

180207047_Azime_Kara

Yazar Azime Kara

Gönderim Tarihi: 11-Oca-2024 03:21PM (UTC+0300)

Gönderim Numarası: 2269306603

Dosya adı: 180207047_Azime_Kara.docx (1.36M)

Kelime sayısı: 5526

Karakter sayısı: 37193

KOCAELİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ

BÖLÜMÜ

BITİRME ÇALIŞMASI



AKILLI ÇİFTLİK UYGULAMASI

180207047

Azime KARA

Danışman: Doç. Dr. Oğuzhan KARAHAN

KOCAELİ, 2024

ÖNSÖZ ve TEŞEKKÜR

Bu tez, web temelli arayüz aracılığıyla çiftliğin izlenmesi ve kontrol edilmesini sağlayan IOT tabanlı bir sistem üzerine odaklanmaktadır. Günümüzde, teknolojinin hızlı gelişimiyle birlikte akıllı çiftlik sistemleri südürebilir bir geleceğe doğru önemli bir adım atılmasını sağlamaktadır. Akıllı çiftlikler, veri odaklı karar verme süreçleriyle daha az kaynak kullanılarak daha fazla üretim yapma imkanı sunuyor. İklim değişiklileri ve doğal kaynakların azalması gibi çağımızın en büyük sorunlarına çözüm sunma potansiyeline sahiptir. Bu çalışmada IOT teknolojisi kullanılarak çiftliklerin gerçek zamanlı olarak izlenmesi, hayvan kontrol ve güvenliğinin sağlanması ve herhangi bir olumsuz durumda önlem alınması amaçlanmaktadır. Ayrıca, web tabanlı arayüz sayesinde kullanıcıların kolaylıkla erişebileceği bir kullanıcı deneyimi de hedeflenmektedir.

Bu tezin hazırlanması sürecinde bana rehberlik eden, bilgi ve tecrübelerini esirgemeyen danışman hocam Doç. Dr. Oğuzhan Karahan'a en içten teşekkürlerimi sunuyorum. Değerli yönlendirmeleri ve destekleri sayesinde bu tezi tamamlamak benim için mümkün oldu. Ayrıca, bana her zaman destek olan ve motivasyonumu yüksek tutan aileme ve arkadaşlarımı da teşekkür etmek isterim. Bu tezin tamamlanmasında emeği geçen herkese minnettarım.

Ocak 2024, KOCAELİ
Azime KARA

İÇİNDEKİLER

ÖNSÖZ ve TEŞEKKÜR	1
Özet	Error! Bookmark not defined.
Abstract	Error! Bookmark not defined.
1. LİTERATÜR ARAŞTIRMASI	8
1.1 IoT Sistemler.....	8
1.2 IoT Tabanlı Akıllı Çiftlik Uygulamaları	8
2. KULLANILAN TEKNOLOJİLER	10
2.1 Yazılım	10
2.1.1 C	10
2.1.2 C#	10
2.1.3 Javascript.....	11
2.1.4 API	13
2.1.5 MSSQL.....	13
2.2 Donanım	13
2.2.1 STM32F407	13
2.2.2 IRF520 Mosfet	14
2.2.3 Touch Sensör	14
2.2.4 Yağmur Sensörü	15
2.2.5 Gaz Sensörü.....	15
2.2.6 Buzzer.....	16
2.2.7 Step Motor.....	16
2.2.8 Servo Motor.....	17
2.2.9 Mesafe Sensörü	17
3. IOT HABERLEŞMESİ VE GÖMÜLÜ SİSTEMLER	18
3.1 IoT Haberleşme Türleri	18
3.2 IoT Haberleşmesi Kapsam Alanları ve Boyutları	19
3.3 Haberleşme Topolojileri	20
3.4 IoT Alanında Gömülü Sistemler	21
4. AKILLI ÇİFTLİK SİSTEMİNİN GERÇEKLEŞTİRİLMESİ	22
4.1 Giriş	22
4.2 Sistemin Donanımsal Tasarımı	22
4.3 Sistemin Yazılımsal Tasarımı	23
4.4 STM32F407 Kartının Kodlanması	23
4.5 ESP32 ile Veri İletimi	24
4.6 Web Arayüzünün Oluşturulması	25
4.6.1 Anasayfa	25
4.6.2 Yağmur Kontrol Paneli	26
4.6.3 Beslenme Sistemi Kontrol Paneli.....	27
4.6.4 Yangın Kontrol Paneli.....	27
4.6.5 Kapı Kontrol Paneli	28
4.6.6 Veri Tabanı Tasarımı	28
4.7 Deneyel Sonuçlar	29
5. SONUÇLAR	30
6. KAYNAKLAR	30

