şekil 3.7’de, Ardunio Mega2560 pro mini modülü gösterilmektedir.



Şekil 3.7. Ardunio Mega2560 pro mini modülü (“Arduino Mega2560 Pro Mini Geliştirme Kartı”, 2023)

3.13. Kamera

Kamera, görsel bilgiyi yakalayan ve kaydeden bir cihazdır. Temel olarak ışığı odaklayan bir lens, bu ışığı algılayan bir görüntü sensörü ve elde edilen görüntüyü kaydetmek için bir görüntü işleme ünitesi içerir. Kameraların farklı çeşitleri bulunmaktadır ve değişik amaçlar için kullanılabilir.

3.13.1. Kamera Türleri ve Kullanım Alanları

- **Dijital Fotoğraf Makineleri**: Bu tür kameralar, statik görüntülerin çekilmesi için kullanılır. Hem profesyonel fotoğrafçılar hem de amatör fotoğrafçılar tarafından yaygın olarak kullanılır. Dijital fotoğraf makineleri, yüksek çözünürlükte fotoğraflar çekebilir ve farklı ayarlarla fotoğraf çekme esnekliği sunar.
- **Dijital Video Kameraları**: Dijital video kameraları, hareketli görüntülerin kaydedilmesi için tasarlanmıştır. Televizyon, film yapımı, belgesel çekimleri ve kişisel video kayıtları için kullanılırlar.
- **Akıllı Telefonlar ve Tabletler**: Akıllı telefonlar ve tabletler, entegre kameralara sahiptir ve fotoğraf çekme, video kaydetme, görüntülü görüşme ve daha fazlası için kullanılır.
- **Güvenlik Kameraları**: Güvenlik kameraları; evlerde, iş yerlerinde ve kamusal alanlarda kullanılır. Gözetim ve güvenlik amaçları için tasarlanmışlardır.

- **Web Kameraları**: Web kameraları bilgisayarlarla bağlantılıdır. Video konferans, canlı yayın ve video görüşmeler için kullanılır.
- **Endoskoplar**: Tıbbi, sanayi ve inşaat alanlarında kullanılan endoskoplar, dar alanlarda ve borularda görüntü yakalamak için kullanılır.
- **Astronomi Kameraları**: Astronomi kameraları, gökyüzü gözlemleri için kullanılır ve derin uzay görüntülerini yakalamak amacıyla teleskoplarla birleştirilir.

Şekil 3.8'de, kamera gösterilmektedir.

Şekil 3.8. Harici kamera ("Piranha 9635 1080P Full HD PC Kamera, Siyah", 2023)

3.14. LDR Modülü

Işığa Duyarlı Direnç (Light-Dependent Resistor - LDR), birçok elektronik uygulamada aydınlatma kontrolünü gerçekleştirmek için kullanılan bir optoelektronik bileşendir. LDR, ışığa karşı hassasiyeti değişen bir direnç sunar ve çevresel ışık seviyesini algılamak için kullanılır.

Işığa Duyarlı Direnç (LDR), çevresel ışık seviyelerine göre direnci değişen yarıiletken bir cihazdır. LDR, aynı zamanda foto-direnç veya fotoiletken olarak da adlandırılır. Bu bileşen, ışık şiddeti arttıkça direncini azaltır ve azaldıkça direncini artırır. Bu özelliği, LDR'nin çevresel aydınlatma seviyelerini algılamak için kullanılmasını sağlar.

LDR'nin çalışma prensibi oldukça basittir. Genellikle LDR'nin yapısı, yarıiletken bir malzemeden oluşur ve bu malzeme ışığa maruz kaldığında elektriksel direnci değiştirir. Temel çalışma prensibi, içsel elektriksel bağların ışığa bağlı olarak değişmesidir.

LDR'nin üzerine ışık düştüğünde, içindeki elektriksel bağlar gevşer ve bu, direncin düşmesine neden olur. Diğer taraftan, karanlık bir ortamda, bağlar sıkışır ve bu da direncin yükselmesine yol açar. Sonuç olarak, LDR'nin ışık şiddetini algılamak için kullanılabileceği anlamına gelir.

3.14.1. LDR Modülünün Kullanım Alanları

- **Aydınlatma Kontrolü**: LDR, otomatik aydınlatma sistemlerinde kullanılır. Örneğin, bir odaya yeterli doğal ışık geldiğinde yapay aydınlatmayı kapatmak için kullanılabilir.
- **Güneş Enerjisi Uygulamaları**: LDR, güneş panellerinin gün boyunca güneşe dönmesini sağlamak için kullanılır.
- **Fotografik Uygulamalar**: Kameralar ve fotoğraf makineleri, LDR'leri pozlama kontrolü için kullanır. Parlaklık seviyelerine göre diyafram ve enstantane hızı ayarları yapılabilir.
- **Hava Durumu İstasyonları**: Hava durumu izleme sistemlerinde, günışığı seviyelerini ölçmek için LDR'ler kullanılır.
- **Aydınlatma Efektleri**: Sahne aydınlatması ve özel efekt uygulamalarında, ışık seviyelerini kontrol etmek için kullanılabilir.
- **Güvenlik Sistemleri**: Güvenlik kameraları ve alarm sistemleri, LDR'leri ışık seviyelerini algılamak için kullanır.

Işığa Duyarlı Direnç (LDR) modülü, çevresel ışık seviyelerini algılamak ve aydınlatma kontrolü gibi birçok uygulama için kullanılan kullanışlı bir sensördür. LDR, elektronik projelerde ve endüstriyel uygulamalarda kullanımı kolay, maliyet düşük ve güvenilirdir. Şekil 3.9’da, LDR modülü gösterilmektedir.

Şekil 3.9. LDR modülü (Semiz, 2018)

3.15. Led

LED (Light Emitting Diode), günümüzde yaygın olarak kullanılan bir aydınlatma teknolojisidir. Bu teknoloji, elektrik enerjisinin doğrudan ışığa dönüştürülmesini sağlar ve geleneksel aydınlatma yöntemlerine göre birçok avantaja sahiptir.

LED'ler, yarı iletken malzemeler kullanılarak yapılan bir elektronik bileşen türüdür. Genellikle ince bir silikon tabaka üzerine yerleştirilen LED, pozitif ve negatif bir uç arasına yerleştirilen bir yarı iletken çip içerir. Elektrik akımı bu çip üzerinden geçtiğinde, elektronlar pozitif yarı iletken tabakadan negatif yarı iletken tabakaya geçerler. Bu süreç sırasında elektronlar enerji seviyelerini değiştirir ve fazla enerjiyi ışık formunda yayarak serbest bırakırlar. Bu nedenle LED'ler "ışık yayan diyotlar" olarak adlandırılır.

LED'lerin çalışma prensibi temelde elektrolüminesansa dayanır. Yarı iletken malzeme içindeki elektronlar enerji seviyelerini değiştirirken, bu enerji seviyeleri bir fotonun (ışık parçacığının) salınmasına neden olur. Bu fotonlar, insan gözü tarafından algılanabilir ışık olarak görünür.

3.15.1. LED’lerin Avantajları

- **Enerji Verimliliği**: LED'ler, geleneksel halojen veya flüoresan lambalara göre çok daha az enerji tüketirler. Bu, enerji maliyetlerini düşürür ve çevresel etkiyi azaltır.
- **Uzun Ömür**: LED'ler, uzun bir kullanım ömrüne sahiptirler. Ortalama olarak 25.000 ila 50.000 saat kadar dayanabilirler.

- **Anında Açma/Kapama**: LED'ler anında yanıp söner, gecikme olmaz. Bu, hızlı aydınlatma gerektiren uygulamalarda çok faydalıdır.
- **Renk Seçeneği**: LED'ler, çeşitli renklerde ışık üretebilirler ve renk değiştirme özellikleri ile özellikle dekoratif aydınlatma için uygundur.
- **Düşük Isı Üretimi**: LED'lerin ısı üretimi oldukça düşüktür, bu da yanıcı malzemelerin yakınında güvenli bir şekilde kullanılmalarını sağlar.

3.15.2. LED'lerin Kullanım Alanları

- **Aydınlatma**: LED'ler, sokak lambalarından otomobil farlarına ve iç mekân aydınlatmasına kadar geniş bir uygulama yelpazesine sahiptir.
- **Göstergeler**: LED'ler; elektronik cihazlarda, trafik sinyallerinde ve bilgisayar monitörlerinde göstergeler olarak kullanılır.
- **Ekranlar**: Büyük ekran televizyonlar, bilgisayar monitörleri ve dijital reklam panoları gibi uygulamalarda LED ekranlar yaygın olarak kullanılır.
- **Dekoratif Aydınlatma**: LED'ler, dekoratif aydınlatma için popüler bir seçenektir. Renk değiştirme yetenekleri sayesinde çeşitli renk ve efektler elde edilebilir.

Şekil 3.10. LED ("3 mm ALICI SP213 LED Tipi", 2023)

LED teknolojisi sürekli olarak gelişmektedir. Daha yüksek enerji verimliliği, daha iyi renk üretimi ve daha düşük maliyetler gibi gelişmeler, LED'lerin daha fazla uygulama alanı bulmasına olanak tanır. Ayrıca, organik LED'ler (OLED'ler) gibi yeni LED türleri, daha esnek ekranlar ve aydınlatma sistemleri için yeni olanaklar sunmaktadır.

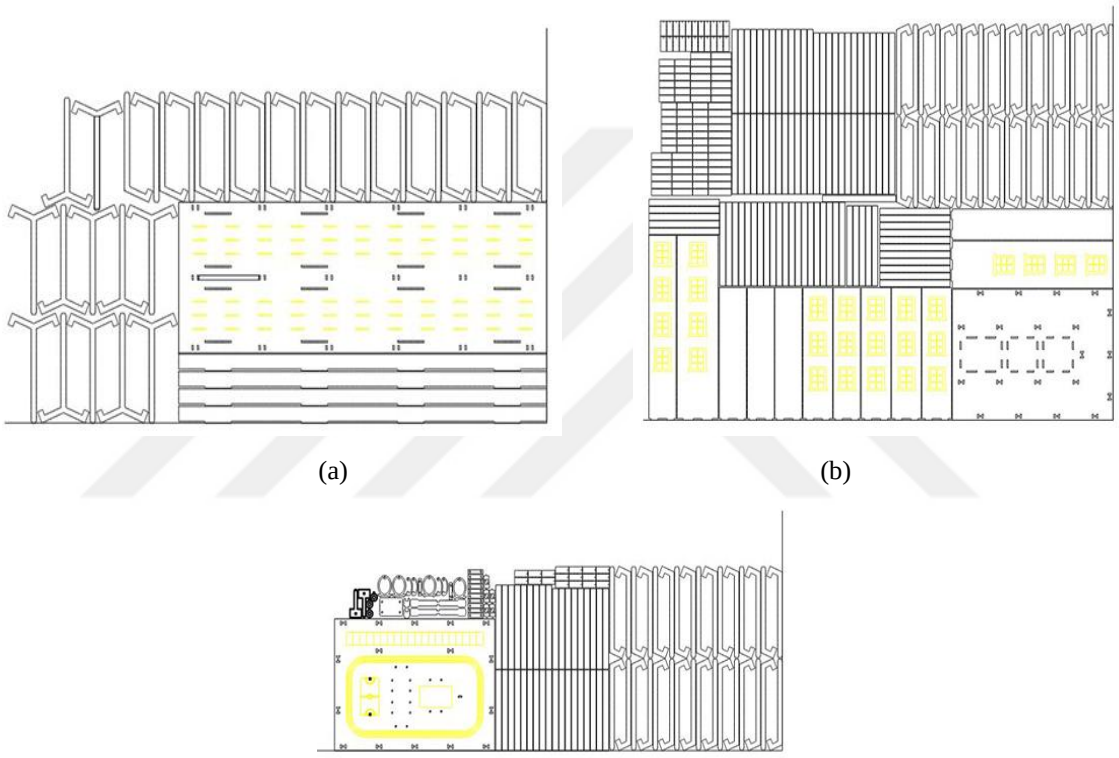
LED teknolojisi; enerji verimliliği, uzun ömür, renk seçenekleri ve daha birçok avantajı ile aydınlatma ve diğer uygulama alanlarında önemli bir rol oynamaktadır.

Gelecekteki gelişmelerle birlikte, LED'lerin daha da yaygınlaşması beklenilmektedir. Bu teknoloji, enerji tasarrufu ve çevre koruma çabalarına büyük katkıda bulunmaktadır ve aydınlatma dünyasını dönüştürmeye devam edecektir. Şekil 3.10’da örnek bir LED gösterilmektedir.

4. ÖNERİLEN AYDINLATMA SİSTEMİNİN TASARIMI

4.1. Prototip Hazırlanması

Projede kullanılacak prototipin hazırlanması için öncelikle iki boyutlu çizimler hazırlanmıştır. Çizimler şehirler arası yol, park ve sokak olmak üzere 3 farklı bölgeye ayrılmış ve her bölge için ayrı ayrı aydınlatma armatür çizimleri yapılmıştır. Şekil 4.1'de bu bölgeler için iki boyutlu çizim programı kullanılarak çizilen prototipler gösterilmektedir.

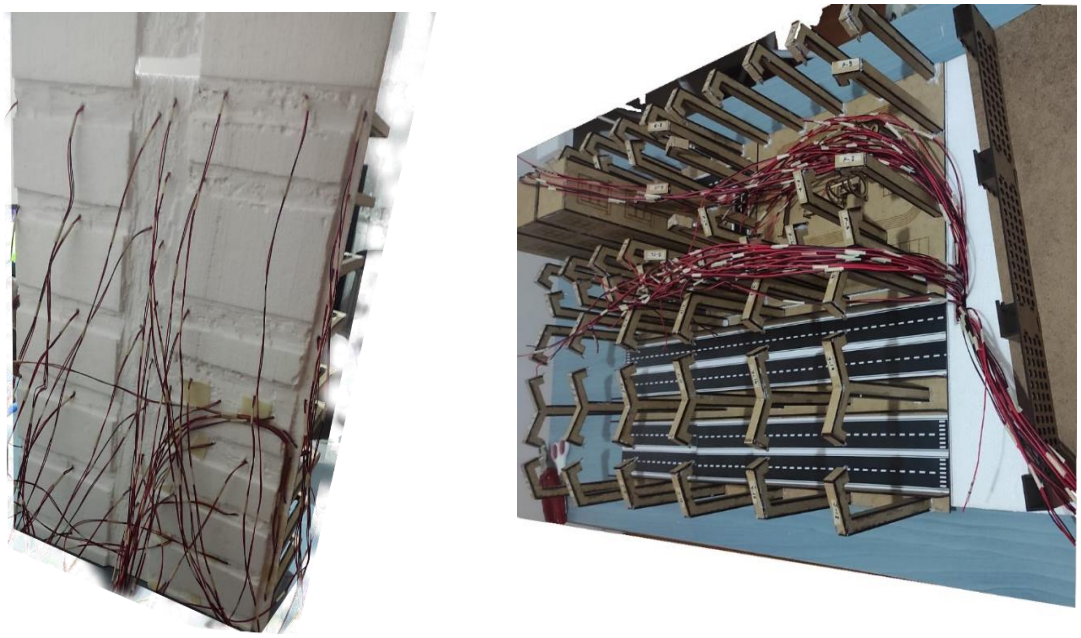


Şekil 4.1. Prototip çizimleri (a) otoyol ve armatür parçalarının çizimi (b) bir sokaktaki bina ve armatür parçalarının çizimi (c) bir park ve armatür parçalarının çizimi

Prototip' in bir örneğini oluşturmak için MDF levhaların CNC (Computer Numerical Control) makinesi ile örnek çizimlere göre kesimi yapılmıştır. Daha sonra bu parçalar her bölge için tek tek birleştirilerek Şekil 4.2'de ki gibi bir prototip elde edilmiştir.

Şekil 4.2. Prototip' in birleştirilmiş görüntüsü

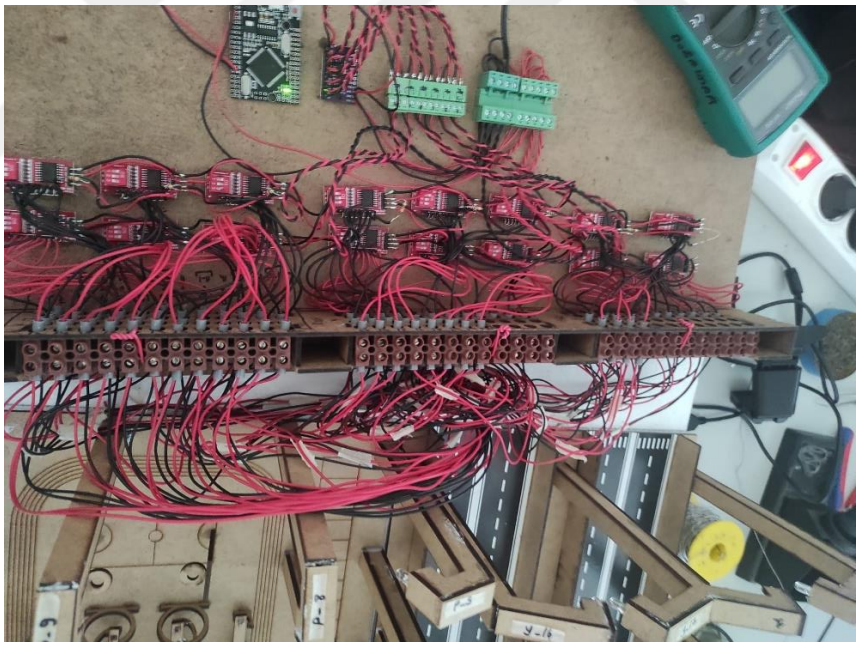
Prototip parçaları birleştirildikten sonra her bölgedeki aydınlatma direklerine LED ampuller eklenmiş ve bunların elektrik bağlantıları için kablolama işlemi gerçekleştirilmiştir. Bu aşamada, her bir LED ışığını doğru şekilde çalıştırmak için gerekli olan elektriksel bağlantılar yapılmıştır. Şekil 4.3'te prototipin elektrik bağlantılarının yapılması sonucu oluşan kablolama işlemi gösterilmektedir.



(a) (b)

Şekil 4.3. Prototip' in elektrik bağlantılarının yapılması (a) Bağlantıların arkadan görünümü (b) Bağlantıların önden görünümü

Prototip’ te aydınlatma armatürlerinin kablolama işleminden sonra Ardunio Mega2560 Pro Mini Modülü, PCD8574 I2C Modülü ve TCA9548A Mux Modülü kullanılarak devre bağlantıları oluşturulmuştur. Sistemde kullanılan Arduino Mega2560 Pro Mini, sensör verilerini okuma, kararlar alma ve çıkış cihazlarına kontrol gönderme gibi görevleri yöneten ana kontrol ünitesidir. PCD8574 I2C Modülü, giriş/çıkış genişletici bir entegredir ve I2C (Inter-Integrated Circuit) protokolünü destekler. Bu sayede mikrodenetleyici veya diğer I2C uyumlu cihazlarla iletişim kurabilir. TCA9548A Mux Modülü, birden fazla I2C cihazını tek bir I2C veriyolu üzerinden yönetmeye yarar. Arduino'nun tek bir I2C hattı kullanarak farklı PCF8574 modüllerine erişmesini sağlar. 8 adet giriş ve 8 adet çıkış olmak üzere 16 farklı I2C kanalına sahiptir. Sistemde 16 adet modül kullanıldığında, Arduino'ya ekstra 128 dijital giriş/çıkış pini sağlar. Bu sayede sistem, geniş bir dijital giriş/çıkış kapasitesine sahip bir sistem oluşturmanın yanı sıra, projelerde sensörleri ve çıkışları gruplamak ve yönetmek için kullanışlıdır. Şekil 4.4’te Modüllerin devre elemanları ile bağlantısı gösterilmiştir.