ŞEKİLLER DİZİNİ

Şekil 2.1: STM32F407	13
Şekil 2.2: IRF520 MOSFET	14
Şekil 2.3: Touch Sensörü	14
Şekil 2.4: Yağmur Sensörü.....	15
Şekil 2.5: Gaz Sensörü	15
Şekil 2.6: Buzzer	16
Şekil 2.7: Step Motor	16
Şekil 2.8: Servo Motor	17
Şekil 2.9: Mesafe Sensörü	17
Şekil 3.1: Örnek IOT Sistemi	18
Şekil 3.2: Haberleşme Protokolleri	19
Şekil 4.1: Elektronik Test Devresi	23
Şekil 4.2: Sensör Verileri Çekme Kodu	24
Şekil 4.3: STM UART Veri Gönderme Kodu.....	24
Şekil 4.4: STM UART Veri Alma Kodu.....	24
Şekil 4.5: ESP UART Veri Gönderme Kodu	25
Şekil 4.6: ESP API Veri Gönderme Kodu	25
Şekil 4.7: ESP API Veri Alma Kodu	25
Şekil 4.8: Hoş geldin Ekranı	26
Şekil 4.9: Yağmur Kontrol Paneli	26
Şekil 4.10: Besleme Sistemi Kontrol Paneli	27
Şekil 4.11: Yangın Kontrol Paneli	27
Şekil 4.12: Kapı Kontrol Paneli	28
Şekil 4.13: Veri Tabanı	28
Şekil 4.14: Akıllı Çiftlik Modellemesi	29
Şekil 5.1: Yangın Alarmı	29

SİMGELER DİZİNİ VE KISALTMALAR

8

MySQL: My Structured Query Language
API: Application Programming Interface
HTTP: Hyper Text Transfer Protocol
IOT: Internet of Things
UART: Universal Asyncchronous Receiver Transmitter
HTML: HyperText Markup Language
XHTML: Extensible HyperText Markup Language
CSS3: Cascading Style Sheets
RSS: Really Simple Syndication
AJAX: Asynchronous JavaScript and XML
XML: Extensible Markup Language
SOA: Service Oriented Architecture
SRS: Supplemental Restraint System
UI: User Interface

Akıllı Çiftlik Uygulaması

Azime Kara

Anahtar Kelimeler: IOT, JavaScript, Embedded System , UI, API

Özet:

Bu proje, IOT teknolojisi kullanılarak oluşturulan bir smart farm sistemidir. Bu sistem, çiftliğin web tabanlı bir arayüz aracılığıyla gerçek zamanlı olarak izlenmesi ve kontrol edilmesini sağlamaktadır.

Projenin frontend kısmı Neighbor Blazor kullanılarak geliştirilmiştir ve kullanıcıların anlık olarak çiftlikteki hayvanların takibini, yangın takibini, yem takibini ve çati takibini yapabileceği bir sistem oluşturulmuştur. Backend tarafı .NET ile C# kullanılarak yazılmıştır. Embedded tarafı ise C ile yazılmış olup gaz sensörü, touch sensörü ve mesafe sensöründen veri almıştır. Alınan veriler doğrultusunda servo motor, step motor ve fan çalıştırılmıştır. Poje sayesinde çiftlik sahiplerinin IOT tabanlı takip sistemi oluşturulmuş olup sitemi herhangi bir yerden izleme ve yönetme imkanı sağlanmıştır.

Smart Farm Application

Azime Kara

Keywords: IOT, JavaScript, Embedded System ,UI, API

Abstract:

This project is a smart farm system created using IOT technology. This system provides real-time monitoring and control of the farm through a web-based interface.