Şekil 4.4. Modüllerin devre elemanları bağlantı gösterimi

Prototipin devre tasarımı tamamlandıktan sonra, Arduino mikrodenetleyicisi kullanılarak aydınlatma armatürlerinin açılıp kapanması kontrol edilmiştir. Bu süreç, Arduino üzerinden gönderilen değerlerin devre tarafından doğru bir şekilde yorumlanması ve bu bilgilere dayanarak aydınlatma armatürlerinin istenilen durumda

çalıştırılmasının kontrol edilmesi ile gerçekleştirilmiştir. Şekil 4.5'de aydınlatma armatürlerinin açık hali gösterilmektedir.

Şekil 4.5. Prototip'in kablolamadan sonraki açık hali

4.2. Arduino Yazılımı

Arduino yazılımdaki setup fonksiyonu, programın başlangıcında yalnızca bir kez çalışan ve genellikle başlangıç ayarlarını ve konfigürasyonları gerçekleştirmek için kullanılan bir fonksiyondur. Arduino kodunda Otoyol, Sokak, Park alanlarının giriş çıkış pinleri ayrı ayrı ayarlanmıştır. Loop fonksiyonunu sürekli olarak çalıştığı için serial bağlantı üzerinde sürekli veri olup olmadığı kontrol edilir. Eğer veri varsa diğer işlemleri gerçekleştirir. Ardunio setup ve loop fonksiyonları EK-3'te verilmiştir.

Setup Fonksiyonu:

- Wire.begin(): I2C haberleşme protokolünü başlatır.

- PCF8574 genişletici entegrelerini farklı I2C adresleriyle başlatır ve başlatma durumunu kontrol eder.
- Seri haberleşmeyi başlatır (Serial.begin(9600)).
- Kanalların çıkış ve giriş pinlerini ayarlar.

Loop Fonksiyonu:

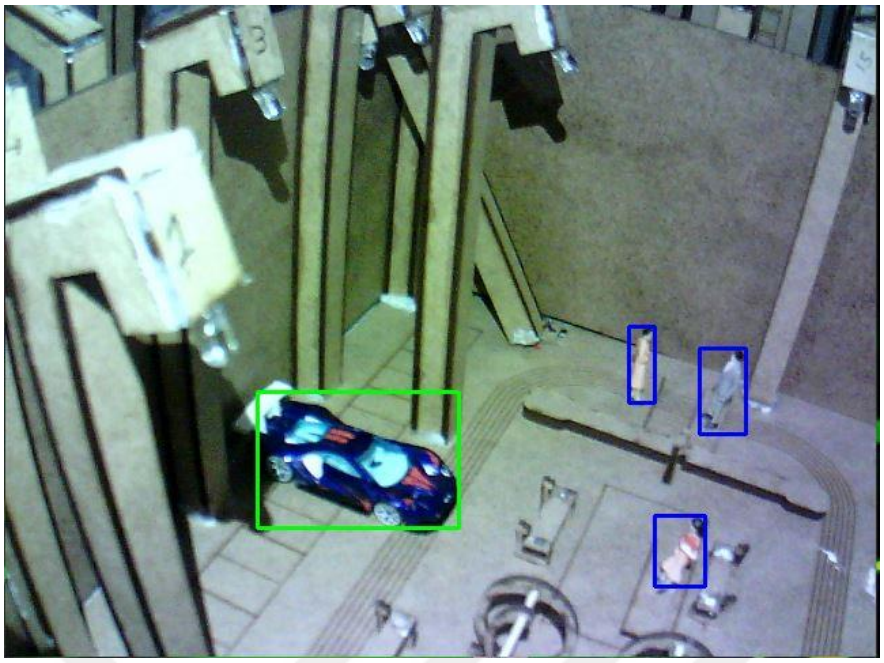
- Seri haberleşme ile gelen verileri okur.
- Gelen verileri işleyerek pin durumlarını günceller.
- Güncellenen pin durumlarını aydınlatma armatürlerine çıktı olarak gönderir.
- Belirli bir süre bekler (delay(1000)) ve ardından pin durumlarını okur ve seri iletişim aracılığıyla gönderir.

4.3. Python yazılımı

Python yazılımı için EK-4'te verilen kod ile görüntü işleme teknikleri kullanılarak bir video akışı analiz edilir. Analiz sonucunda, görüntü üzerinde tespit edilen insanlar belirlenir ve bu tespit sonuçlarına göre belirli bir zaman diliminde (saat 02:00 ile 05:00 arasında) bir web servisi kullanılarak armatürlerin açma ve kapatma işlemleri gerçekleştirilir. Kodun diğer bir kısmında ise donanım kısmında kullanılan LDR (Light-Dependent Resistor) modülünden gelen ışık şiddeti değeri izlenerek sistemde bulunan armatürlerin açılması veya kapanması sağlanır.

İlk olarak, görüntü işleme bölümünde, bir kamera tarafından video akışı alınır ve bu akış işlenerek içerisindeki insanlar tespit edilir. Bu tespit işlemi, önceden eğitilmiş bir nesne algılama modeli kullanılarak gerçekleştirilir. İnsanlar tespit edildikçe, bu bilgi kullanılarak belirli bir web servis aracılığıyla armatürlerin durumu kontrol edilir. Eğer belirli bir zaman diliminde insanlar algılanıyorsa, armatürler belirli bir duruma getirilir; aksi takdirde armatürler kapatılır. Şekil 4.6' da görüntü işleme teknikleri kullanılarak alınan görüntüden nesnelerin tespit edilmesi gösterilmektedir.

İkinci olarak, donanım bölümünde, LDR modülü ile gelen ışık şiddetinin değeri ölçülür. Bu değer, çevresel ışık koşullarını yansıtır. Eğer ışık şiddeti belirli bir eşik değerinin altında ise (örneğin, gece veya karanlık ortam), sistemde bulunan armatürler açılır. Işık şiddeti eşik değerinin üzerindeyse (örneğin, gündüz veya aydınlık ortam), armatürler kapatılır.



Şekil 4.6. Görüntü işleme teknikleri ile nesne takibi

Bu iki bölümün entegrasyonu, kullanıcıya enerji tasarrufu ve güvenlik sağlama esnekliği sunmaktadır. Görüntü işleme ile insanların algılanması, gece saatlerinde otomatik olarak sokak aydınlatmasını artırmaktadır. Donanım tarafındaki LDR değeri ise, gün boyunca çevresel ışık koşullarına adapte olacak şekilde armatürlerin kontrol edilmesini sağlamaktadır. Bu şekilde, enerji verimliliği sağlanarak şehir altyapısının daha akıllı bir şekilde yönetilmesine imkân sunulmaktadır.

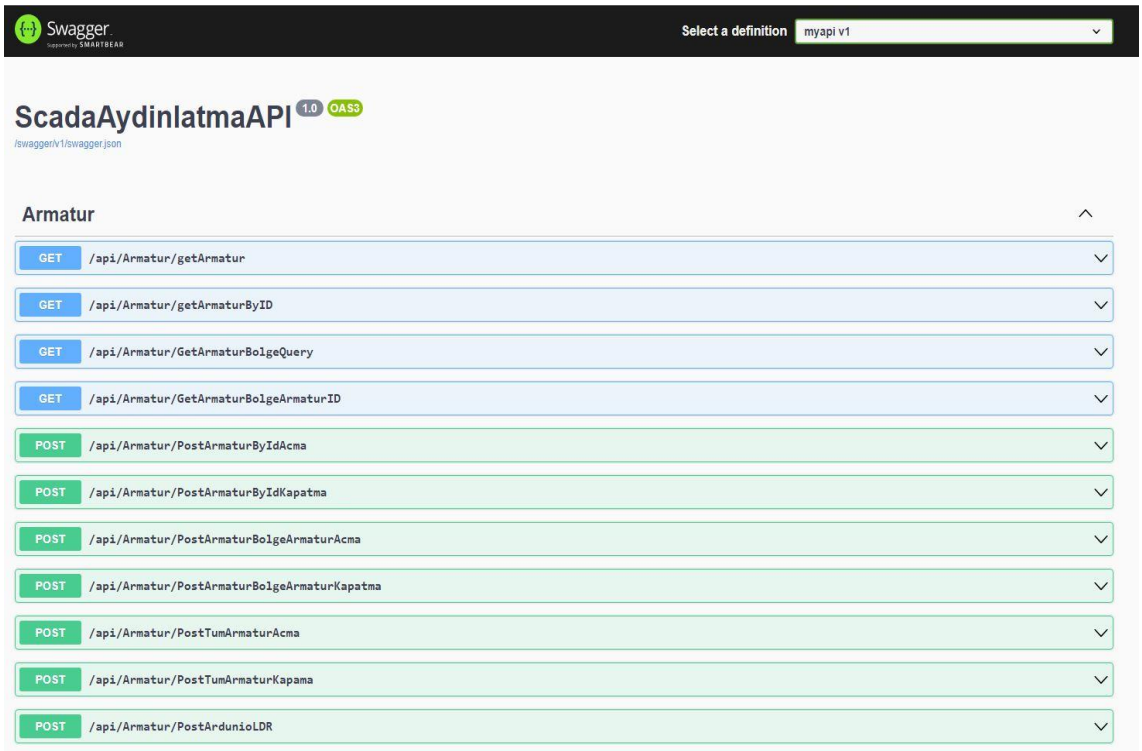
4.4. Asp.Net Core Web API

Projede kullanılacak uygulamaların farklı platformlar arasında, etkili bir veri iletişimini sağlamak amacıyla geliştirilmiş olan arayüz, çeşitli HTTP protokolü temelli işlemleri destekler. Bu işlemler arasında özellikle POST, GET, DELETE ve PUT gibi CRUD (Create, Read, Update, Delete) operasyonları bulunmaktadır. Bu arayüz sayesinde, projenin çeşitli bileşenleri arasında veri alışverişi ve etkileşim kolayca gerçekleştirilebilmektedir.

Armatürlerin API ekranı, uygulamalara sunulan arayüzün kullanıcı dostu bir görselleştirmesini sağlamaktadır. Şekil 4.7'de gösterilen Armatürlerin API ekranı, sistemde bulunan armatürlerle ilgili çeşitli işlemlerin yapılabileceği bir kontrol panelini ifade etmektedir. CRUD işlemleri, yani yeni armatürlerin eklenmesi, mevcut armatür bilgilerinin okunması, güncellenmesi ve silinmesi gibi temel işlemler bu ekran üzerinden gerçekleştirilebilmektedir. Özellikle, POST işlemi aracılığıyla yeni armatür bilgilerinin

eklenmesi, sistemde yeni armatürlerin entegrasyonunu kolaylaştırmaktadır. GET işlemi ile mevcut armatür bilgileri elde edilebilir, DELETE işlemi ile bir armatürün sistemden kaldırılabilir ve PUT işlemi ile armatür bilgileri güncellenebilir. Bu API, farklı uygulama ve cihazlar arasında veri uyumluluğunu sağlayarak sistemdeki armatürlerin etkili bir şekilde yönetilmesini mümkün kılmaktadır.

Projede kullanılan bu arayüz, geliştiricilere ve sistem yöneticilerine, armatürlerin hızlı ve güvenilir bir şekilde yönetilmesini sağlayan bir kontrol noktası sunmaktadır. Ayrıca, farklı platformlardan bu arayüze erişim sağlama esnekliği, projenin geniş bir kullanıcı kitlesi tarafından kolayca benimsenmesine olanak tanır.



Şekil 4.7. Armatürler API ekranı

Projede sunulan Kullanıcılar API ekranı, kullanıcılara çeşitli Kullanıcılar ile ilgili işlemleri gerçekleştirebilmeleri için bir arayüz sunmaktadır. Temel olarak CRUD (Create, Read, Update, Delete) işlemleri bu ekran üzerinden kolayca yapılabilmektedir. Bu API, projedeki Kullanıcı yönetimini daha etkili ve kullanıcı dostu haline getirmeyi amaçlamaktadır. Şekil 4.8'de gösterilen Kullanıcıların API ekranı, bu işlemlerin yapıldığı bir kontrol panelini temsil etmektedir.

CREATE (Oluştur): Kullanıcılar, bu ekrandan yeni kullanıcılar ekleyebilirler. Bu işlem ile sisteme yeni bir kullanıcı tanımlama süreci başlatılır. Yeni kullanıcı bilgileri,

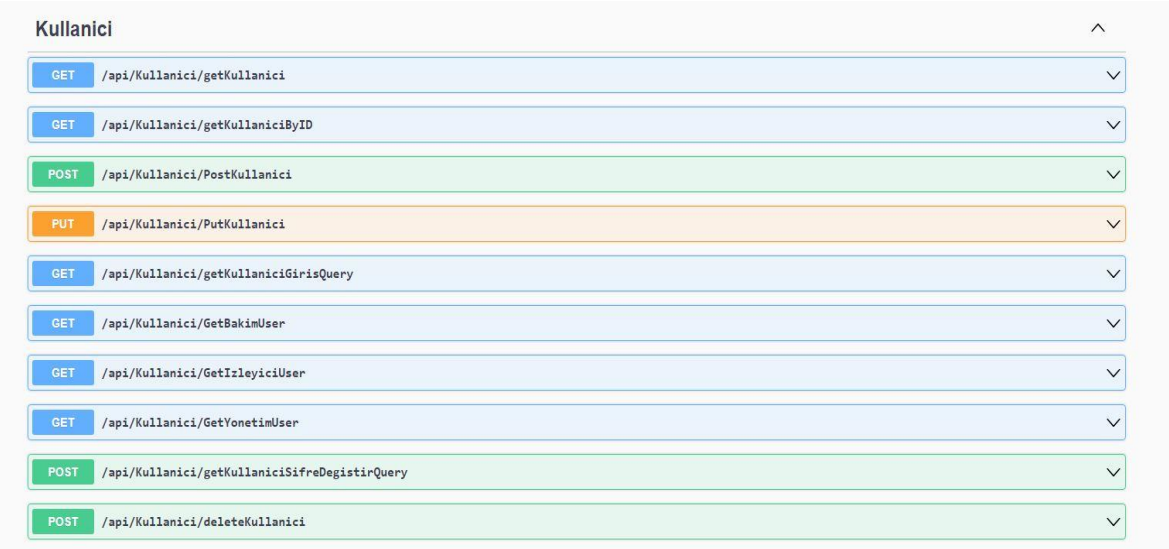
gerekli alanlar doldurularak ve isteğe bağlı olarak roller ve yetkiler atanarak sisteme dahil edilebilir. Bu, projede yeni katılımcıların hızlı ve güvenli bir şekilde eklenmesini sağlar.

READ (Oku): Mevcut kullanıcı bilgileri, bu API arayüzü üzerinden okunabilir. Kullanıcılar, sistemde kayıtlı olan kullanıcıların detaylarına erişebilirler. Bu özellik, bir kullanıcının rolü, yetkileri ve diğer ilgili bilgileri gibi detayları görmelerine olanak tanır. Böylece, sistemin genel kullanıcı yönetimine daha iyi bir bakış sağlanır.

UPDATE (Güncelle): Kullanıcılar, mevcut kullanıcı bilgilerini bu ekran aracılığıyla güncelleyebilirler. Kullanıcı bilgilerinde yapılan değişiklikler, sisteme hemen uygulanır. Bu, bir kullanıcının şifresini güncelleme veya iletişim bilgilerini değiştirme gibi durumları içerir. Güncel bilgilerin tutulması, sistemde doğru ve güncel kullanıcı bilgilerinin bulundurulmasını sağlar.

DELETE (Sil): Sistemden bir kullanıcının silinmesi işlemi de bu API aracılığıyla gerçekleştirilebilir. Bu, bir kullanıcının sistemden tamamen kaldırılmasını ifade eder. Silinen bir kullanıcının daha sonra sisteme geri eklenmesi, CREATE işlemiyle yeniden yapılabilir.

Bu API ekranı, kullanıcıların projedeki Kullanıcı yönetimini daha etkili bir şekilde gerçekleştirmelerine yardımcı olmaktadır. Kullanıcıların işlemleri bu ekran üzerinden güvenli ve kullanıcı dostu bir arayüzle gerçekleştirilebilir. Bu da projenin genel kullanıcı deneyimini iyileştirmekte ve yönetimini kolaylaştırmaktadır.



Şekil 4.8. Kullanıcılar API ekranı

Projede sunulan Bakım API ekranı, sistemin bakım süreçlerini etkili bir şekilde yönetmek için tasarlanmış bir kontrol panelidir. Bu ekran aracılığıyla kullanıcılar, çeşitli

bakım işlemlerini gerçekleştirebilirler. Temelde CRUD (Create, Read, Update, Delete) operasyonlarını destekler ve kullanıcılara sistemin bakımını yönetme esnekliği sunar. Şekil 4.9'da gösterilen Bakım API ekranı, bu işlemlerin yapıldığı bir arayüzü temsil etmektedir.

Şekil 4.9. Bakım API ekranı

CREATE (Oluştur): Bu işlemle kullanıcılar, sisteme yeni bakım görevleri ekleyebilirler. Yeni bir bakım görevi eklemek, sistemin düzenli bakımını ve güncellemelerini planlamak için kullanılır. Kullanıcılar, bakım görevlerini tanımlayarak, hangi sistem veya bileşenin ne zaman bakıma alınacağını belirleyebilirler. Bu, sistemdeki donanım ve yazılım bileşenlerinin sağlığını ve güvenilirliğini artırmak için kritik bir adımdır.

READ (Oku): Mevcut bakım görevleri, bu API ekranı üzerinden okunabilir. Kullanıcılar, sistemde planlanan ve geçmiş bakım görevlerinin detaylarına erişebilirler. Bu, hangi bileşenin ne zaman bakıma alındığı, bakım sürecinde hangi işlemlerin gerçekleştirildiği gibi bilgileri içerir. Bu özellik, sistem yöneticilerine sistemin durumu hakkında genel bir bakış sağlar.

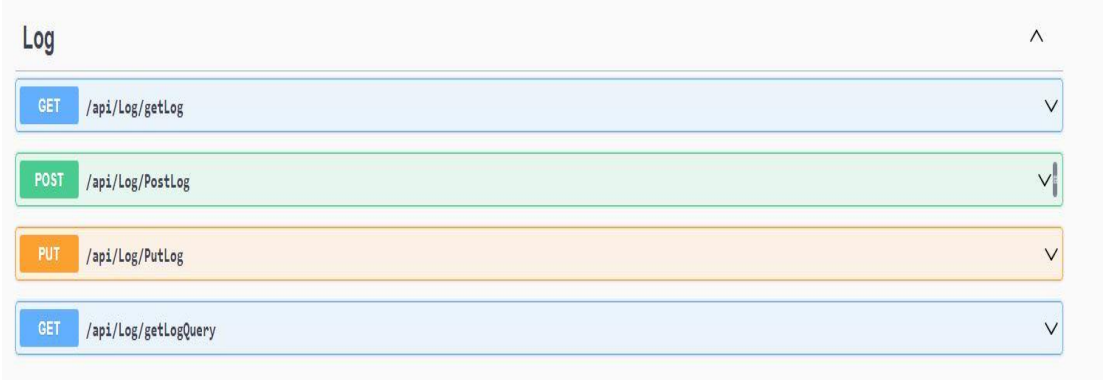
UPDATE (Güncelle): Bu işlem, mevcut bakım görevlerinin bilgilerini güncelleme imkânı sunar. Kullanıcılar, önceki bir bakım görevini düzenleyerek, güncel bilgileri ekleyebilir veya değiştirebilirler. Bu özellik, özellikle bir bakım görevinin planlamasında veya içeriğinde değişiklik yapılmak istendiğinde kullanılır.

DELETE (Sil): Sistemdeki bir bakım görevini silmek, bu görevin artık gereksiz olduğunu veya yanlışlıkla eklenmiş olduğunu ifade eder. Silme işlemi, sistemin bakım

yönetimi verimliliğini artırmak ve gereksiz bilgilerden arınmak için önemlidir. Bu adım, gereksiz bilgilerin birikmesini önleyerek sistem performansını artırabilir.

Bakım API ekranı, sistem bakımını daha düzenli ve etkili bir şekilde yönetmek isteyen kullanıcılara önemli bir araç sunmaktadır. Bu arayüz, bakım süreçlerini planlama, takip etme ve güncelleme açısından geniş bir esneklik sağlamakta, böylece sistem sürekli olarak yüksek performans ve güvenilirlikle çalıştırılır.

Projede sunulan Log API ekranı, sistemdeki olayları (logları) kayıt altına almak, izlemek ve yönetmek amacıyla tasarlanmış bir kontrol panelidir. Kullanıcılar, bu ekran aracılığıyla çeşitli CRUD (Create, Read, Update, Delete) işlemlerini gerçekleştirerek log verilerini yönetebilirler. Şekil 4.10'da gösterilen Log API ekranı, bu işlemlerin yapıldığı bir arayüzü temsil etmektedir.



Şekil 4.10. Log API ekranı

CREATE (Oluştur): Kullanıcılar, bu işlemle yeni log kayıtları ekleyebilirler. Yeni bir olay veya durum oluştuğunda, bu API ekranı üzerinden bu olayın detaylarını sisteme eklemek mümkündür. Bu, sistemin genel performansını izlemek, hataları belirlemek ve çeşitli olayları takip etmek için önemlidir.

READ (Oku): Mevcut log kayıtları, bu API ekranı üzerinden okunabilir. Kullanıcılar, sistemdeki olayların geçmişini inceleyebilir, belirli bir zaman aralığındaki logları görüntüleyebilir ve olaylar arasındaki ilişkileri anlamak için detaylı bilgilere erişebilirler. Bu, sistem yöneticilerine sorun giderme ve analiz için önemli bir kaynak sağlar.

UPDATE (Güncelle): Bu işlem, mevcut log kayıtlarının güncellenmesine olanak tanır. Örneğin, bir hata düzeltildiğinde veya bir durum çözüldüğünde, ilgili log kaydının güncellenmesi bu işlemle gerçekleştirilebilir. Bu, sistemdeki olaylarla ilgili bilgilerin güncel ve doğru olmasını sağlar.

DELETE (Sil): Sistemdeki belirli bir log kaydını silmek, bu kaydın artık gereksiz veya geçerli olmadığını ifade eder. Bu işlem, sistem yöneticilerine gereksiz bilgilerden arınma ve depolama alanını optimize etme imkânı tanır. Ancak, dikkatlice kullanılmalıdır, çünkü silinen log kayıtları geri getirilemez.

Log API ekranı, sistemin olaylarını ve durumlarını izlemek, takip etmek ve analiz etmek isteyen kullanıcılara önemli bir araç sunmaktadır. Bu arayüz, kullanıcıların sistem performansını değerlendirmelerine, hataları tespit etmelerine ve genel olarak sistem sağlığını daha etkili bir şekilde yönetmelerine yardımcı olmaktadır.

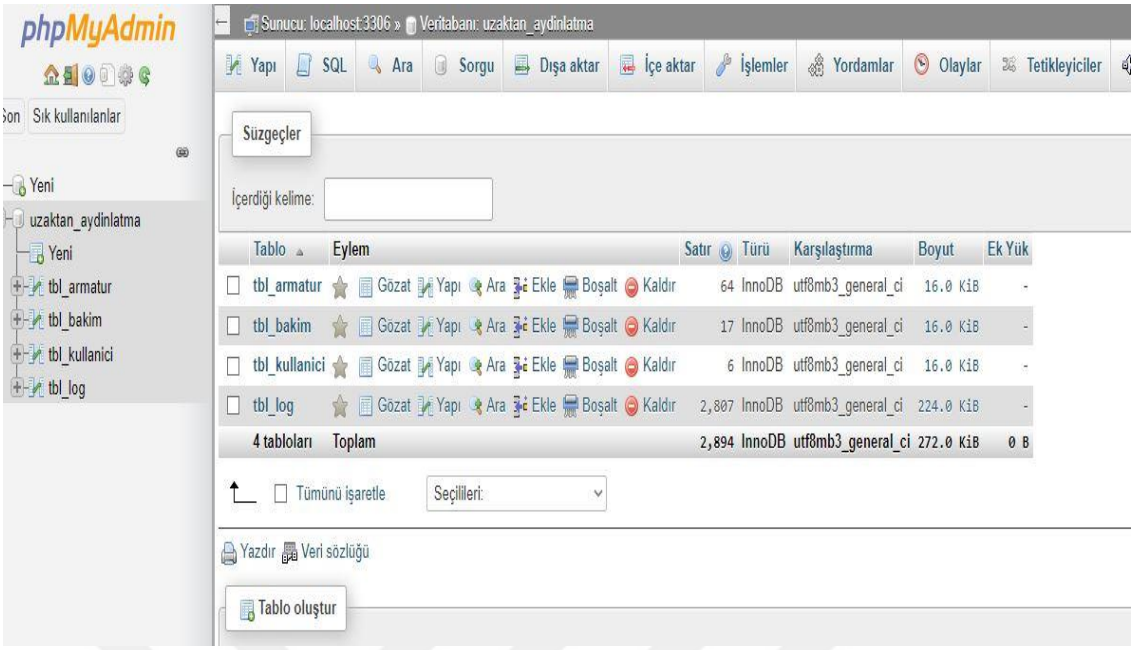
4.5. MySQL

Projede kullanılacak olan veri tabanı, uygulamalardan ve sensörlerden elde edilen çeşitli verilerin düzenli bir şekilde depolanmasını sağlayan kritik bir bileşendir. Bu veri tabanı, MySQL kullanılarak phpMyAdmin aracı ile oluşturulan özel tabloları içerir. Bu tablolar, projenin ihtiyaçlarına uygun olarak tasarlanmış ve farklı veri türlerini organize etmek amacıyla özel alanlara sahiptir.

Uygulama içerisinde, phpMyAdmin aracı kullanılarak hazırlanan veri tabanı tablolarının tamamını gösteren bir ekran bulunmaktadır. Bu ekran, projedeki veri tabanı yapısını bütünsel bir şekilde sunar ve kullanıcılara farklı veri tabloları arasındaki ilişkileri anlama ve veri tiplerini keşfetme imkânı tanır. Şekil 4.11’de bu tüm tabloların bulunan ekran gösterilerek kullanıcılara görsel bir rehberlik sağlar.

Tüm tabloların gösterildiği bu ekran, projenin veri yönetimi süreçlerine katkıda bulunur. Kullanıcılar, bu ekran üzerinden veri tabanı tablolarındaki her bir alanın içeriğini inceleyebilir ve veriler arasındaki ilişkileri anlamlandırabilir. Bu, projenin genel veri yönetim stratejisinin anlaşılmasına ve optimize edilmesine katkı sağlar. Bu aynı zamanda, kullanıcıların verilere daha etkili bir şekilde erişmelerine ve projedeki veri analizi süreçlerini iyileştirmelerine olanak tanır.

Uygulamada kullanılan aydınlatma armatürlerine ait değerli bilgilerin depolandığı özel bir "armatür" tablosu bulunmaktadır. Bu tablo, projenin aydınlatma yönetim sisteminin temelini oluşturan ve armatürlerin özelliklerini içeren kritik bilgileri barındırır. Her bir armatür, bu tablo içerisinde belirli alanlara sahip olarak detaylı bir şekilde kaydedilmiştir.

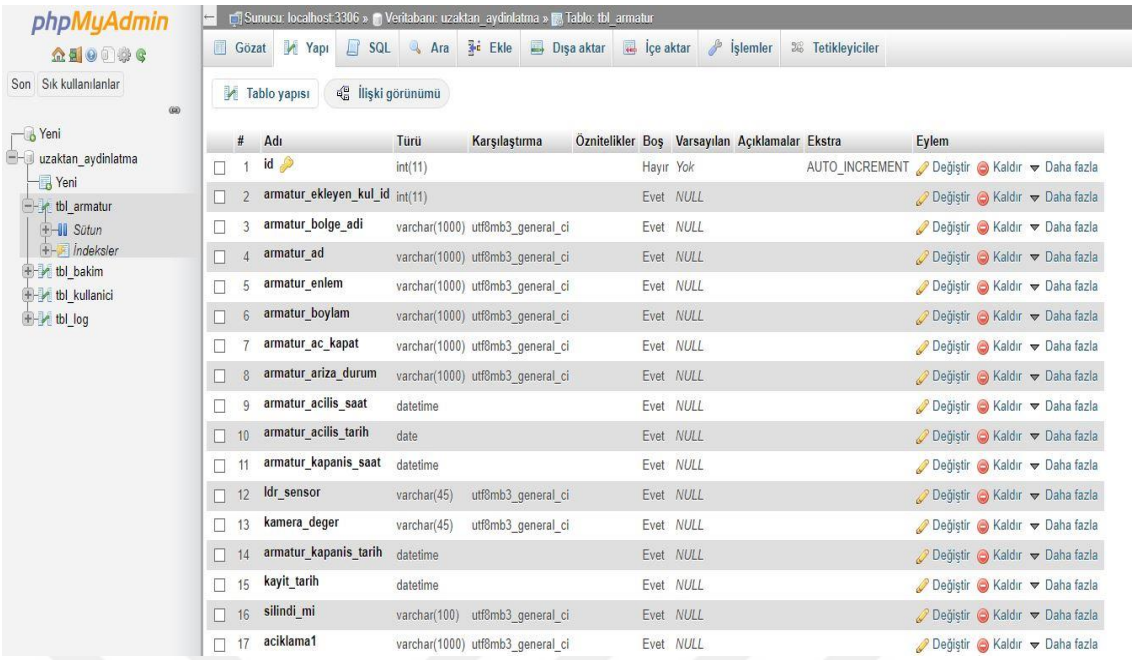


Şekil 4.11. Tüm tablolar ekranı

Uygulama içerisinde, bu özel "armatür" tablosunun ve ilgili alanlarının tamamını gösteren bir ekran bulunmaktadır. Şekil 4.12'de bu ekranı detaylı bir şekilde sunarak kullanıcılara her bir armatürün özellikleri ve ilişkili bilgiler hakkında genel bir bakış sunar. Bu sayede, kullanıcılar armatürlerin konumları, bölge adı, bağlantı durumu gibi önemli detayları kolayca inceleyebilir ve gerektiğinde bu bilgileri güncelleyebilir.

Bu özel ekran, aydınlatma sistemine dair bilgilerin şeffaf bir şekilde görüntülenmesine olanak tanır ve armatürlerin özelliklerinin kolayca yönetilebilmesini sağlar. Bu, projenin aydınlatma yönetimini optimize etmek ve kullanıcıların armatürlerle ilgili verilere daha etkili bir şekilde erişmelerini sağlamak amacıyla tasarlanmıştır. Bu sayede, uygulamanın kullanıcı dostu arayüzü, projenin genel performansını artırarak veri yönetimi süreçlerini daha da güçlendirir.

Uygulamayı aktif olarak kullanan kullanıcıların bireysel bilgilerinin güvenli bir şekilde saklandığı özel bir "kullanıcı" tablosu mevcuttur. Bu tablo, uygulamanın kullanıcı yönetimini sağlayan ve her bir kullanıcının benzersiz kimlik bilgilerini, oturum durumlarını ve diğer önemli detayları içeren bir veri deposudur. Her kullanıcı, bu tablo içerisinde ayrı bir kayıt olarak yer almaktadır.

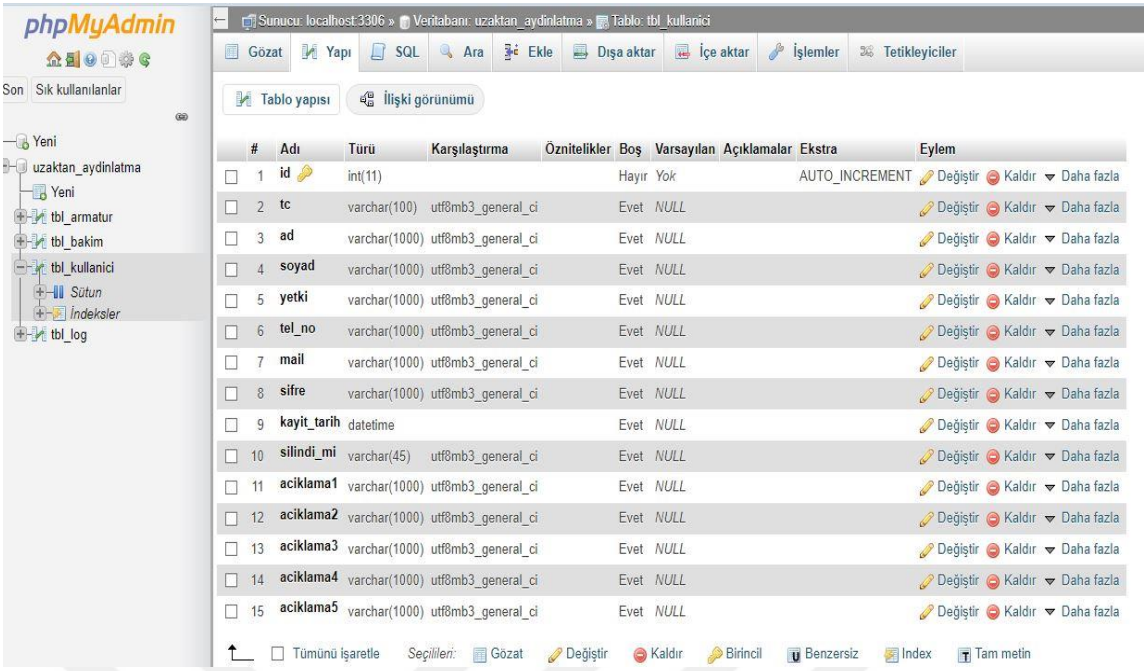


Şekil 4.12. Armatür tablosu ekranı

Uygulama içerisinde, bu özel "kullanıcı" tablosunun ve ilgili alanlarının tamamını gösteren bir ekran bulunmaktadır. Şekil 4.13'te bu ekranı detaylı bir şekilde sunarak kullanıcılara tüm kayıtlı kullanıcıların bilgilerini içeren genel bir bakış sunar. Bu sayede, uygulama yöneticileri veya yetkililer, kullanıcıların kimlik bilgileri, kayıt tarihleri, yetkilendirme seviyeleri gibi detayları kolayca gözden geçirebilir ve gerektiğinde bu bilgileri güncelleyebilir.

Bu özel ekran, kullanıcı yönetimi ve güvenliği açısından önemli bir araçtır. Kullanıcı bilgilerinin bu şekilde düzenli bir şekilde görüntülenmesi, uygulamanın güvenliğini artırır ve yöneticilere etkili bir kullanıcı yönetimi sağlar. Bu da projenin genel performansını güçlendirir ve kullanıcıların uygulamayı daha etkili bir şekilde yönetmelerine olanak tanır.

Uygulamayı aktif olarak kullanan kullanıcılar arasında özellikle bakım personellerinin gerçekleştirdiği bakım işlemlerine dair detaylı bilgilerin saklandığı özel bir "bakım" tablosu bulunmaktadır. Bu tablo, bakım personellerinin hangi armatürlere müdahalede bulunduklarına dair kayıtları içerir. Her bir bakım kaydı, ilgili armatürün kimlik bilgilerini, bakım tarihini, gerçekleştirilen işlemleri ve bakım personelinin bilgilerini içerir.

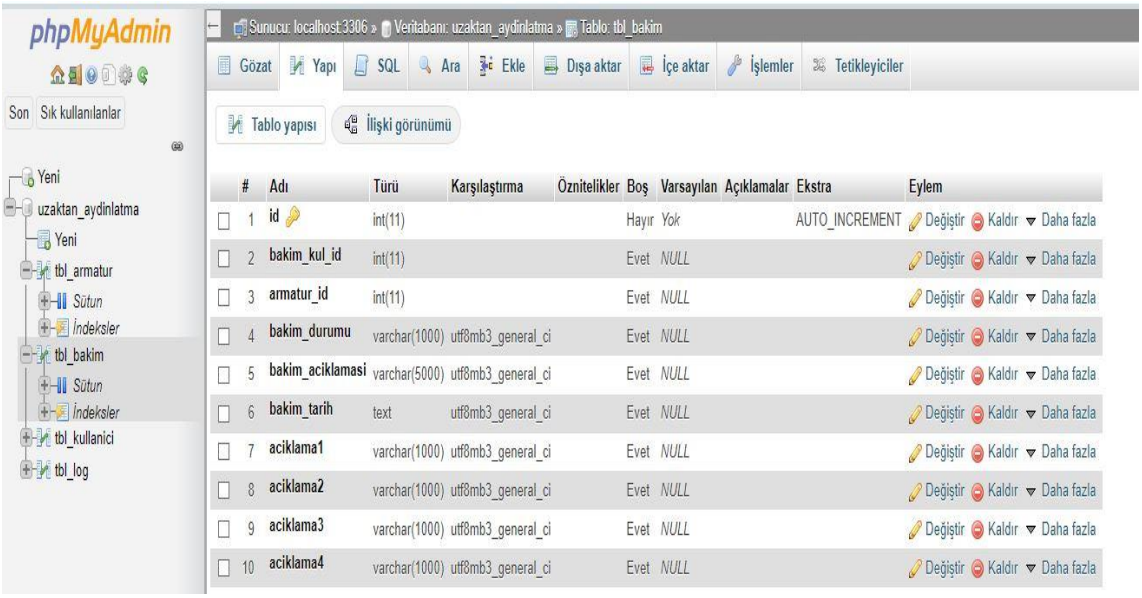


Şekil 4.13. Kullanıcılar tablosu ekranı

Uygulama içerisinde, bu özel "bakım" tablosunun ve ilgili alanlarının tamamını gösteren bir ekran bulunmaktadır. Şekil 4.14'te bu ekranı detaylı bir şekilde sunarak kullanıcılara bakım personellerinin gerçekleştirdiği bakım işlemleri ile ilgili genel bir bakış sağlar. Bu sayede, yöneticiler veya ilgili ekipler, hangi armatürlerin ne zaman bakıma alındığı, hangi bakım personelinin müdahalede bulunduğu gibi detayları kolayca takip edebilir ve gerektiğinde bu bilgileri güncelleyebilir.

Bu özel ekran, bakım süreçlerinin yönetimi ve izlenmesi açısından önemlidir. Bakım tablosu, uygulamayı kullanan kullanıcılar arasında etkili bir iş birliği ve bakım süreçlerinin şeffaf bir şekilde yönetilmesine olanak tanır. Bu da projenin genel güvenilirliğini ve performansını artırır, aynı zamanda kullanıcıların bakım işlemlerini daha etkili bir şekilde yönetmelerine olanak sağlar.

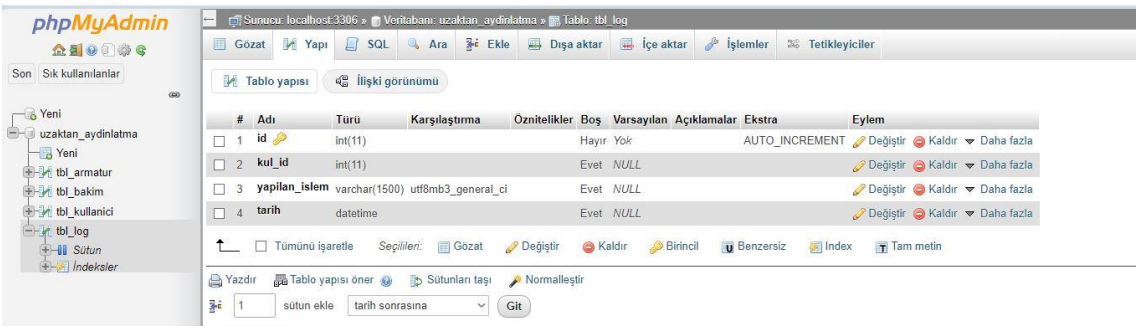
Uygulamayı aktif olarak kullanan kullanıcılar ve armatürlerle ilgili yapılan çeşitli işlemlere dair detaylı bilgilerin saklandığı özel bir "log" tablosu mevcuttur. Bu tablo, kullanıcıların uygulama üzerinde gerçekleştirdikleri eylemleri ve armatürlerle ilgili yapılan değişiklikleri içeren kayıtları barındırır. Her bir log kaydı, ilgili kullanıcının veya sistem tarafından gerçekleştirilen işlemin tarihini, yapılan değişiklikleri ve ilgili armatürün bilgilerini içerir.