The frontend part of the project was developed using Neighbor Blazor and a system was created where users can instantly track animals on the farm, fire tracking, feed tracking and roof tracking. The backend side is written using .NET and C#. The embedded side is written in C and data is taken from the gas sensor, touch sensor, rain sensor and distance sensor. Servo motor, stepper motor and fan were operated in line with the data received. Thanks to the project, an IoT-based tracking system was created for farm owners, allowing them to monitor and manage the system from anywhere.

1. LİTERATÜR ARAŞTIRMASI

Akıllı çiftlik sistemleri ve Nesnelerin İnterneti (IoT) teknolojileri, tarım sektöründe ciddi bir dönüşüm yaratmaktadır. Bu yenilikler, çiftlik yönetimini daha verimli, sürdürülebilir ve otomatize edilmiş bir hale getirerek, geleneksel tarım yöntemlerini modernize ediyor. Bu teknolojilerin temel odak noktaları, hassas tarım, tarım dronları, hayvancılık izleme ve akıllı seralar gibi alanlardır. Hassas tarım, çiftçilere toprak özelliklerini ve çevresel faktörleri sürekli olarak izleme imkanı sunarak tarımsal faaliyetleri daha etkili ve kontrollü bir şekilde yönetme fırsatı verir. Bu alandaki teknolojik gelişmeler, sensörler, GPS ve veri analizi gibi araçları içerir. Örneğin, Çin'de IoT tabanlı tarımsal mekanizasyonun kullanılması, tarım makinalarının daha etkin kullanımını, işgücü verimliliğinin artmasını ve tarımsal üretim maliyetlerinin düşmesini sağlamıştır. Tarım dronları, İnsansız Hava Araçları (İHA) olarak da bilinir ve tarımsal alanların izlenmesinde kullanılır [1]. Bu dronlar, dijital görüntüleme sistemleri ve çeşitli algılayıcılarla donatılmış olup, toplanan verilerin analiz edilmesiyle çiftçilere verimlilik artışı ve risk yönetiminde yardımcı olur. Akıllı çiftliklerde hayvanların izlenmesi, sağlık durumlarının takibi ve davranış analizi gibi işlevleri yerine getiren sistemler bulunmaktadır. Bu sistemler, hayvanların beslenme alışkanlıklarından sağlık durumlarına kadar pek çok faktörü sürekli olarak izleyebilir. Avrupa'da bu tür akıllı çiftlik sistemleriyle donatılmış yaklaşık 20 çiftlik bulunmaktadır. Bu literatür incelemesi, IoT teknolojilerinin tarım sektörüne getirdiği yenilikleri ve bu teknolojilerin çiftlik yönetiminde nasıl bir dönüşüm yarattığını ortaya koyuyor. Akıllı çiftlik sistemleri, tarımın geleceğini şekillendiren önemli bir rol oynuyor ve bu alandaki uygulamalar sürekli olarak gelişmektedir.

1.1 IoT Sistemler

IoT (Nesnelerin İnterneti), çevremizdeki cihazları ve nesneleri internet üzerinden birbirine bağlayan ileri teknoloji ağının bir parçasıdır. Bu ağ, sensörler, yazılımlar ve diğer teknolojik araçlar kullanarak veri toplar ve bu verileri işler. IoT sistemlerinin temel amacı, toplanan bu verileri analiz ederek daha akıllı, verimli ve otomatik iş süreçleri oluşturmaktır. Ömek olarak, evlerdeki ısıtma ve aydınlatma sistemlerinin otomatik kontrolünden, akıllı tarım uygulamalarına, hatta kentsel altyapı yönetimine