Şekil 4.14. Bakım tablosu ekranı

Uygulama içerisinde, bu özel "log" tablosunun ve ilgili alanlarının tamamını gösteren bir ekran bulunmaktadır. Şekil 4.15'te bu ekranı detaylı bir şekilde sunarak kullanıcılara uygulama üzerindeki tüm etkileşimleri ve armatürlerle ilgili yapılan değişiklikleri içeren genel bir bakış sunar. Bu sayede, yöneticiler veya ilgili ekipler, uygulamayı kullanan her bir kullanıcının gerçekleştirdiği işlemleri ve armatürler üzerindeki değişiklikleri kolayca takip edebilir.

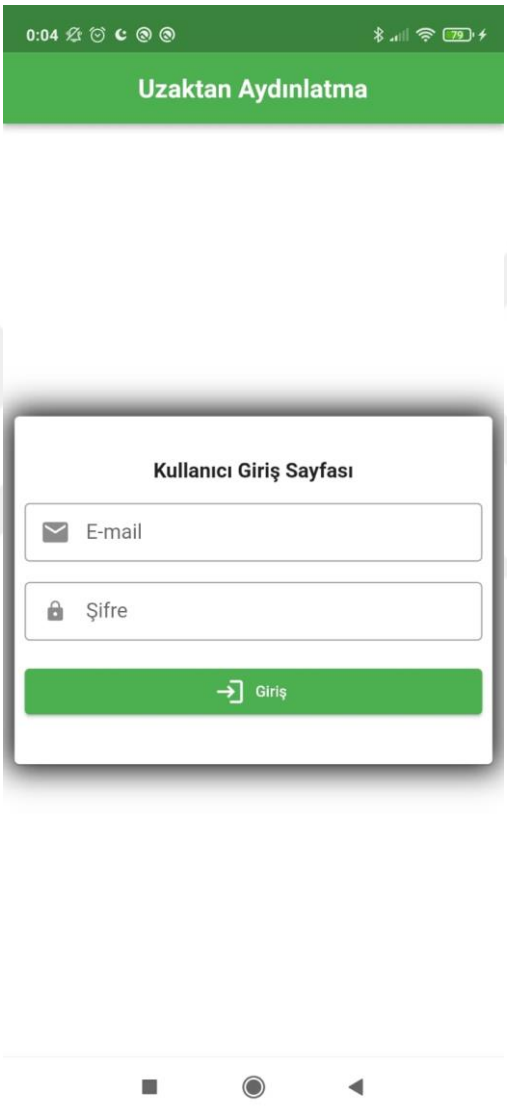
Bu özel ekran, uygulamanın güvenliği ve izlenebilirliği açısından kritik bir öneme sahiptir. Log tablosu, kullanıcıların uygulama üzerindeki etkileşimlerini kaydederek, herhangi bir hata ayıklama sürecinde veya güvenlik denetimlerinde değerli bir kaynak sunar. Ayrıca, bu log bilgileri, uygulama performansını izlemek ve gerektiğinde kullanıcıları bilgilendirmek için kullanılabilir. Bu da projenin genel güvenilirliğini artırır ve kullanıcıların uygulamayı daha etkili bir şekilde kullanmalarına olanak tanır.



Şekil 4.15. Log tablosu ekranı

4.6. Flutter

Projede kullanılacak olan mobil uygulamanın sayfaları, Flutter kullanılarak hazırlanmıştır. Kullanıcılara yönelik doğrulama sürecini başlatmak amacıyla ilk olarak e-mail ve şifre ile giriş yapılacak bir ekran tasarlanmıştır. Şekil 4.16'da gösterilen bu giriş ekranı, kullanıcıların kimliklerini doğrulamalarına ve uygulamaya erişim sağlamalarına olanak tanır.

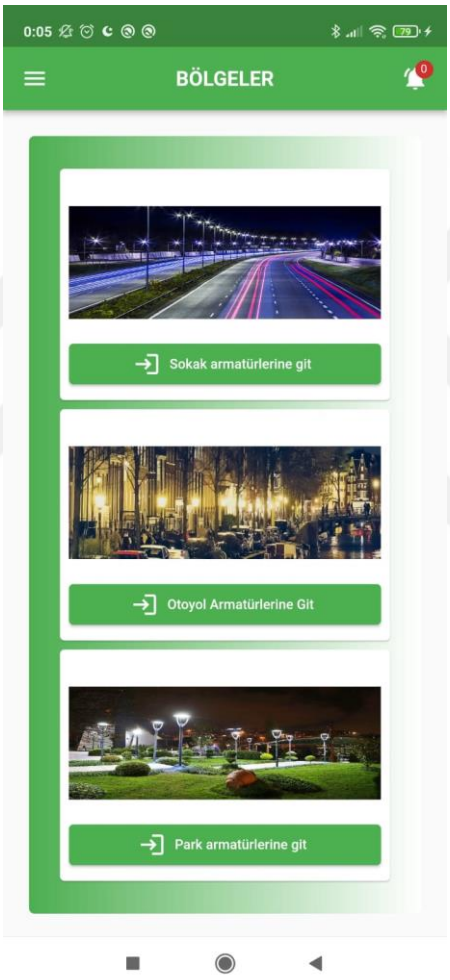


Şekil 4.16. Kullanıcı giriş ekranı

Uygulamada, kullanıcıların giriş yapmasının ardından, her bir kullanıcının tanımlanmış olan yetkilerine göre özel ekranlar gösterilir. Bu yetkiler, kullanıcıların uygulama içinde hangi işlemleri gerçekleştirebileceğini belirler. Örneğin, bir kullanıcı

yönetici yetkisine sahipse, farklı ekranlara ve işlevlere erişim sağlayabilirken, izleyici yetkisine sahip olan bir kullanıcı sadece izleme yetkisine sahip işlemleri yapabilmektedir.

Bu giriş ekranı, uygulamanın güvenliğini artırmak ve kullanıcıların kişisel bilgilerini korumak amacıyla tasarlanmıştır. Kullanıcılar, e-mail ve şifreleri aracılığıyla güvenli bir şekilde giriş yaparak, uygulamanın sunduğu özelliklere erişebilirler. Bu da projenin genel kullanıcı deneyimini iyileştirirken, aynı zamanda güvenlik standartlarını da sağlamaya yönelik bir adımdır.



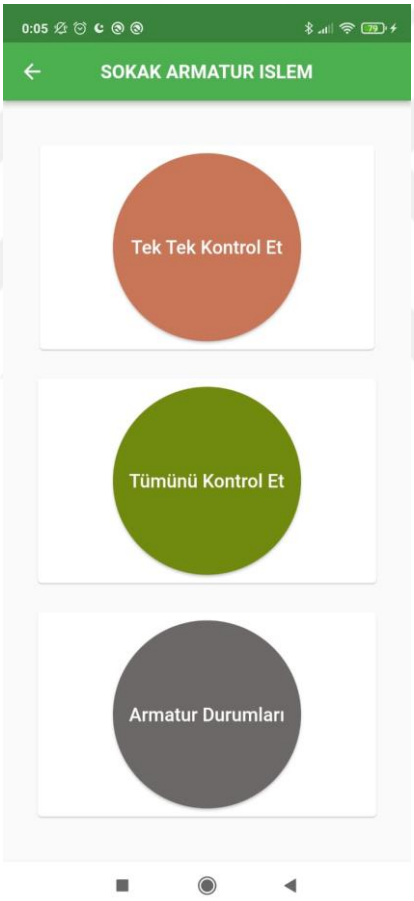
Şekil 4.17. Armatür bölgeler ekranı

Kullanıcının giriş bilgileri doğru olduğunda, uygulama tarafından belirlenmiş olan yetkilere bağlı olarak kullanıcının ilk karşılaşacağı ekranlar, özel olarak tasarlanmıştır. Bu ekranlar, kullanıcının uygulama içindeki işlevselliklere hızlı ve etkili bir şekilde erişim sağlamasını amaçlar. Şekil 4.17'de gösterilen armatür bölgeleri ekranı,

kullanıcının belirli işlemleri gerçekleştirmek için kullanabileceği farklı bölge seçeneklerini içeren bir genel bakış sunar.

Bu bölgeler ekranı, kullanıcının profiline ve yetkilerine bağlı olarak değişen özellikleri içerir. Örneğin, bir kullanıcı yönetici yetkisine sahipse, bu ekran üzerinden çeşitli yönetim işlemlerini gerçekleştirebilir. Diğer yandan, izleyici yetkisine sahip bir kullanıcı için bu ekran, sadece uygulamanın sunduğu temel özelliklere erişimi içerir.

Bu tasarım, kullanıcıların ihtiyaçlarına uygun olarak özelleştirilmiş bir kullanıcı arayüzü sunarak, uygulama içindeki deneyimlerini optimize etmeyi amaçlar. Bu da projenin genel kullanılabilirliğini artırır ve kullanıcıların uygulamayı daha etkili bir şekilde kullanmalarını sağlar.



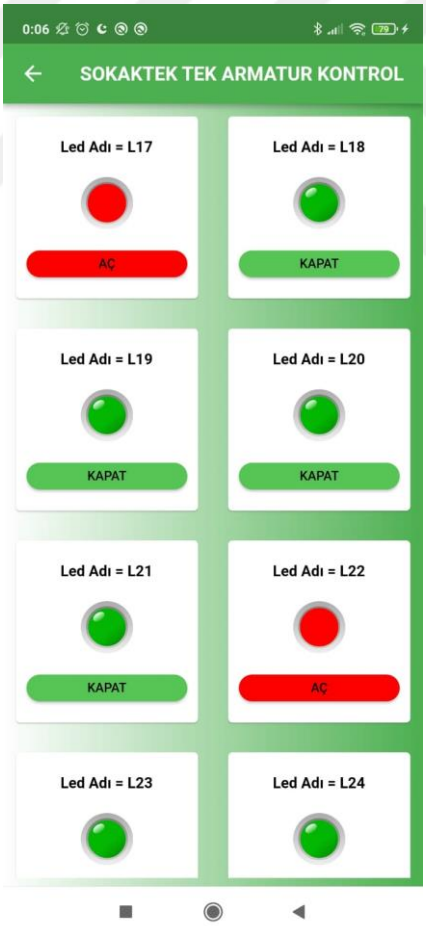
Şekil 4.18. Sokak armatürleri işlem

Kullanıcı, Sokak Armatürleri hakkında bir dizi işlem gerçekleştirmek istediğinde, uygulamada bulunan "Sokak Armatürleri" bölgesine yönlendiren bir "Git" butonuna tıkladığında, özel olarak tasarlanmış olan Sokak Armatürleri işlem ekranı kullanıcıya

gösterilir. Şekil 4.18'de gösterilen sokak armatür işlem ekranını, kullanıcının Sokak Armatürleri ile ilgili gerçekleştirebileceği çeşitli işlemleri içeren bir genel bakış sunar.

Sokak Armatürleri işlem ekranı, kullanıcının bu özel bölge içindeki armatürlerle ilgili çeşitli eylemleri gerçekleştirmesine olanak tanır. Örneğin, kullanıcılar bu ekran üzerinden belirli bir armatürün durumunu kontrol edebilir, tüm armatürleri kontrol edebilir veya armatürlerin durumlarını izleyebilir. Kullanıcı dostu bir arayüz ve anlaşılır simgelerle desteklenen bu ekran, kullanıcıların Sokak Armatürleriyle etkileşimde bulunmalarını kolaylaştırır.

Bu özel işlem ekranı, kullanıcıların uygulamayı daha etkili bir şekilde kullanmalarını sağlar ve Sokak Armatürleriyle ilgili işlemleri yönetmelerine olanak tanır. Bu tasarım, kullanıcı deneyimini geliştirmeyi hedefler ve kullanıcılara belirli bir bölgedeki armatürlerle ilgili işlemleri anlamalarını ve yönetmelerini kolaylaştırır.

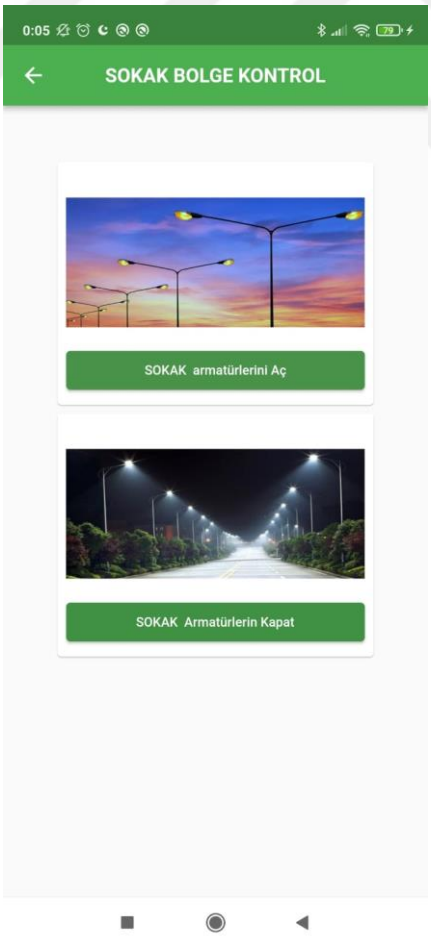


Şekil 4.19. Sokak armatürleri tek tek açma kapatma

Kullanıcı, Sokak Armatürleri üzerinde tek tek açma ve kapatma işlemlerini gerçekleştirmek istediğinde, uygulamada bulunan özel "Sokak Armatürleri Tek Tek Açma/Kapatma" butonuna tıklandıktan sonra, Sokak Armatürleri Tek Tek Açma/Kapatma ekranı kullanıcıya gösterilir. Şekil 4.19'da gösterilen sokak armatürleri tek tek açma kapatma ekranı, kullanıcının her bir sokak armatürünü ayrı ayrı açma veya kapatma işlemlerini gerçekleştirmesine olanak tanıyan bir arayüz sunar.

Bu ekran, kullanıcılara her bir armatürü tek tek kontrol etme ve yönetme imkânı sağlar. Kullanıcılar, bu ekran üzerinden sokak bölgesinin armatürünü seçerek, onu açabilir veya kapatabilir. Ayrıca, armatürlerin durumu hakkında bilgi alabilir.

Bu tasarım, kullanıcılara Sokak Armatürleri ile etkileşimde bulunma konusunda hassas bir kontrol sağlar. Kullanıcı dostu arayüz ve basit bir kullanım mantığı, kullanıcıların her bir armatürün durumunu anlamalarını ve ihtiyaçlarına uygun şekilde yönetmelerini kolaylaştırır. Bu da projenin genel kullanılabilirliğini artırır ve kullanıcıların sokak aydınlatma sistemini daha etkili bir şekilde yönetmelerine olanak tanır.



Şekil 4.20. Sokak armatürleri tümünü açıp kapatma

Kullanıcı, Sokak Armatürleri üzerinde tümünü açma ve kapatma işlemlerini gerçekleştirmek istediğinde, uygulamada bulunan özel "Sokak Armatürleri Tümünü Aç/Kapat" butonuna tıklandıktan sonra, Sokak Armatürleri Tümünü Aç/Kapat ekranı kullanıcıya gösterilir. Şekil 4.20'de gösterilen bu sokak armatürleri tümünü aç kapat ekranı, kullanıcının tüm sokak armatürlerini aynı anda açma veya kapatma işlemlerini kolayca gerçekleştirebilmesini sağlayan bir arayüz sunar.

Bu ekran, kullanıcının büyük bir alandaki tüm armatürleri tek bir eylemle kontrol etmesine olanak tanır. Kullanıcılar, bu ekran üzerinden tüm armatürleri topluca açabilir veya kapatabilirler. Aynı zamanda, tüm armatürlerin durumu hakkında anlık bilgileri görüntüleyebilirler.

Bu tasarım, kullanıcılara pratik bir kontrol sağlayarak, büyük alanlarda enerji tasarrufu veya acil durumlarda hızlı müdahale gibi senaryolarda etkili bir şekilde kullanımını sağlar. Kullanıcı dostu arayüz, kullanıcıların tüm armatürleri tek bir ekranda yönetmelerini sağlayarak işlemleri basitleştirir ve projenin genel kullanılabilirliğini artırır.

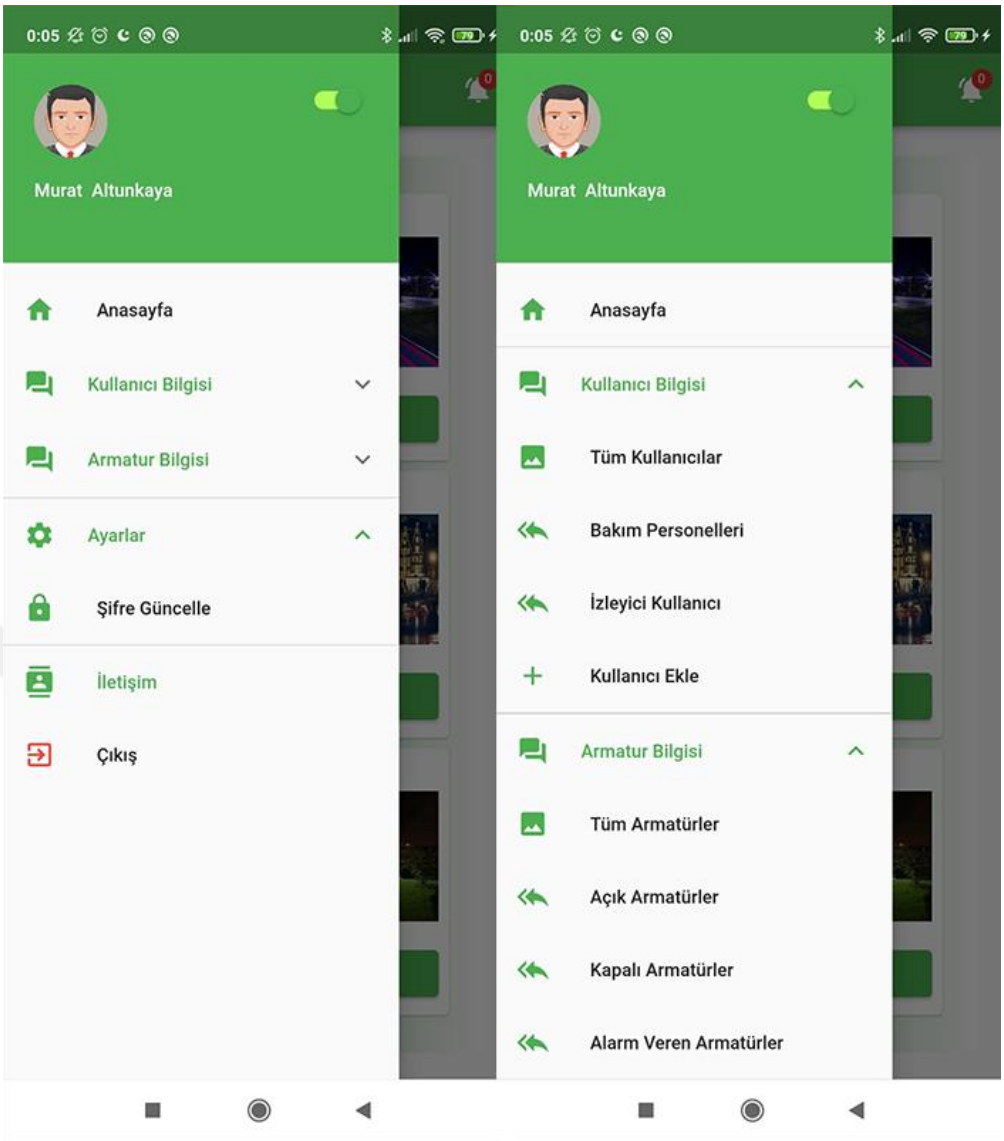
Kullanıcının Sokak Armatürlerinin tümünü izleme ihtiyacını karşılamak amacıyla farklı bir ekran hazırlanmıştır. Kullanıcı, bu özellik için tasarlanmış olan "Sokak Armatürleri Tümünü İzleme" butonuna tıkladığında, Sokak Armatürleri Tümünü İzleme ekranı kullanıcıya gösterilir. Şekil 4.21'de gösterilen sokak armatürlerini izleme ekranı, kullanıcının tüm sokak armatürlerinin anlık durumunu detaylı bir şekilde gözlemleyebilmesine imkân tanıyan bir arayüz sunar.

Bu ekran, tüm armatürlerin güncel durum bilgilerini içerir. Bu sayede, kullanıcılar geniş bir alanı etkili bir şekilde izleyebilir ve gerektiğinde armatürlerin durumuyla ilgili anında bilgi alabilirler.

Şekil 4.21. Sokak armatürleri tümünü izleme

Kullanıcının uygulamada gerçekleştirebileceği çeşitli işlemleri anlamak ve erişim sağlamak amacıyla tasarlanmış "Uygulama Menü" ekranı Şekil 4.22'de gösterilmiştir. Bu menü, kullanıcının uygulama içindeki çeşitli özelliklere ve işlevlere erişimini kolaylaştıran bir arayüz sağlar.

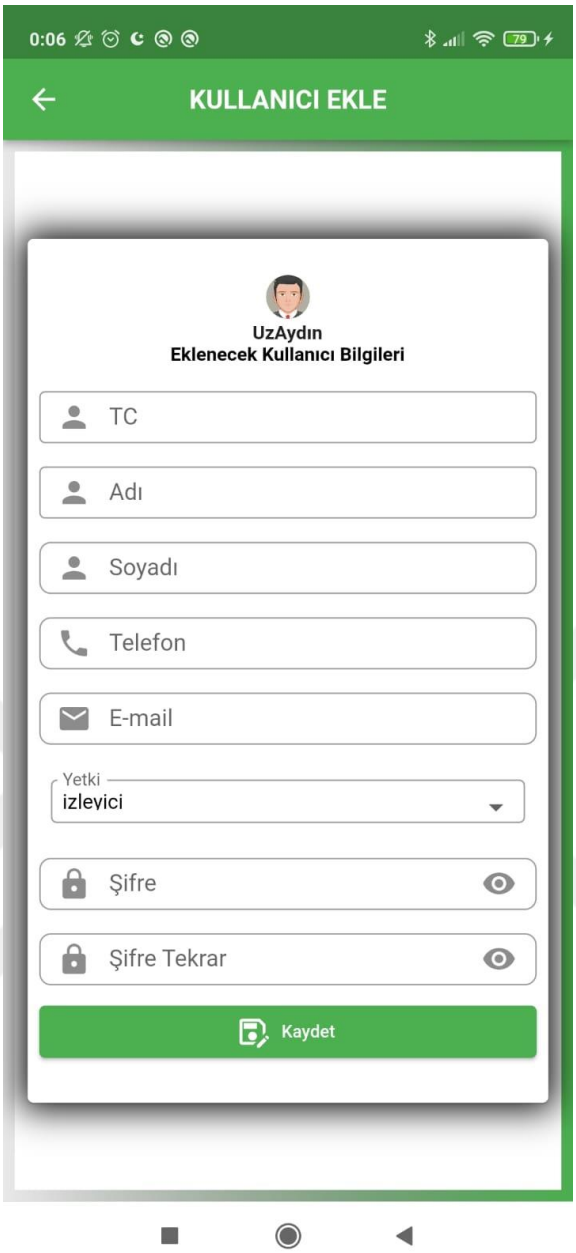
Uygulama Menü ekranı, kullanıcılara farklı bölümlere hızlı bir şekilde erişim sağlama imkânı sunar. Bu bölümler, genellikle kullanıcı profilini yönetme, armatürlerle ilgili işlemleri gerçekleştirme ve izleme gibi temel işlevleri içerir. Kullanıcılar, bu menü üzerinden ihtiyaçlarına uygun bir şekilde hareket edebilir, uygulamanın genel özelliklerini keşfedebilir ve istedikleri bölüme kolayca geçiş yapabilirler.



Şekil 4.22. Uygulama menü ekranı

Yönetici yetkisine sahip kullanıcı, uygulamaya farklı bir kullanıcı eklemek istediğinde, bu işlemi gerçekleştirmek amacıyla özel olarak tasarlanmış olan "Kullanıcı Ekleme" ekranına yönlendirilir. Şekil 4.23'te gösterilen kullanıcı ekleme ekranı, kullanıcının eklemek istediği yeni kullanıcıyı belirlerken aynı zamanda kullanıcının hangi rolde olacağını seçmesine olanak tanıyan bir arayüz sunar.

Kullanıcı Ekleme ekranı, kullanıcılara yeni bir kullanıcı oluştururken belirli bir rol seçme esnekliği sunar. Kullanıcı, eklenen kullanıcının yetkilerini belirleyerek, ona özel bir rol atayabilir. Örneğin, yönetici, bakım personeli veya izleyici kullanıcı gibi farklı roller arasından seçim yapabilir. Bu da uygulamanın güvenliği ve kullanıcı yönetimi açısından önemli bir esneklik sağlar.

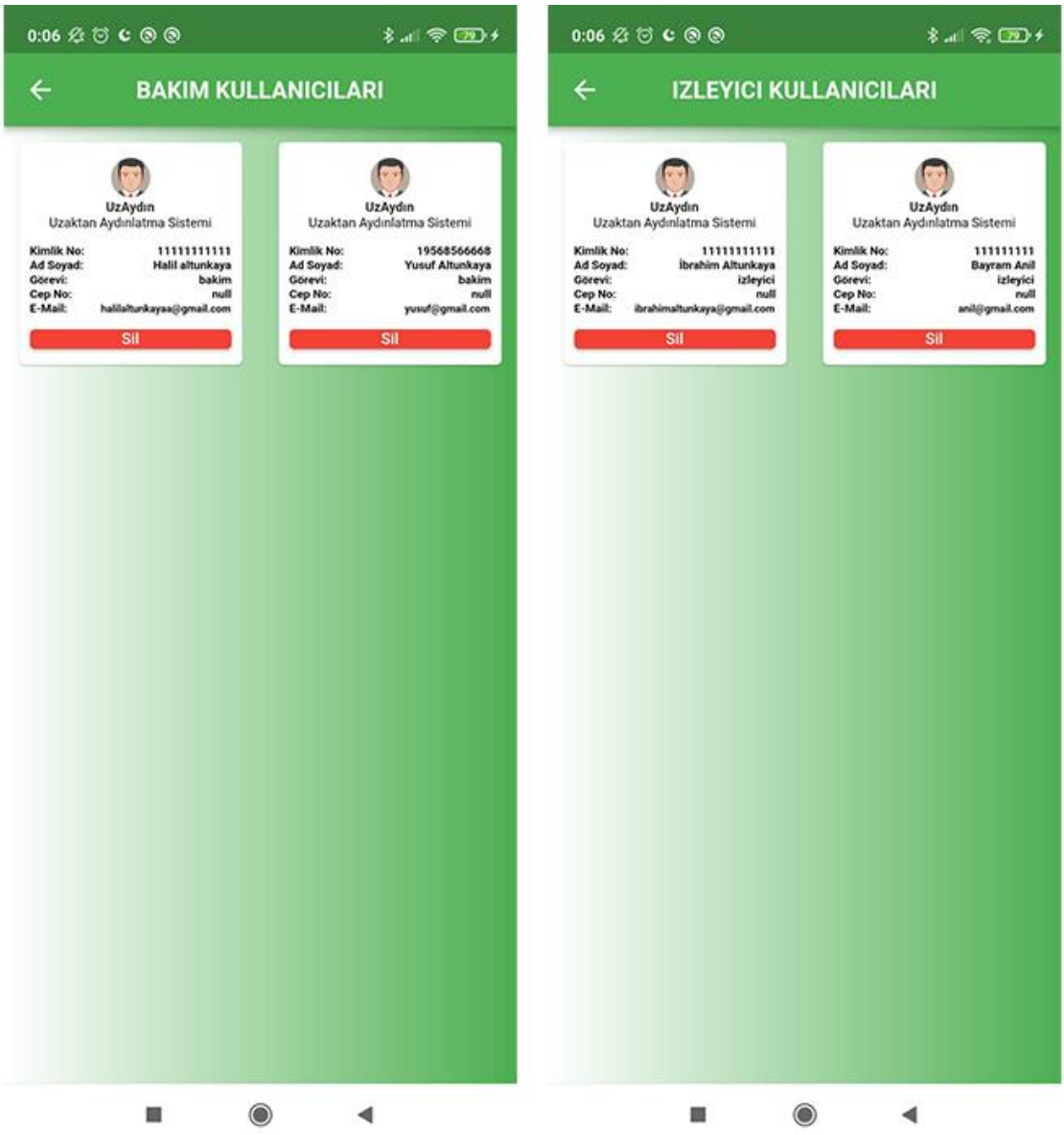


Şekil 4.23. Kullanıcı ekleme ekranı

Kullanıcı, eklenmiş olan armatürlerin bakım personellerini ve izleyici kullanıcı listesini inceleyerek, gerekli düzenlemeleri yapabilmektedir. Şekil 4.24'te bu işlevselliği içeren bakım personel listesi ve izleyici kullanıcı listesi ekranı gösterilmiştir. Bu, kullanıcının armatürlerle ilgili bakım ve izleme sorumluluklarına sahip olan personelleri detaylı bir şekilde görmesine olanak tanır. Bu ekran ile kullanıcılar bakım personellerini ve izleyici kullanıcıları görüntüleyebilir, düzenleyebilir veya gerektiğinde silebilirler.

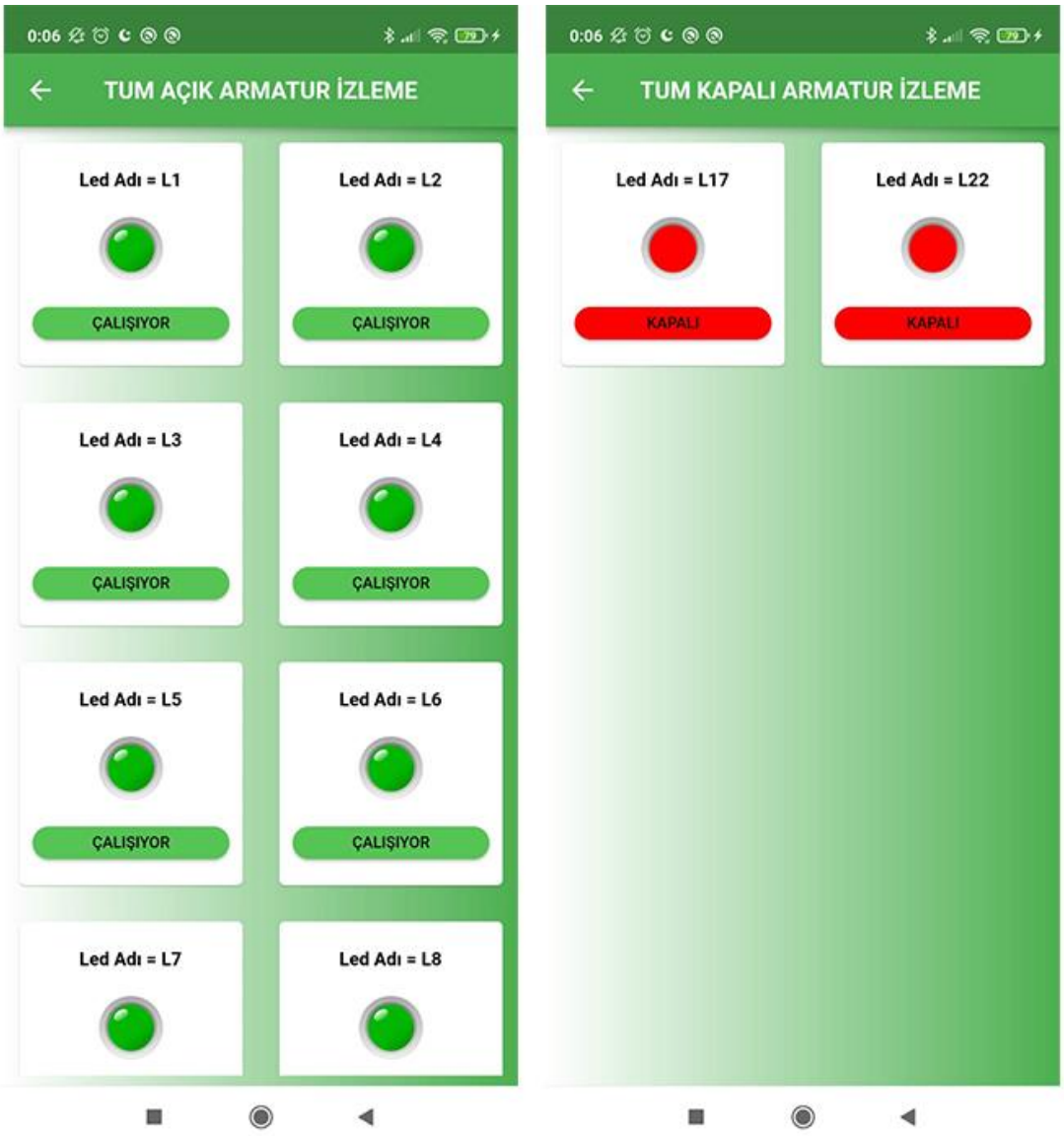
Bakım Personel Listesi ekranı, kullanıcılara eklenmiş olan personelleri rol bazlı bir şekilde yönetme yeteneği sunar. Bu, armatürlerin bakımı ve izlenmesiyle ilgili

görevleri belirli kullanıcılara atamak ve bu görevleri etkili bir şekilde yönetmek için kullanılır. Kullanıcılar, bu ekran üzerinden her bir personelin rolünü, iletişim bilgilerini ve diğer ilgili bilgilerini görüntüleyerek, gerektiğinde güncelleme veya silme işlemlerini gerçekleştirebilirler.



Şekil 4.24 Bakım ve izleyici kullanıcı listesi ekranı

Kullanıcı, uygulama içindeki Sokak Armatürleri üzerindeki durumu etkili bir şekilde gözlemlemek için özel olarak tasarlanmış olan "Açık ve Kapalı Sokak Armatürleri Tümünü İzleme" özelliğini kullanabilir. Şekil 4.25'te açık ve kapalı armatürleri izleme ekranını gösterilmiştir ve bu arayüz kullanıcının tüm sokak armatürlerinin açık veya kapalı durumunu anlık olarak görüntülemesine olanak tanır.



Şekil 4.25 Açık ve kapalı sokak armatür izleme ekranı

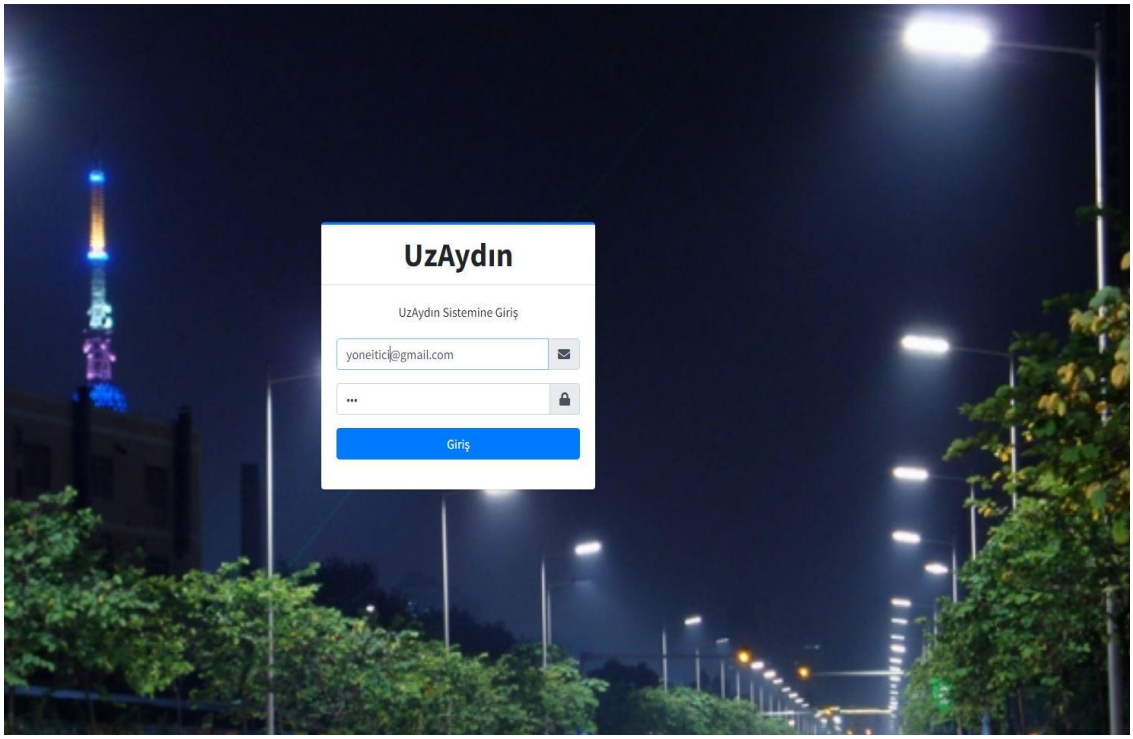
4.7. Web

Projede kullanılacak web uygulamasının sayfaları hazırlanmış ve ASP.NET MVC ile kodlaması yapılmıştır. Kullanıcılara yönelik doğrulama sürecini başlatmak amacıyla ilk olarak e-mail ve şifre ile giriş yapılacak bir ekran tasarlanmıştır. Bu giriş ekranı, kullanıcıların kimliklerini doğrulamalarına ve uygulamaya erişim sağlamalarına olanak tanır. Şekil 4.26'da gösterilen bu kullanıcı giriş ekranını, uygulamanın temel erişim noktasını kullanıcılara sunar.

Uygulamada, kullanıcıların giriş yapmasının ardından, her bir kullanıcının tanımlanmış olan yetkilerine göre özel ekranlar gösterilir. Bu yetkiler, kullanıcıların

uygulama içinde hangi işlemleri gerçekleştirebileceğini belirler. Örneğin, belirli bir kullanıcı yönetici yetkisine sahipse, farklı ekranlara ve işlevlere erişim sağlanabilirken, izleyici yetkisine sahip bir kullanıcı sadece izleme işlemlerini yapabilmektedir.

Bu giriş ekranı, uygulamanın güvenliğini artırmak ve kullanıcıların kişisel bilgilerini korumak amacıyla tasarlanmıştır. Kullanıcılar, e-mail ve şifreleri aracılığıyla güvenli bir şekilde giriş yaparak, uygulamanın sunduğu özelliklere erişebilirler. Bu da projenin genel kullanıcı deneyimini iyileştirirken, aynı zamanda güvenlik standartlarını da sağlamaya yönelik bir adımdır.