kadar geniş bir kullanım alanı sunar. Bu sistemler, gerçek zamanlı veri akışı sağlayarak, daha dinamik ve etkileşimli bir ortam oluşturur. IoT, gelişmiş veri analizi ve makine öğrenimi teknolojileri ile birleştiğinde, olağanüstü derecede akıllı çözümler sunar. Örneğin, bir fabrika ortamında, IoT cihazları makinelerin performansını izleyerek arızaları önceden tahmin edebilir ve böylece bakım süreçlerini optimize eder. Sağlık sektöründe ise, IoT cihazları hastaların sağlık durumlarını sürekli izleyerek, anomal durumları arasında tespit edip uyarı verebilir. Bu teknoloji, enerji yönetimi, trafik kontrolü, çevresel izleme ve daha birçok alanda da etkili çözümler sağlar [2]. IoT'un sunduğu bu entegre ve otomatik sistemler, gerek bireysel gerekse kurumsal düzeyde yaşam kalitesini ve iş verimliliğini önemli ölçüde artırma potansiyeline sahiptir. Gelecekte, IoT'un daha da gelişerek hayatımızın her alanında vazgeçilmez bir rol oynaması beklenmektedir.

1.2 IoT Tabanlı Akıllı Çiftlik Sistemleri

IoT (Nesnelerin İnterneti) Tabanlı Akıllı Çiftlik Sistemleri, tarım sektöründe teknolojinin getirdiği devrimsel değişimlerin en önemli örneklerinden biridir. Bu sistemler, çiftlik işletmeciliğini dijitalleştirerek, verimliliği artırır, kaynak kullanımını optimize eder ve daha sürdürülebilir tarım uygulamalarına olanak sağlar. IoT tabanlı sistemler, çeşitli sensörler ve cihazlar aracılığıyla çiftlikteki hayvanların sağlık durumunu, bitki büyümemesini, toprak nemini ve diğer önemli çevresel faktörleri sürekli olarak izler. Bu veriler, merkezi bir veri işleme platformunda toplanır ve analiz edilir [3]. Akıllı çiftlik sistemlerinde kullanılan IoT cihazları, çiftlik yöneticilerine gerçek zamanlı veri sağlar. Bu veriler sayesinde çiftlik yöneticileri, su ve gübre gibi kaynakların kullanımını, hayvanların sağlık durumunu ve bitkilerin büyümeye koşullarını daha iyi yönetebilirler. Örneğin, toprak nem sensörleri, sulama sistemlerinin ne zaman ve ne kadar süreyle çalıştırılması gereğine dair hassas bilgiler sunar. Benzer şekilde, hayvan hareketleri ve yem tüketimi üzerine kurulu sensörler, hayvan sağlığını izlemek ve verimli beslenme programları oluşturmak için kullanılabilir. Ayrıca, IoT tabanlı sistemler, çiftlik operasyonlarını uzaktan izleme ve kontrol etme imkanı sunar. Çiftlik sahipleri ve yöneticileri, mobil cihazlar veya bilgisayarlar üzerinden çiftliklerini izleyebilir, gereğinde müdahale edebilir. Bu, özellikle geniş çiftliklerde veya birden fazla çiftlik işleten kişiler için büyük bir kolaylık sağlar. Akıllı çiftlik sistemlerinin bir başka önemli yönü ise veri toplama ve analiz kabiliyetidir. Toplanan veriler, makine

öğrenimi ve yapay zeka algoritmaları kullanılarak analiz edilebilir. Bu analizler sayesinde, ürün verimi, hastalık riski ve diğer önemli faktörler hakkında öngörülerde bulunulabilir. Böylece, çiftlik yöneticileri olası sorunları önceden tespit edebilir ve gerekli önlemleri alabilirler. Genel olarak, IoT Tabanlı Akıllı Çiftlik Sistemleri, tarım sektöründe verimliliği, sürdürülebilirliği ve karlılığı artıran yenilikçi bir yaklaşım sunmaktadır. Bu sistemler, geleneksel tarım yöntemlerini teknolojiyle birleştirerek, daha verimli, çevreye duyarlı ve ekonomik çiftlik işletmeciliği yapılmasına imkan tanır.