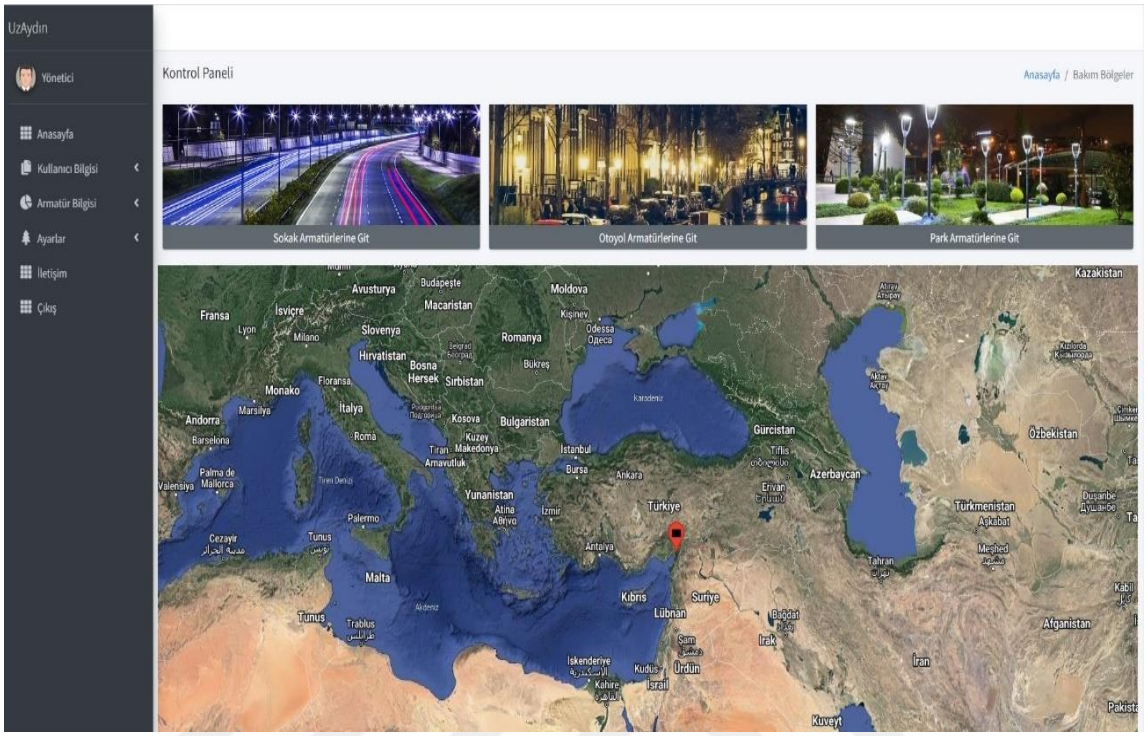


Şekil 4.26. Web Giriş Ekranı

Kullanıcının giriş bilgileri doğru olduğunda, uygulama tarafından belirlenmiş olan yetkilere bağlı olarak kullanıcının ilk karşılaşacağı web anasayfa ekranı Şekil 4.27'de gösterilmiştir. Bu ekran, kullanıcının uygulama içindeki işlevselliklere hızlı ve etkili bir şekilde erişim sağlamasını amaçlar. Ayrıca, kullanıcının belirli işlemleri gerçekleştirmek için kullanabileceği farklı bölgeleri ve haritada armatürlerin eklendiği konumların bilgilerini içerir.

Bu bölgeler ekranı, kullanıcının profiline ve yetkilerine bağlı olarak değişen özellikleri içerir. Örneğin, bir kullanıcı yönetici yetkisine sahipse, bu ekran üzerinden

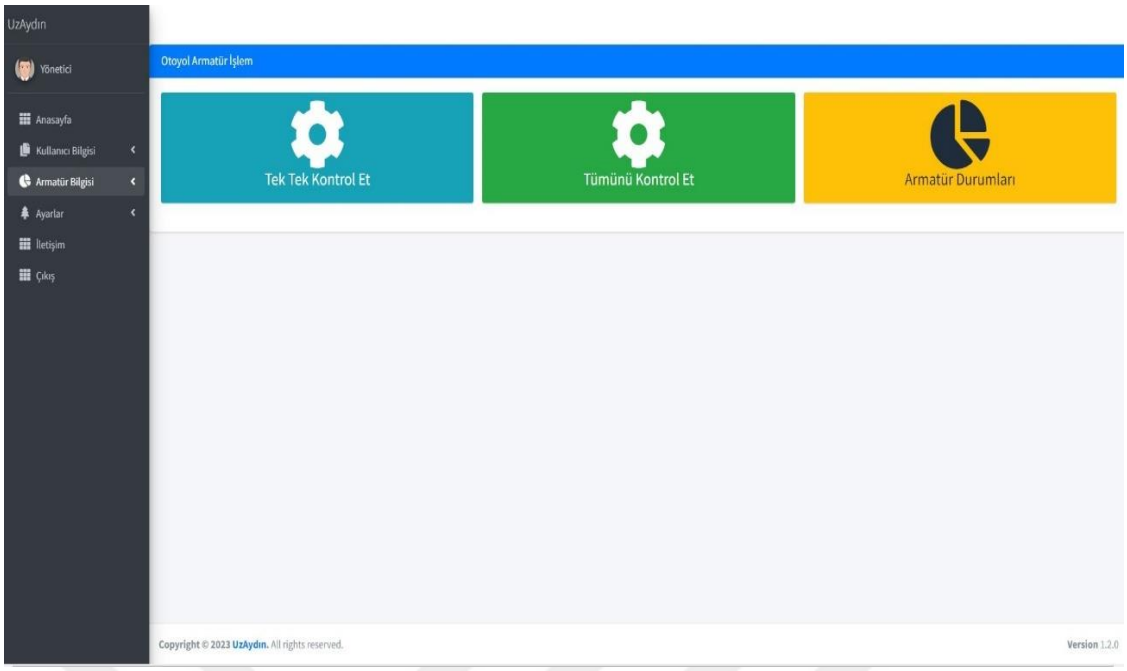
çeşitli yönetim işlemlerini gerçekleştirebilir. Diğer yandan, bakım personeli veya izleyici bir kullanıcı için bu ekran, sadece uygulamanın sunduğu temel özelliklere erişimi içerir.



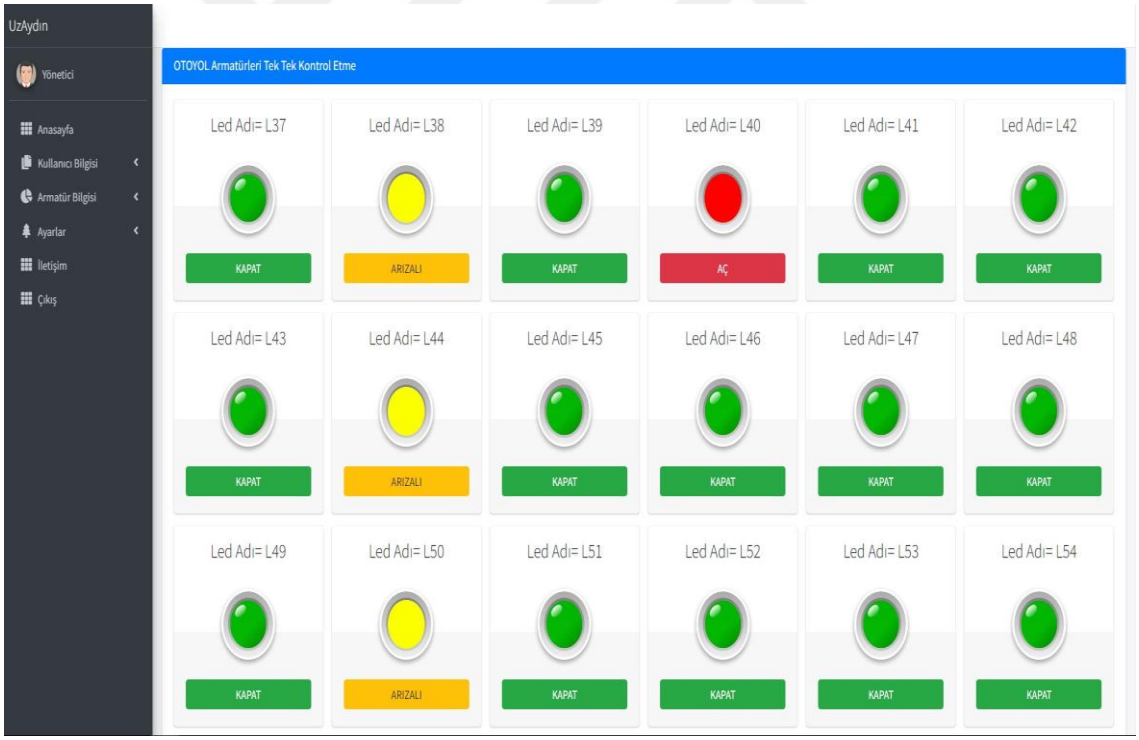
Şekil 4.27 Web anasayfa

Kullanıcı, Otoyol armatürleri hakkında bir dizi işlem gerçekleştirmek istediğinde, uygulamada bulunan "Otoyol armatürleri" bölgesine yönlendiren bir "Git" butonuna tıklar ve özel olarak tasarlanmış olan otoyol armatürleri işlem ekranı kullanıcıya gösterilir. Şekil 4.28'de gösterilen otoyol armatür işlem ekranını, kullanıcının Otoyol armatürleri ile ilgili gerçekleştirebileceği çeşitli işlemleri içeren genel bir bakış sunar.

Otoyol armatürleri işlem ekranı, kullanıcının bu özel bölge içindeki armatürlerle ilgili çeşitli eylemleri gerçekleştirmesine olanak tanır. Örneğin, kullanıcılar bu ekran üzerinden belirli bir armatürün veya tüm armatürlerin durumunu kontrol edebilir ya da armatürlerin durumlarını izleyebilir. Kullanıcı dostu bir arayüz ve anlaşılır simgelerle desteklenen bu ekran, kullanıcıların Otoyol armatürleriyle etkileşimde bulunmalarını kolaylaştırır.



Şekil 4.28. Otoyol armatür işlem ekranı

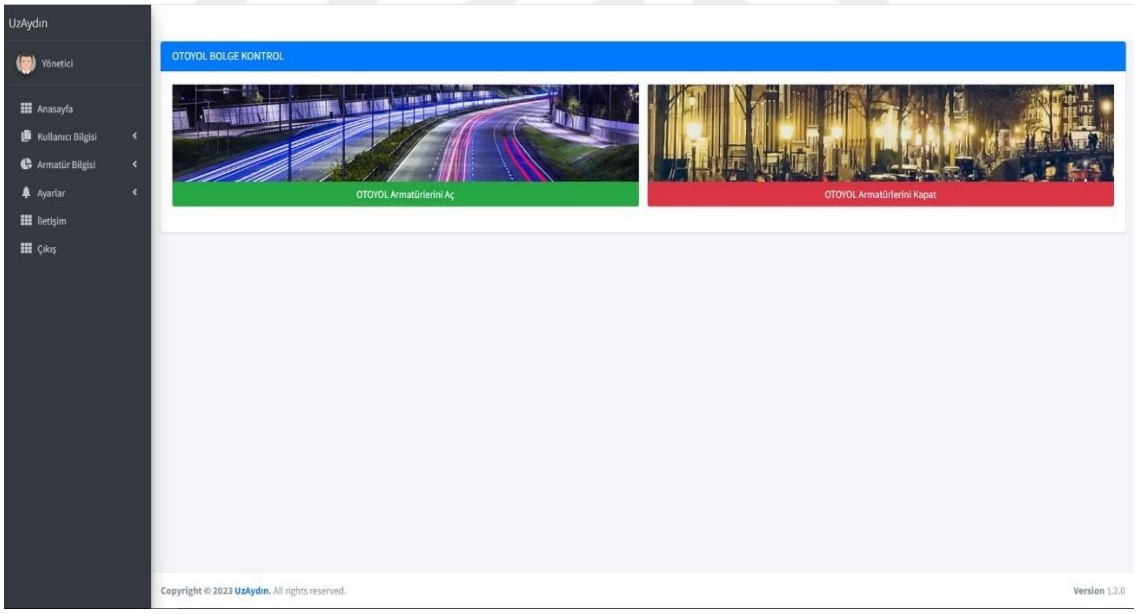


Şekil 4.29. Otoyol armatür tek tek kontrol ekranı

Kullanıcı, Otoyol armatürleri üzerinde tek tek açma ve kapatma işlemlerini gerçekleştirmek istediğinde, uygulamada bulunan özel "Otoyol Armatürleri Tek Tek Açma/Kapatma" butonuna tıklar ve Otoyol Armatürleri Tek Tek Açma/Kapatma ekranı kullanıcıya gösterilir. Şekil 4.29'da gösterilen otoyol armatürleri tek tek açma kapatma

ekranı, kullanıcının her bir otoyol armatürünü ayrı ayrı açma veya kapatma işlemlerini gerçekleştirmesine olanak tanıyan bir arayüz sunar. Bu ekran, kullanıcılara her bir armatürü tek tek kontrol etme imkânı sağlar. Kullanıcılar, bu ekran üzerinden otoyol bölgesinin armatürünü seçerek, onu açabilir veya kapatabilirler. Ayrıca, armatürlerin durumu hakkında bilgi alabilir. Bu tasarım, kullanıcılara Otoyol armatürleri ile etkileşimde bulunma konusunda hassas bir kontrol sağlar.

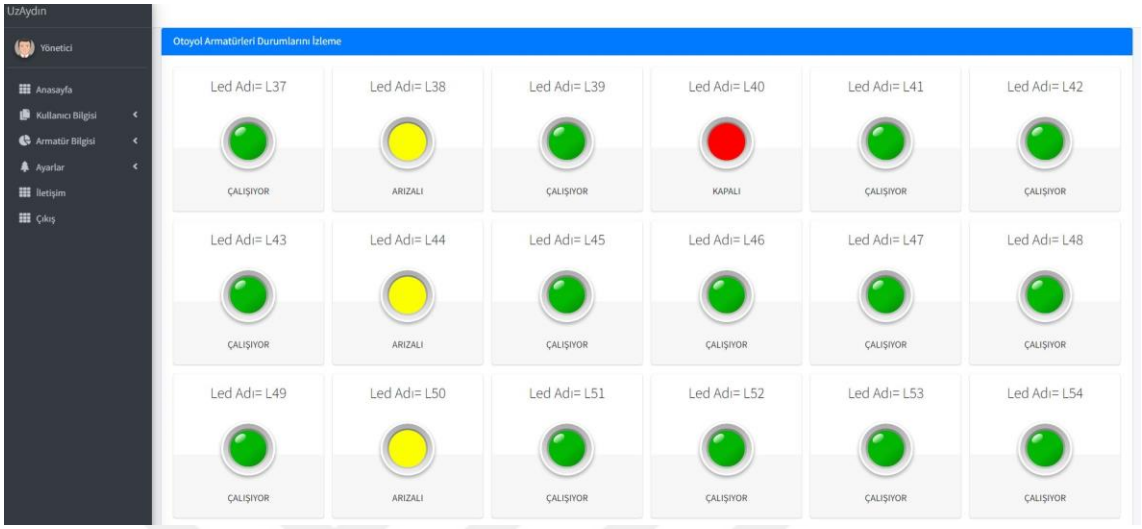
Kullanıcı, Otoyol üzerinde bulunan armatürlerin tümünü açma ve kapatma işlemlerini gerçekleştirmek istediğinde, uygulamada bulunan özel "Otoyol armatürleri tümünü Aç/Kapat" butonuna tıklar ve Otoyol armatürleri tümünü Aç/Kapat ekranı kullanıcıya gösterilir. Şekil 4.30'da gösterilen otoyol armatürleri tümünü aç kapat ekranı, kullanıcının tüm otoyol armatürlerini aynı anda açma veya kapatma işlemlerini kolayca gerçekleştirebilmesini sağlayan bir arayüz sunar. Bu ekran, kullanıcının büyük bir alandaki tüm armatürleri tek bir eylemle kontrol etmesine olanak tanır. Kullanıcılar, bu ekran üzerinden tüm armatürleri topluca açabilir veya kapatabilirler. Aynı zamanda, tüm armatürlerin durumu hakkında anlık bilgileri görüntüleyebilirler.



Şekil 4.30. Otoyol armatür bölge kontrol ekranı

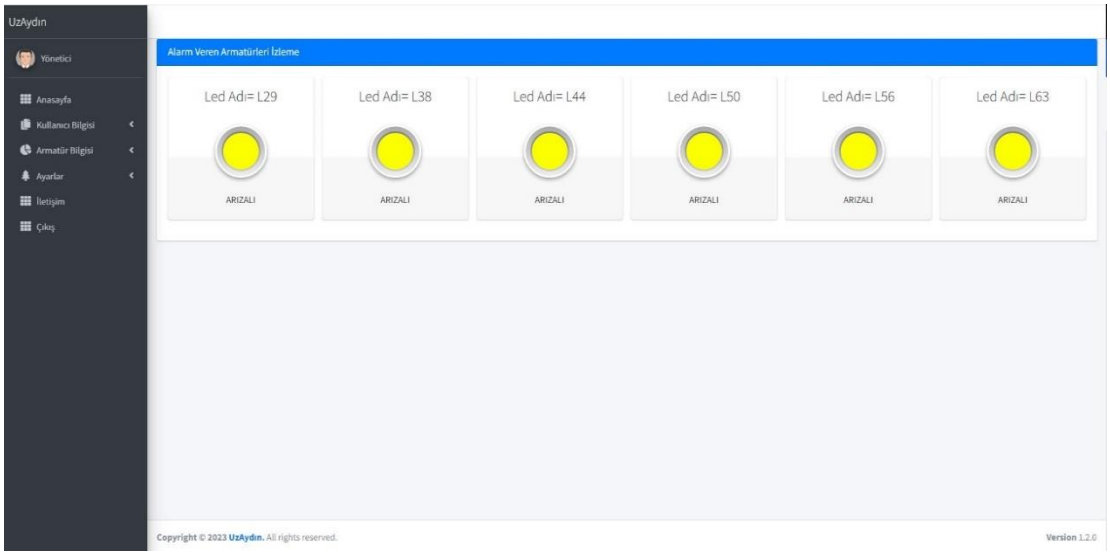
Kullanıcının Otoyol armatürlerinin tümünü izleme ihtiyacını karşılamak amacıyla "Otoyol armatürleri tümünü izleme" sayfası hazırlanmıştır. Şekil 4.31'de gösterilen bu sayfa, kullanıcının tüm otoyol armatürlerinin anlık durumunu detaylı bir şekilde gözlemleyebilmesine imkân tanıyan bir arayüz sunar. Bu ekran, kullanıcılara tüm

armatürlerin güncel durum bilgilerini gösterir. Bu sayede, kullanıcılar geniş bir alanı etkili bir şekilde izleyebilir ve gerektiğinde armatürlerin durumuyla ilgili anında bilgi alabilirler.



Şekil 4.31. Otoyol armatür durum izleme ekranı

Kullanıcı, uygulama içindeki Otoyol armatürleri üzerindeki durumu etkili bir şekilde gözlemlemek için özel olarak tasarlanmış olan "Otoyol alarm veren armatürleri izleme" özelliğini kullanabilir. Şekil 4.32'de gösterilen Otoyol alarm veren armatürleri izleme ekranını, kullanıcının tüm otoyol armatürlerinin alarm veren armatürlerin durumunu anlık olarak görüntülemesine olanak tanıyan bir arayüz sunar.



Şekil 4.32.Tüm alarm veren armatür ekranı

5. ARAŞTIRMA BULGULARI VE TARTIŞMA

Bu çalışma kapsamında donanım ve yazılım için kullanılan bilgisayarın özellikleri Tablo 5.1'de gösterilmiştir.