2. KULLANILAN TEKNOLOJİLER

2.1 Yazılım

2.1.1 C

C programlama dili, 1972'de Ken Thompson ve Dennis Ritchie tarafından AT&T Bell Laboratuvarları'nda geliştirilmiştir.¹¹ Bu dil, UNIX İşletim Sistemi'nin geliştirilmesi için Bell Labs tarafından tasarlanmış ve B dilinden evrilmiştir. Brian Kernighan ve Dennis M. Ritchie'nin "C Programlama Dili" adlı kitabının yayımılanması, C dilinin popülerliğini artırmıştır. Sistem ve sürücü yazılımları, işletim sistemi modülleri gibi yüksek performans gerektiren alanlarda sıkılıkla kullanılır. C, programcılara geniş özgürlükler sunarken, bu özgürlüklerin yanlış kullanılması ciddi hatalara yol açabilir. Zamanla, birçok C programcısı, nesne yönelimli programlama özellikleri sunan C++ diline geçiş yapmıştır [4]. C programlamada her program main() fonksiyonu ile başlar ve bu fonksiyon zorunludur. printf() fonksiyonu, formatlanmış metni ekrana yazdırma için kullanılır ve return ifadesi, main() fonksiyonundaki programın sonlanmasını sağlar. Programınızda stdio.h gibi başlık dosyalarını kullanmanız gerekmektedir. C dilinde büyük/küçük harf ayrimı önemlidir ve ifadeler genellikle noktalı virgülle sonlandırılır. Etkili bir öğrenme süreci için pratik yapmak, hata ayıklama araçlarını öğrenmek ve C programlama topluluklarına katılmak önemlidir [5] .

2.1.2 C#

C# olarak bilinen ve Microsoft tarafından geliştirilen bu programlama dili, nesneye yönelik ve bileşen tabanlı özelliklere sahiptir. 2000 yılında Anders Hejlsberg liderliğindeki bir ekip tarafından tasarılmaya başlanan C#, 2002 yılında ilk sürümü olan C# 1.0 ile piyasaya sürüldü ve sürekli olarak güncellenmektedir. Bu dil, modern sistemlerle uyumluluğu ve ileri düzey özellikleri sayesinde yazılım geliştiriciler arasında yüksek popülerlik kazanmıştır [6].

C# programlarının kodları, doğrudan makine koduna değil, ara bir form olan IL

koduna dönüştürülür. Bu, farklı işletim sistemlerinde C# programlarının çalışmasını kolaylaştırır. Bu kodların derlenmesi genellikle Visual Studio gibi gelişmiş editörlerle yapılır, ancak basit denemeler için Not Defteri ve .NET Framework içindeki csc.exe derleyicisi yeterlidir. C# içerisinde iki ana tür bulunmaktadır: değer ve başvuru türleri. Değer türleri doğrudan verilerini içerirken, başvuru türleri verilere yapılan referansları barındırır. C#'da basit türler (int, long, bool gibi), enum türleri, yapı türleri ve daha fazlası bulunur [7].

C# olarak bilinen ve Microsoft tarafından geliştirilen bu programlama dili, nesneye yönelik ve bileşen tabanlı özelliklere sahiptir. 2000 yılında Anders Hejlsberg liderliğindeki bir ekip tarafından tasarılmaya başlanan C#, 2002 yılında ilk sürümü olan C# 1.0 ile piyasaya sürüldü ve sürekli olarak güncellenmektedir. Bu dil, modern sistemlerle uyumluluğu ve ileri düzey özellikleri sayesinde yazılım geliştiriciler arasında yüksek popülerlik kazanmıştır. C# programlarının kodları, doğrudan makine koduna değil, ara bir form olan IL koduna dönüştürülür. Bu, farklı işletim sistemlerinde C# programlarının çalışmasını kolaylaştırır. Bu kodların derlenmesi genellikle Visual Studio gibi gelişmiş editörlerle yapılır, ancak basit denemeler için Not Defteri ve .NET Framework içindeki csc.exe derleyicisi yeterlidir. C# içerisinde iki ana tür bulunmaktadır: değer ve başvuru türleri. Değer türleri doğrudan verilerini içerirken, başvuru türleri verilere yapılan referansları barındırır. C#'da basit türler (int, long, bool gibi), enum türleri, yapı türleri ve daha fazlası bulunur. C# ile gerçekleştirilebilecek projeler oldukça çeşitlidir. Web ve mobil uygulamalar, oyunlar, konsol uygulamaları, DLL dosyaları ve Windows Form Uygulamaları bu dil ile geliştirilebilir. Özellikle ASP.NET ile web uygulamaları ve Xamarin platformu kullanılarak iOS, Android ve Windows için mobil uygulamalar geliştirme oldukça yaygındır. Ayrıca Unity gibi oyun motorları da C# ile uyumlu çalışır. C# dili, nesne tabanlı programlama prensiplerini temel alır ve metodlar, sınıflar, nesneler gibi kavramları içerir. Bu metodlar, programın düzenli ve okunabilir olmasını sağlar. Sınıflar, nesneler için tasarım şablonları işlevi görürken, nesneler ise bu sınıflardan türetilen örneklerdir. C# öğrenmek görece kolaydır ve dilin esnek, açık kaynaklı yapısı mevcuttur. Bu dil ile yüksek performanslı uygulamaların hızlı bir şekilde geliştirilmesi mümkündür. C# öğrenme süreci, önceki programlama tecrübelerinize bağlı olarak birkaç ay sürebilir. C# geliştiricilere yönelik talebin artması, bu dili öğrenmenin kariyer için iyi bir fırsat olduğunu göstermektedir [8].