Tablo 5.1. Çalışmada kullanılan bilgisayar özellikleri

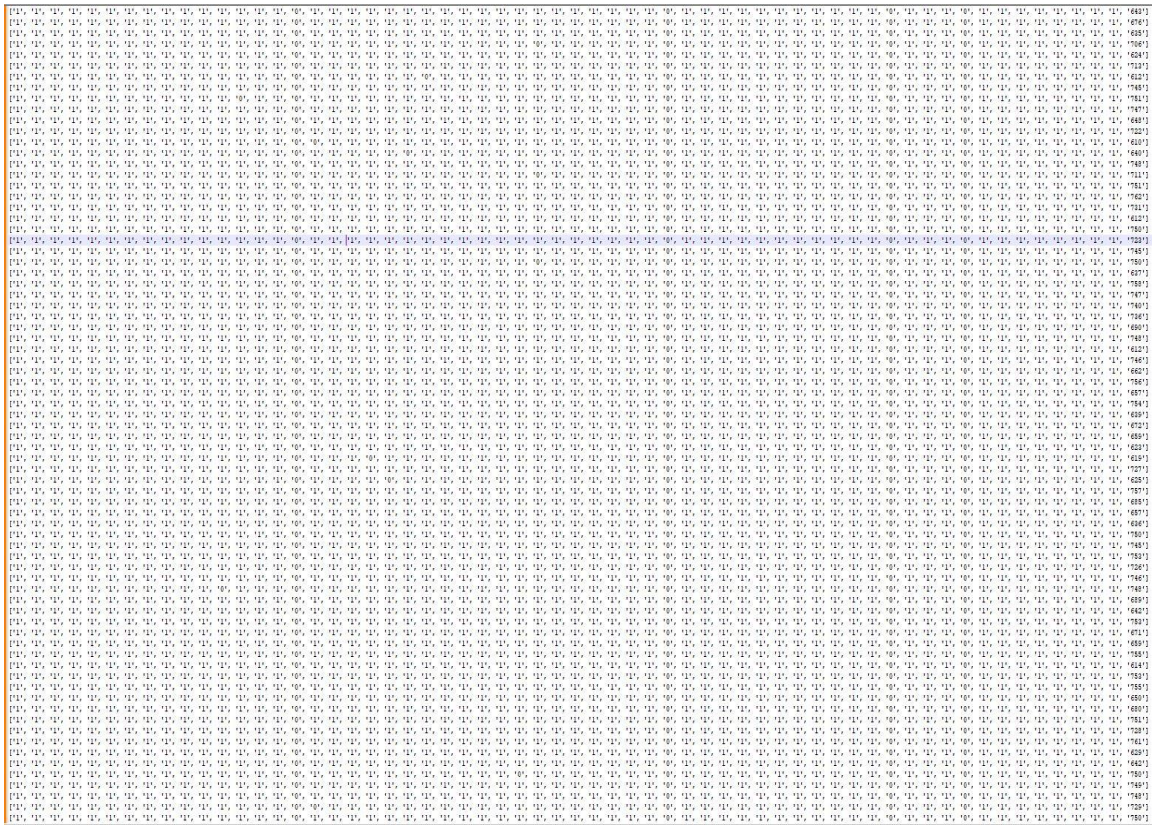
Donanım	**Özellikler**
CPU	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz (12 CPUs)
İşletim Sistemi	Windows 10 Pro 64-bit
Grafik Kartı	NVIDIA GeForce GTX 1650
RAM	32 GB

Bu çalışma, SCADA sistemlerine web ve mobil uygulamalar aracılığıyla uzaktan erişim sağlama imkânı sunar. Bu, kullanıcılara sistem üzerinde kontrol ve izleme yeteneği kazandırarak, aydınlatma sistemini istedikleri zaman ve yerden yönetmelerini sağlar. Fakat sisteme birden fazla kişinin dahil olduğu durumlarda sistemin kontrol edilmesi ve yönetilmesi sistemde kararsızlığa neden olacaktır. Bu karmaşıklığı önlemek için çok rollü kullanıcı sistemi önerilerek sistem içerisinde admin, bakım personeli ve yönetici olmak üzere üç farklı kullanıcı ile izin sistemi oluşturulmuştur.

Otomatik aydınlatma kontrolü, çevresel etkiyi en aza indirir. Işıkların gereksiz yere açık kalmasını önleyerek, çevresel sürdürülebilirliği destekler. LDR sensörü çevresel ışık koşullarını algılamak amacıyla bu projeye entegre edilmiştir. Bu sensör gece veya gündüz olduğunu algıladığında sistemdeki ışıkları otomatik olarak açıp kapatma yeteneği sağlar. Bu da sistemdeki aydınlatmanın insan müdahalesi olmadan otomatik olarak kontrol edilmesini sağlar. Ayrıca, bu çalışmada görüntü işleme teknikleri kullanılarak elde edilen görüntülerin analizi mümkün hale gelmiştir. Bu sayede alınan görüntülerden insan ve araç takibi hızlı bir şekilde yapılmış ve bu bilgiler SCADA sistemine gönderilerek sistemin otomatik olarak çalışması sağlanmıştır. Bu da özellikle gece saatlerinde insanların olmadığı zamanlarda ışığın boş yere yanmasını engelleyerek enerji verimliliği ve maliyet tasarrufu sağlar.

Gerçek zamanlı aydınlatma kontrolünün en önemli dezavantajlarından biri de sensör, elektrik ve internetten kaynaklı sorunlardan dolayı haberleşmenin güvensiz ve süreksiz olabilmesidir. Sunucu üzerine düşen bilgilerin kesilmesi veya hatalı gelmesi sistemin kontrolünü zorlaştırarak sistemin kararlılığını etkilemektedir. Bu çalışmada yapılan testler sonucunda prototipin iki saat boyunca çalışmasından sonra elde edilen

veriler Şekil 5.1’ de gösterilmiştir. Bu veriler incelendiğinde aralarda kirli veri geldiği gözlemlenmiştir. Bunun önüne geçebilmek için her üç okuma sonunda veriler kaydedilerek karşılaştırılma yapılmış ve verilerin kontrolü sağlanarak daha kararlı bir sistem oluşturulmuştur.



Şekil 5.1. Prototip’ ten elde edilen veriler

6. SONUÇLAR VE ÖNERİLER

Enerji tüketimi dünya genelinde hızla artarken, bu durum çeşitli çevresel sorunlara neden olmaktadır. Fosil yakıtların yaygın kullanımı, sera gazlarının atmosfere salınmasına ve buna bağlı olarak küresel ısınma sürecine neden olmaktadır. Bu tez çalışması, gereksiz enerji tüketim sorununun nasıl çözülebileceğine dair önemli bulgular sunmaktadır.

Aydınlatma sistemlerinin gerçek zamanlı görüntü işleme teknikleri ve otonom özelliklerle kontrol edilmesi enerji tüketimini optimize ederek israfı engeller. Bu sistem, kullanıcıların SCADA sistemini takip etmelerine ve gerektiğinde müdahale etmelerine olanak tanır. Aydınlatma sistemlerindeki eksiklikler veya arızalar genellikle düzenli olarak izlenemez ve kontrol edilemez. Bu durum, enerji israfına ve gereksiz maliyetlere yol açar. Düzenli bakım süreçlerinin kurulması ve kullanıcıların hızlı bir şekilde arızaları bildirebilmeleri için etkili bir geri bildirim mekanizmasının olması enerji verimliliğini artırmak için önemli bir adımdır. Buna ek olarak, açık hava aydınlatma sistemlerindeki yetersizlikler, suç oranlarının ve güvenlik risklerinin artmasına yol açabilir. Yetersiz aydınlatma koşullarında, güvenlik kameraları etkisiz hale gelebilir ve güvenlik personeli için riskli bir ortam oluşturabilir. Bu bağlamda, aydınlatma sistemlerinin etkili bir şekilde yönetilmesi, suçun azaltılması ve genel güvenliğin artırılması açısından önemlidir.

Bu tez çalışması, şehir içindeki sokaklar, parklar ve şehirlerarası yollar için tasarlanan bir uzaktan aydınlatma sistemi üzerine odaklanmıştır. Gerçek zamanlı görüntü işleme tabanlı SCADA sistemi kullanılarak geliştirilen bu sistem, aydınlatma armatürlerinin etkin bir şekilde izlenmesini, denetlenmesini ve kontrol edilmesini sağlamaktadır. Sistemin doğruluğunu değerlendirmek için bir prototip tasarlanmıştır. Tasarlanan prototipte sistemin izlenilmesini ve kontrolünü sağlayan Arduino Mega 2560 Pro Mini mikrodenetleyici kullanılmıştır. Ayrıca ortamdaki insanların tespiti için kamera ve ışık şiddetini ölçmek için LDR sensörü kullanılmıştır. Bu özellikler, enerji tasarrufunun sağlanması ve aydınlatmanın koşullara göre uyarlanabilmesi için entegre edilmiştir.

Gerçek zamanlı görüntü işleme, sistemin anlık olarak çeşitli parametreleri değerlendirmesine ve gerektiğinde otomatik düzenlemeler yapmasına olanak tanımaktadır. Görüntü işleme tabanlı SCADA sistemi kullanılarak tasarlanan prototip üzerinde gerçekleştirilen testler ve analizler, aydınlatma armatürlerinin uzaktan, hatasız ve verimli bir şekilde çalıştığını göstermiştir.

Önerilerimiz arasında sistemin daha geniş bir ölçekte uygulanması için altyapının güçlendirilmesi ve sistem tarafından tüketilen enerjinin belirlenip optimize edilmesi bulunmaktadır. Ayrıca, kullanıcı dostu bir arayüzün geliştirilmesi ve sistemin güvenlik önlemlerinin daha da güçlendirilmesi önerilmektedir. Gelecekteki çalışmalarda, enerji verimliliği ve çevresel sürdürülebilirlik odaklı geliştirmelerin yapılması da önemli bir alan olacaktır. Bu, uzaktan aydınlatma sistemlerinin daha geniş bir uygulama alanına yayılmasını sağlayarak şehirlerin daha akıllı ve sürdürülebilir hale gelmelerine katkı sağlayabilir.

KAYNAKLAR

3 mm ALICI SP213 LED Tipi. t.y. Geliş tarihi 11 Aralık 2023, gönderen https://www.robotistan.com/3mm-alici-sp213-led-tipi

Abdullah, A., Yusoff, S. H., Zaini, S. A., Midi, N. S., Mohamad, S. Y. 2018. "Smart Street Light Using Intensity Controller". Proceedings of the 2018 7th International Conference on Computer and Communication Engineering, ICCCE 2018, 361-365.

Acar Vural, R., SERT, M. Y., Karaköse, B. 2018. "Gerçek Zamanlı Sürücü Yorgunluk Tespit Sistemi Real Time Driver Fatigue Detection System". Marmara Fen Bilimleri Dergisi, 249-259.

Adafruit TCA9548A 1-to-8 I2C Multiplexer Breakout. 2015. Geliş tarihi 11 Aralık 2023, gönderen https://learn.adafruit.com/adafruit-tca9548a-1-to-8-i2c-multiplexer-breakout/overview

Amazon DynamoDB'nin Özellikleri. t.y. Geliş tarihi 13 Aralık 2023, gönderen https://aws.amazon.com/tr/dynamodb/features/

Arduino Mega2560 Pro Mini Geliştirme Kartı. t.y. Geliş tarihi 11 Aralık 2023, gönderen https://www.direnc.net/mega-2560-pro-atmega2560-gomulu-gelistirme-karti

ASP.NET documentation. 2023. Geliş tarihi 11 Aralık 2023, gönderen https://learn.microsoft.com/en-us/aspnet/core/?view=aspnetcore-6.0

Aydemir, F. 2019. "Internet of Things Based Parking Lot LED Lighting System". European Journal of Science and Technology, (17), 71-76.

Azure Cosmos DB – MS Azure Turkey. 2017. Geliş tarihi 13 Aralık 2023, gönderen http://www.msazureturkey.com/azure-cosmos-db/

Bartlett, Kynn. 2006. "Sams teach yourself CSS in 24 hours", 489.

Berners-Lee, T., Connelly, D. 1995. "Hypertext Markup Language – 2.0". RFC.

Bhairi, M. N., Kangle, S. S., Edake, M. S., Madgundi, B. S., Bhosale, V. B. 2018. "Design and implementation of smart solar LED street light". Içinde *Proceedings - International Conference on Trends in Electronics and Informatics, ICEI 2017* (C. 2018-January, ss. 509-512). Institute of Electrical and Electronics Engineers Inc.

Gencer, Ç., Üstündağ, A., Deli, E. 2020. "PLC Controlles Milk Mizing Machine". Bayburt Üniversitesi Fen Bilimleri Dergisi, *3*(1).

Çağıl, G., Yıldırım, B. 2020. "Detection of an Assembly Part with Deep Learning and Image Processing Bir Montaj Parçasının Derin Öğrenme ve Görüntü İşleme ile Tespiti". Zeki Sistemler Teori ve Uygulamaları Dergisi, 31-37.

Çelik, H., Büyükkınacı, B., Yurtseveni, M. B. 2017. "Aydınlatma Otomasyon Teknikleri: Cendere Caddesi Örneği". Içinde *IX. Ulusal Aydınlatma Sempozyumu*. Geliş tarihi gönderen https://www.emo.org.tr/ekler/a0db794346aa39d_ek.pdf

Çelikpençe, M. 2016. "Farklı Lokasyonlardaki RES ve HES SCADA Sistemlerinden Alınan Verilerin Merkezileştirilerek Ortak Platformdan Raporlanması". Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendislik Dergisi.

de Andrade, R., Hodel, K. N., Justo, J. F., Laganá, A. M., Santos, M. M., Gu, Z. 2018. "Analytical and Experimental Performance Evaluations of CAN-FD Bus". IEEE Access, *6*, 21287-21295.

Eldem, A., Eldem, H., Palalı, A. 2017. "Görüntü İşleme Teknikleriyle Yüz Algılama Sistemi Geliştirme". BEU Journal of Science, 44-48.

Elejoste, P., Angulo, I., Perallos, A., Chertudi, A., Zuazola, I. J. G., Moreno, A., … Villadangos, J. 2013. "An Easy to Deploy Street Light Control System Based on Wireless Communication and LED Technology". Sensors 2013, Vol. 13, Pages 6492-6523, *13*(5), 6492-6523.

Flutter documentation. 2024. Geliş tarihi 11 Aralık 2023, gönderen https://docs.flutter.dev/

Gasperi, M., Hurbain, P. 2010. "Chapter 13: I2C Bus Communication". Extreme NXT. Geliş tarihi gönderen https://books.google.com/books?id=vtUPNDYSTssC

Gökalp, Ö. M. 2021. "Bilgisayar Ağları ve Adli Bilişim".

Gündoğdu, S., Şahin, Ö. 2008. "Su Dağıtım Sistemlerinde Scada Uygulaması". DEÜ Mühendislik Fakültesi Fen ve Mühendislik Dergisi, *10*(3).

Işık, Ş., Vatansever, G., Anagün, Y., Çelikhan, M., Özkan, K. 2020. "Real Time Pedestrian Alert System For Vehicles". Eskişehir Technical University Journal of Science and Technology A-Applied Sciences And Engineering, *2020*(3), 446-453.

I^2C. t.y. Geliş tarihi 11 Aralık 2023, gönderen https://en.wikipedia.org/wiki/I%C2%B2C

İbrahim, M. H. 2022. "SCADA Sistemi: Şehir İçi ve Şehirlerarası Yolların Aydınlatma Sisteminin Kontrolü ve Otomasyonu". Cukurova University Journal of the Faculty of Engineering, *37*(3), 2022.

Karaçor, M., Keleş, K. 2007. "*Otomasyon Sistemlerinin Bileşenleri*".

Karataş, M. Y., Erçetin, R. 2018. "Dizel Jeneratörlerde Scada Uygulamaları". İstanbul Aydın Üniversitesi Dergisi, *4*, 113-126.

Kılıç, M., Özdemir, Ş. 2010. "SCADA Sistemi ile Bir İşletmenin Dış Saha Otomasyonu". AKÜ Fen Bilimleri Dergisi, 59-67.

Kodali, R., Yerroju, S. 2018. "Energy efficient smart street light". Proceedings of the 2017 3rd International Conference on Applied and Theoretical Computing and Communication Technology, iCATccT 2017, 190-193.

Kürşat Çevik, K., Çakır, A. 2010. "Görüntü İşleme Yöntemleriyle Araç Plakalarının Tanınarak Kapı Kontrolünün Gerçekleştirilmesi". AKÜ Fen Bilimleri Dergisi, 31-38.

Semiz, T. Y. 2018. "LDR Nedir? Foto Dirençlerin Çalışma Mantığı". Geliş tarihi 11 Aralık 2023, gönderen https://maker.robotistan.com/ldr/

Mamak, U., Zeki Konyar, M., Solak, S., Hikmet, M., Uçar, B., Üniversitesi, K., … Kocaeli, T. 2020. "Gerçek Zamanlı Yüz Tanıma Tabanlı Personel Kontrol ve Takip Sistemi Tasarımı". European Journal of Science and Technology, (19).

MEGEP. 2007. "*Elektrik Elektronik Teknolojisi SCADA Sistemleri*".

MongoDB. t.y. Geliş tarihi 13 Aralık 2023, gönderen https://tr.wikipedia.org/wiki/MongoDB

MySQL. t.y. Geliş tarihi 11 Aralık 2023, gönderen https://tr.wikipedia.org/wiki/MySQL

MySQL Documentation. 2024. Geliş tarihi 11 Aralık 2023, gönderen https://dev.mysql.com/doc/

Okay, O., Keçecioğlu, F., Gani, A. 2019. "Endüstriyel Bir Doğal Gaz Kompresör İstasyonu İçin Scada Sisteminin Geliştirilmesi". Içinde *UEMK 2019 Proceedings Book 383 24/25 October 2019 Gaziantep University*.

Özseven, T., Sağlam, N. 2021. "IoT Based Street Lighting and Computer Aided Control". Türk0 Doğa ve Fen Dergisi, *10*(2), 265-274.

Perdahçı, C., Ünal, Z., Bıkmaz, B. 2019. "IOT Tabanlı Akıllı Yol Aydınlatma Sistemi".

Piranha 9635 1080P Full HD PC Kamera, Siyah. t.y. Geliş tarihi 11 Aralık 2023, gönderen https://www.amazon.com.tr/Piranha-9635-1080P-USB-Webcam/dp/B08SL4GV2J

PostgreSQL Nedir? Ne İşe Yarar? t.y. Geliş tarihi 13 Aralık 2023, gönderen https://www.pomelosoft.com/blog/postgre-sql-nedir

Remote 8-bit I/O expander for I2C-bus with interrupt. 2013. Semiconductors, NXP.

Scada Nedir ? 2020. Geliş tarihi 17 Ekim 2022, gönderen https://www.ercanotomasyon.com.tr/scada-nedir/

Sorif, S. M., Saha, D., Dutta, P. 2021. "Smart street light management system with automatic brightness adjustment using bolt IoT platform". 2021 IEEE International IOT, Electronics and Mechatronics Conference, IEMTRONICS 2021 - Proceedings.

Taştan, M. 2019. "Akıllı Ev Uygulamaları için Yeni Nesil IoT Denetleyici ile Gerçek Zamanlı Uzaktan İzleme ve Kontrol Uygulaması". Süleyman Demirel University Journal of Natural and Applied Sciences, *23*(2), 481-487.

TCA9548A Low-Voltage 8-Channel I2C Switch with Reset. 2019. Texas Instruments Incorporated.

Universal asynchronous receiver/transmitter (UART). 2006. SCC2691, 14. Geliş tarihi gönderen https://www.nxp.com/docs/en/data-sheet/SCC2691.pdf#page=14

Üçgün, H., Gömbeci, F., Yüzgeç, U., Yalçın, N. 2020. "IoT Tabanlı Platform ile Gerçek Zamanlı İç Ortam Hava Kalitesi İzleme Sistemi". Bilecik Seyh Edebali University Journal of Science, *7*(1), 370-381.

Veritabanı Nedir? t.y. Geliş tarihi 13 Aralık 2023, gönderen https://www.oracle.com/tr/database/what-is-database/

Yanık, H., Turan, B. 2023. "Image processing based distance measurement with Image Meter". Journal of the Faculty of Engineering and Architecture of Gazi University, *38*(2), 1129-1139.

Yılmaz, S., Sungur, C. 2020. "Kamu Binalarında Mevcut Aydınlatma Elemanlarının LED Aydınlatma Elemanlarına Dönüştürülmesi ile Elde Edilecek Elektrik Enerjisi Tasarrufunun Belirlenmesi". European Journal of Science and Technology Special Issue, 214-218.

Yüksek Gerilim Kontrol & Scada & Mimik Panosu. t.y. Geliş tarihi 04 Ekim 2022, gönderen http://www.panel.com.tr/images/Yuksek_Gerilim_Kontrol_Scada_Mimik_Panosu_006.jpg

EKLER

EK-1. HTML Etiketleri

HTML'nin temel bileşenleri, aşağıdaki gibi yaygın olarak kullanılan etiketlerdir

- <html>: Bir HTML belgesini başlatır ve sonlandırır.
- <head>: Sayfanın başlık, meta bilgiler ve stil tanımlamalarını içerir.
- <title>: Sayfanın başlığını belirler, tarayıcı sekmesinde görünen metni tanımlar.
- <body>: Sayfanın görünür içeriğini içerir.
- <h1>, <h2>, <h3>, ... <h6>: Başlık seviyelerini belirler (h1 en yüksek seviye, h6 ise en düşük seviyedir).
- <p>: Paragraf oluşturur.
- <a>: Bağlantı oluşturur ve URL'ye yönlendirir.
- <img>: Resimleri görüntüler.
- : Sırasız liste oluşturur.
- : Sıralı liste oluşturur.
- : Liste öğelerini tanımlar.
- <table>: Tablo oluşturur.
- <tr>: Tablo satırını belirler.
- <td>: Tablo hücresini tanımlar.
- : Blok düzeyinde bir bölme oluşturur.
-
: Bir satır sonlandırma etiketidir. Metin veya öğeleri alt alta düzenlemek için kullanılır.
- <hr>: Bir yatay çizgi ekler ve sayfada bölümleri ayırmak veya içerikleri ayırmak için kullanışlıdır.
- : Metni vurgulamak veya italik yapmak için kullanılır.
- : Metni kalın yapmak veya daha fazla vurgulamak için kullanılır.
- <u>: Metni altı çizili yapmak için kullanılır.
- <a>: Bağlantı oluşturur ve farklı sayfalara veya kaynaklara yönlendirir.
- <img>: Resimleri görüntüler ve src özelliği ile resim dosyasının URL'sini belirtir.
- : Sırasız liste oluşturur ve her liste öğesi <li> içerir.
- : Sıralı liste oluşturur ve her liste öğesi <li> içerir.
- : Liste öğelerini tanımlar ve <ul> veya <ol> etiketi içinde bulunur.

- <table>: Tablo oluşturur ve tablonun başlıkları <th>, satırları <tr>, hücreleri ise <td> veya <th> içerir.
- <th>: Tablo başlığını temsil eder ve genellikle kalın veya ortalanmış metin içerir.
- <tr>: Tablo satırını temsil eder ve hücreleri içerir.
- <td>: Tablo hücresini temsil eder ve metin veya içerik içerir.
- : Blok düzeyinde bir bölme oluşturur ve CSS ile biçimlendirme için kullanışlıdır.
- : Satır içi bir bölme oluşturur ve metni veya öğeleri gruplamak için kullanılır.
- <form>: Form elemanlarını gruplamak için kullanılır ve kullanıcı girdilerini sunar.
- <input>: Form içinde kullanıcıdan veri almak için kullanılır ve türü (örneğin metin kutusu veya düğme) belirtir.
- <textarea>: Çok satırlı metin girişi almak için kullanılır.
- <button>: Bir düğme oluşturur ve kullanıcıya tıklama eylemi sunar.
- <label>: Form elemanları ile ilişkilendirilen metin etiketi oluşturur.
- <select>: Bir liste veya açılır menü oluşturur ve seçenekleri içerir.
- <option>: Seçim listesi içindeki seçenekleri belirtir.
- <iframe>: Başka bir web sayfasını içermek veya göstermek için kullanılır.
- <audio>: Ses dosyalarını çalmak için kullanılır ve ses dosyasının kaynağını belirtir.
- <video>: Video dosyalarını çalmak için kullanılır ve video dosyasının kaynağını belirtir.

(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)

EK-2. CSS Style Özellikleri

- color: Metin rengini belirler.
- font-family: Yazı tipini belirler.
- font-size: Yazı tipi boyutunu belirler.
- font-weight: Yazı tipinin kalınlığını belirler.
- text-align: Metin hizalamasını belirler.

- text-decoration: Metin süslemelerini belirler (altı çizili, çizgili vb.)
- background-color: Arka plan rengini belirler.
- margin: Kenar boşluklarını belirler.
- padding: İçerik ile kenarlar arasındaki boşluğu belirler.
- border: Kenar stili, kalınlığı ve rengini belirler.
- width: Öğenin genişliğini belirler.
- height: Öğenin yüksekliğini belirler.
- display: Öğenin görüntülenme türünü belirler.
- float: Öğenin yüzme davranışını belirler.
- position: Öğenin konumunu belirler.
- z-index: Öğelerin yığındaki sırasını belirler.
- opacity: Öğenin opaklığını belirler. .
- box-shadow: Öğeye bir gölge ekler.
- border-radius: Kenar köşelerinin yuvarlatılmasını belirler.
- text-transform: Metin büyüklüğünü ve küçüklüğünü belirler.
- line-height: Satır yüksekliğini belirler.
- letter-spacing: Harf aralığını belirler.
- text-shadow: Metin üzerine gölge ekler.
- cursor: Fare işaretçisinin görünümünü belirler.
- background-image: Arka plan resmini belirler.
- text-overflow: Metin içeriği çok uzunsa, taşan metni nasıl işleyeceğinizi belirler.
- overflow: İçerik bir öğenin sınırlarından taşıyorsa, bu taşan içeriği nasıl yöneteceğinizi belirler.
- white-space: Metin içeriğinin boşlukları ve satır sonları nasıl işleyeceğini belirler.
- text-transform: Metin içeriğinin büyüklüğünü ve küçüklüğünü belirler.
- box-sizing: Öğelerin içerik kutularının yanı sıra kenarlıklarını ve dolgularını da içerecek şekilde boyutlandırılıp boyutlandırılmayacağını belirler.
- outline: Öğenin odaklandığında veya vurgulandığında etrafında görünen çerçeveyi belirler.
- list-style: Sırasız liste veya sıralı liste öğelerinin işaretlerini ve stilini belirler.
- text-align: Metin hizalamasını belirler.

- vertical-align: Öğelerin dikey hizalamasını belirler. Özellikle, metin veya görsel öğeleri ortalamak için kullanışlıdır.
- position: Öğelerin konumunu ve yerleşimini belirler.
- transition: CSS geçişleri tanımlar ve öğelerin belirli bir özelliği (örneğin, renk veya boyut) değiştirildiğinde nasıl geçiş yapacaklarını belirler.
- animation: CSS animasyonlarını tanımlar ve öğelerin belirli bir hareket veya değişim sürecini gösterir.
- transform: Öğeleri döndürme, ölçeklendirme veya kaydırma gibi dönüşümleri tanımlar.
- filter: Öğelerin görsel filtrelerini belirler.

- user-select: Kullanıcı seçimini kontrol eder ve metin seçilemez veya metin seçimi engellenir.
- box-decoration-break: Öğelerin içerik kutularının nasıl kesileceğini belirler.
- word-wrap: Uzun kelimeler veya URL'ler ile başa çıkma stratejisini belirler.

(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)(Berners-Lee ve Connelly, 1995)

EK-3.Ardunio Setup ve Loop fonksiyonları

```
void setup()
{
   while (!Serial);
   delay(1000);

   Wire.begin();

   Serial.begin(9600);

   //Serial.println("\nTCA9548 Multiple PCF8574 LED Control ready!");

   // PARK OUTPUT CHANNEL-------------------------------------
   tcaselect(CHANNEL_2);

   if (!pcf1.begin(0x20, &Wire)) {
     Serial.println("Couldn't find PCF8574-1");
     //while (1);
   }
   if (!pcf2.begin(0x21, &Wire)) {
     Serial.println("Couldn't find PCF8574-2");
```

```
  //while (1);
}
// PARK INPUT CHANNEL-------------------------------------
tcaselect(CHANNEL_3);

if (!pcf3.begin(0x22, &Wire)) {
  Serial.println("Couldn't find PCF8574-3");
  //while (1);
}
if (!pcf4.begin(0x23, &Wire)) {
  Serial.println("Couldn't find PCF8574-4");
  //while (1);
}


// SOKAK OUTPUT CHANNEL-----------------------------------
tcaselect(CHANNEL_4);

if (!pcf5.begin(0x20, &Wire)) {
  Serial.println("Couldn't find PCF8574-5");
  //while (1);
}
if (!pcf6.begin(0x21, &Wire)) {
  Serial.println("Couldn't find PCF8574-6");
  //while (1);
}
if (!pcf7.begin(0x22, &Wire)) {
  Serial.println("Couldn't find PCF8574-7");
  //while (1);
}
// SOKAK INPUT CHANNEL------------------------------------
tcaselect(CHANNEL_5);

if (!pcf8.begin(0x23, &Wire)) {
  Serial.println("Couldn't find PCF8574-8");
  //while (1);
}
if (!pcf9.begin(0x24, &Wire)) {
  Serial.println("Couldn't find PCF8574-9");
  //while (1);
}
if (!pcf10.begin(0x25, &Wire)) {
  Serial.println("Couldn't find PCF8574-10");
  //while (1);
}

// OTOYOL OUTPUT CHANNEL-----------------------------------
tcaselect(CHANNEL_6);

if (!pcf11.begin(0x20, &Wire)) {
```

```
    Serial.println("Couldn't find PCF8574-11");
    //while (1);
  }
  if (!pcf12.begin(0x21, &Wire)) {
    Serial.println("Couldn't find PCF8574-12");
    //while (1);
  }
  if (!pcf13.begin(0x22, &Wire)) {
    Serial.println("Couldn't find PCF8574-13");
    //while (1);
  }

  // OTOYOL INPUT CHANNEL-------------------------------------
  tcaselect(CHANNEL_7);

  if (!pcf14.begin(0x23, &Wire)) {
    Serial.println("Couldn't find PCF8574-14");
    //while (1);
  }
  if (!pcf15.begin(0x24, &Wire)) {
    Serial.println("Couldn't find PCF8574-15");
    //while (1);
  }
  if (!pcf16.begin(0x25, &Wire)) {
    Serial.println("Couldn't find PCF8574-16");
    //while (1);
  }
  //Serial.println("\nALL Channel Active!");
  // PARK OUTPUT CHANNEL
  tcaselect(CHANNEL_2);
  pcf1.pinMode(0, OUTPUT);pcf1.pinMode(1, OUTPUT);pcf1.pinMode(2,
OUTPUT);pcf1.pinMode(3, OUTPUT);
  pcf1.pinMode(4, OUTPUT);pcf1.pinMode(5, OUTPUT);pcf1.pinMode(6,
OUTPUT);pcf1.pinMode(7, OUTPUT);
  pcf2.pinMode(0, OUTPUT);pcf2.pinMode(1, OUTPUT);pcf2.pinMode(2,
OUTPUT);pcf2.pinMode(3, OUTPUT);
  pcf2.pinMode(4, OUTPUT);pcf2.pinMode(5, OUTPUT);pcf2.pinMode(6,
OUTPUT);pcf2.pinMode(7, OUTPUT);
  delay(100);
  // PARK INPUT CHANNEL
  tcaselect(CHANNEL_3);
  pcf3.pinMode(0, INPUT);pcf3.pinMode(1, INPUT);pcf3.pinMode(2,
INPUT);pcf3.pinMode(3, INPUT);
  pcf3.pinMode(4, INPUT);pcf3.pinMode(5, INPUT);pcf3.pinMode(6,
INPUT);pcf3.pinMode(7, INPUT);
  pcf4.pinMode(0, INPUT);pcf4.pinMode(1, INPUT);pcf4.pinMode(2,
INPUT);pcf4.pinMode(3, INPUT);
  pcf4.pinMode(4, INPUT);pcf4.pinMode(5, INPUT);pcf4.pinMode(6,
INPUT);pcf4.pinMode(7, INPUT);
  delay(100);
```

```
  // SOKAK  OUTPUT CHANNEL
  tcaselect(CHANNEL_4);
  pcf5.pinMode(0, OUTPUT);pcf5.pinMode(1, OUTPUT);pcf5.pinMode(2,
OUTPUT);pcf5.pinMode(3, OUTPUT);
  pcf5.pinMode(4, OUTPUT);pcf5.pinMode(5, OUTPUT);pcf5.pinMode(6,
OUTPUT);pcf5.pinMode(7, OUTPUT);
  pcf6.pinMode(0, OUTPUT);pcf6.pinMode(1, OUTPUT);pcf6.pinMode(2,
OUTPUT);pcf6.pinMode(3, OUTPUT);
  pcf6.pinMode(4, OUTPUT);pcf6.pinMode(5, OUTPUT);pcf6.pinMode(6,
OUTPUT);pcf6.pinMode(7, OUTPUT);
  pcf7.pinMode(0, OUTPUT);pcf7.pinMode(1, OUTPUT);pcf7.pinMode(2,
OUTPUT);pcf7.pinMode(3, OUTPUT);
  pcf7.pinMode(4, OUTPUT);pcf7.pinMode(5, OUTPUT);pcf7.pinMode(6,
OUTPUT);pcf7.pinMode(7, OUTPUT);
  delay(100);
  // SOKAK INPUT CHANNEL
  tcaselect(CHANNEL_5);
  pcf8.pinMode(0, INPUT);pcf8.pinMode(1, INPUT);pcf8.pinMode(2,
INPUT);pcf8.pinMode(3, INPUT);
  pcf8.pinMode(4, INPUT);pcf8.pinMode(5, INPUT);pcf8.pinMode(6,
INPUT);pcf8.pinMode(7, INPUT);
  pcf9.pinMode(0, INPUT);pcf9.pinMode(1, INPUT);pcf9.pinMode(2,
INPUT);pcf9.pinMode(3, INPUT);
  pcf9.pinMode(4, INPUT);pcf9.pinMode(5, INPUT);pcf9.pinMode(6,
INPUT);pcf9.pinMode(7, INPUT);
  pcf10.pinMode(0, INPUT);pcf10.pinMode(1, INPUT);pcf10.pinMode(2,
INPUT);pcf10.pinMode(3, INPUT);
  pcf10.pinMode(4, INPUT);pcf10.pinMode(5, INPUT);pcf10.pinMode(6,
INPUT);pcf10.pinMode(7, INPUT);
  delay(100);
  //OTOYOL OUTPUT CHANNEL
  tcaselect(CHANNEL_6);
  pcf11.pinMode(0, OUTPUT);pcf11.pinMode(1, OUTPUT);pcf11.pinMode(2,
OUTPUT);pcf11.pinMode(3, OUTPUT);
  pcf11.pinMode(4, OUTPUT);pcf11.pinMode(5, OUTPUT);pcf11.pinMode(6,
OUTPUT);pcf11.pinMode(7, OUTPUT);
  pcf12.pinMode(0, OUTPUT);pcf12.pinMode(1, OUTPUT);pcf12.pinMode(2,
OUTPUT);pcf12.pinMode(3, OUTPUT);
  pcf12.pinMode(4, OUTPUT);pcf12.pinMode(5, OUTPUT);pcf12.pinMode(6,
OUTPUT);pcf12.pinMode(7, OUTPUT);
  pcf13.pinMode(0,    OUTPUT);pcf13.pinMode(1,    OUTPUT);pcf13.pinMode(2,
OUTPUT);pcf13.pinMode(3, OUTPUT);
  pcf13.pinMode(4,    OUTPUT);pcf13.pinMode(5,    OUTPUT);pcf13.pinMode(6,
OUTPUT);pcf13.pinMode(7, OUTPUT);
  delay(100);
  // OTOYOL INPUT CHANNEL
  tcaselect(CHANNEL_7);
  pcf14.pinMode(0,    INPUT);pcf14.pinMode(1,    INPUT);pcf14.pinMode(2,
INPUT);pcf14.pinMode(3, INPUT);
```

```
   pcf14.pinMode(4, INPUT);pcf14.pinMode(5, INPUT);pcf14.pinMode(6,
INPUT);pcf14.pinMode(7, INPUT);
   pcf15.pinMode(0, INPUT);pcf15.pinMode(1, INPUT);pcf15.pinMode(2,
INPUT);pcf16.pinMode(3, INPUT);
   pcf15.pinMode(4, INPUT);pcf15.pinMode(5, INPUT);pcf15.pinMode(6,
INPUT);pcf16.pinMode(7, INPUT);
   pcf16.pinMode(0, INPUT);pcf16.pinMode(1, INPUT);pcf16.pinMode(2,
INPUT);pcf16.pinMode(3, INPUT);
   pcf16.pinMode(4, INPUT);pcf16.pinMode(5, INPUT);pcf16.pinMode(6,
INPUT);pcf16.pinMode(7, INPUT);
   for (int i = 0; i < 64; i++) {
     pinsState_inputs[i] = false;
   }
   bool pinStates[64];
   for (int i = 0; i < 64; i++) {
     pinStates[i] = false;
   }
   outputAllPins(pinStates);
}

void loop()
{

  if (Serial.available()) {
   String input = Serial.readStringUntil('\n');
   char dataChars[MAX_DATA_LENGTH];
   input.toCharArray(dataChars, MAX_DATA_LENGTH);
   updatePinStates(dataChars);
   outputAllPins(pinStates);
   delay(1000);
   readAllPins();
   printAllPinsState();
  }
}
```

EK-4.Ardunio Setup ve Loop fonksiyonları

```
while saat > "02"  and saat < "05":
        print("saat 22 ile 24 arası")
        ret, frame = cap.read()
        if not ret:
            break
        frame = cv2.resize(frame, desired_size)
        img_tensor = torchvision.transforms.functional.to_tensor(frame)
        img_tensor = img_tensor.unsqueeze(0)
        with torch.no_grad():
            predictions = model(img_tensor)
        boxes = predictions[0]['boxes']
        labels = predictions[0]['labels']
```

```
        scores = predictions[0]['scores']

        human_boxes = boxes[labels == 1]
        human_scores = scores[labels == 1]

        for i in range(human_boxes.shape[0]):
            box = human_boxes[i].detach().cpu().numpy()
            score = human_scores[i].detach().cpu().numpy()
            if score > 0.5:
                cv2.rectangle(frame, (int(box[0]), int(box[1])), (int(box[2]), int(box[3])), (255,
0, 0), 2)
                print(score)
                response = requests.get(url_getArmatur, verify=False)  # SSL doğrulamayı
devre dışı bırak
                if response.status_code == 200:
                    data = response.json()  # JSON yanıtını çözümleme
                    # print(data)
                    for x in range(len(data)):
                        if data[x]["armatur_ariza_durum"] == "CALISIYOR":
                            send_url = url_PostArmaturByIdAcma + "?id=" + f"{x + 1}"
                            response = requests.post(send_url, verify=False)
                            if response.status_code == 200:
                                # data = response.json()
                                print("Yanıt verisi:", data)
                            else:
                                print("Hata kodu:", response.status_code)
                                print("Hata mesajı:", response.text)
                else:
                    print("error")
            else:
                print("insan yok kameralar kapalı")
                response = requests.get(url_getArmatur, verify=False)  # SSL doğrulamayı
devre dışı bırak
                if response.status_code == 200:
                    data = response.json()  # JSON yanıtını çözümleme
                    # print(data)
                    for x in range(len(data)):
                        if data[x]["armatur_ariza_durum"] == "CALISIYOR":
                            send_url = url_PostArmaturByIdKapatma + "?id=" + f"{x + 1}"
                            response = requests.post(send_url, verify=False)
                            if response.status_code == 200:
                                # data = response.json()
                                print("Yanıt verisi:", data)
                            else:
                                print("Hata kodu:", response.status_code)
                                print("Hata mesajı:", response.text)
                else:
                    print("error")

        cv2.imshow('Cars and Humans Detected', frame)
```

```
        if cv2.waitKey(1) == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()

            if ldr_value < '250':
                print('ldr degeri 250\'den küçük')
                # aşağıdaki for da ldr değeri 250'den küçük olduğu için çalışan bütün ledleri
sunucuda ON yapıyoruz
                for x in range(len(data)):
                    if data[x]["armatur_ariza_durum"] == "CALISIYOR":
                        send_url = url_PostArmaturByIdAcma + "?id=" + f"{x + 1}"
                        response = requests.post(send_url, verify=False)
                        data = response.json()
                        if response.status_code == 200:
                            # data = response.json()
                            print("Yanıt verisi:", data)
                        else:
                            print("Hata kodu:", response.status_code)
                print("Hata mesajı:", response.text)
```