2.1.3 Javascript

JavaScript, web sayfalarına dinamik özellikler ve interaktiflik eklemek için kullanılan bir betik dilidir. HTML ve CSS ile birlikte çalışarak, kullanıcıların deneyimini zenginleştiren web uygulamaları oluşturulmasını sağlar. JavaScript, HTML içeriğini yapılandırmak ve anlam kazandırmak için kullanılırken, CSS ile içeriğe stil vermek için birlikte çalışır. Bu etkileşim, web sayfalarının daha görsel ve kullanıcı odaklı hale gelmesine olanak tanır. JavaScript'in temel sözdizimi, değişkenler, veri türleri, operatörler ve kontrol yapıları gibi programlama yapılarını içerir. Bu yapılar, JavaScript kodlarının nasıl yazılacağı ve nasıl çalıştırılacağı konusunda temel bir anlayış sağlar. JavaScript ayrıca fonksiyonlar ve olay işleme mekanizmaları aracılığıyla daha karmaşık işlevler ve kullanıcı etkileşimleri yönetimi sağlar [9]. Örneğin, bir web sayfasındaki bir butona tıklandığında gerçekleşen bir olayı işleyebilir ve buna göre dinamik bir yanıt üretебilir. JavaScript kullanılarak yapılan pratik uygulamalar, teorik bilginin yanı sıra gerçek dünya senaryolarında nasıl kullanılacağını gösterir. İnteraktif web elementleri, kullanıcı girişlerine yanıt veren dinamik içerikler ve kullanıcı deneyimini iyileştiren animasyonlar bu dili kullanarak oluşturulabilir. JavaScript ile yapılan bu uygulamalar, web geliştiricilerinin araç kutusunda vazgeçilmez bir yere sahiptir ve modern web geliştirmenin temel taşlarından biridir.

2.1.4 API

API (Uygulama Programlama Arayüzü), farklı yazılımlar arasında veri aktarımı ve iletişim kolaylaştıran bir arayüzdür. Bu arayüz, özellikle internet tabanlı uygulamalar, yazılım kütüphaneleri, donanım, işletim sistemleri ve veritabanları gibi alanlarda geniş kullanım alanına sahiptir. İnternet üzerinden erişilebilen uygulamalarda API'ler, istemci ve sunucu arasındaki bağlantıyı kurarak önemli bir işlev görür. API'lerin temel çalışma mekanizması, bir uygulamanın talebini API üzerinden sunucuya ileterek, sunucunun bu isteği işleyip bir yanıt dönmesi ve API'nin bu yanıtını uygulamaya aktarmasıdır. Bu süreçte, API kullanıcıların veri güvenliğini koruyarak, yalnızca gerekli bilgilerin sunucuya aktarılmasını sağlar. API'lerin tarihi, 1940'larda modüler yazılım kitaplıklarının geliştirilmesiyle başlar. 1968'de "Application Program Interface" terimi ilk kez bir makalede kullanılmış ve

1970'lerde daha geniş tanınırlık kazanmıştır. 1990'larda ise API, belirli işlevleri gerçekleştiren hizmetler dizisi olarak tanımlanmıştır. API türleri arasında Herkese Açık API'lar, Dahili API'lar ve Partner API'lar yer alır. Herkese Açık API'lar, geliştiricilere, üçüncü taraf uygulamalarda kullanılacak verileri sunar. Sosyal medya, harita, hava durumu ve ödeme sistemleri gibi API'lar buna örnek gösterilebilir. Dahili API'lar, yazılım geliştirme süreçlerini verimli hale getirirken, Partner API'lar iş ortaklarının ve müşterilerin özel kullanımı için tasarlınır ve iş birliklerini destekler. Farklı programlama dillerinde ve platformlarda geliştirilebilen API'ler, web üzerinden erişime sunulabilir [10]. Örneğin, bir e-ticaret sitesinde kullanıcının sepete eklediği ürün bilgileri ve ödeme süreçleri, API'ler aracılığıyla yönetilir.

2.1.5 MSSQL

Sürüm 7.0²'dan önce "kod tabanı", Microsoft'un kurumsal düzeyde veritabanı pazarına girişi olan Sybase SQL Server tarafından Microsoft'a satılıyordu. Temel olarak Sybase SQL Server 3.0 ile aynı olan ilk sürüm, SQL Server 1.0¹'ı oluşturmak ve pazarlamak için Microsoft, Sybase ve Ashton-Tate ile işbirliği yaptı. Microsoft SQL Sunucusu 1992 yılında sevk edildi. Daha sonra Microsoft SQL Server 4.21, Windows NT 3.1 ile aynı anda piyasaya sürüldü. Microsoft SQL Server 6.0, Windows NT için tasarlanan ilk sürümü ve Sybase'in talimatları olmadan piyasaya sürüldü. SQL Server 7.0, eski Sybase koduyla yazılmış bir "yeniden yazma" sürümü haline geldi ve yerini SQL Server 2000 aldı. SQL Server 2000, IA-64 mimarisinden farklı yazılan ilk sürümü. SQL Server 2000'den on yıl sonra performans artışı ve IDE araçlarını ve diğer tamamlayıcı sistemleri içeren SQL Server 2005 piyasaya sürüldü [11].

2.2 Donanım

2.2.1 STM32F407

STM32F407 Discovery, STMicroelectronics tarafından üretilen, ARM Cortex-M4 çekirdekli STM32F407VGT6 mikrodenetleyiciye dayanan bir geliştirme kartıdır. Bu kart, 168 MHz'e kadar çalışma frekansı, 1 MB Flash bellek ve 192 KB SRAM gibi özelliklerle donatılmıştır ve DSP ile FPU desteği sunar. Gelişmiş bağlantı seçenekleri arasında Ethernet, USB OTG, CAN, SPI, I2C ve USART bulunurken,

kart üzerindeki LED'ler ve bir ST-LINK/V2 debugger/programmer görsel geri bildirim sağlar. Genişleme yuvaları, analog çevre birimleri, çeşitli zamanlayıcılar ve enerji verimli tasarımlı sayesinde, STM32F407 Discovery, endüstriyel otomasyon, robotik, IoT ve diğer gömülü sistem projeleri için ideal bir platformdur. Kart, C/C++ gibi dillerle programlanabilir ve ST'nin geniş yazılım kütüphaneleriyle uyumlu olarak gelişmiş IDE'lerle çalışabilir. Bu özellikler, STM32F407 Discovery'yi hem eğitim hem de profesyonel uygulamalar için uygun ve güçlü bir seçenek yapar. Şekil 2.1' de gösterilmektedir [12].



Şekil 2.1: STM32F407

2.2.2 IRF540 Mosfet Sürücüsü

IRF520 bir güç MOSFET'idir (Metal Oksit yarı iletken alan Etkili Transistör) sürücüsü. bu entegre devre genellikle yüksek güçlü anahtarlama uygulamaları için kullanılır.DC motorlar, röleler ve yüksek akım ve gerilim gerektiren uygulamalar gibi. Şekil 2.2' de gösterilmektedir.



Şekil 2.2: IRF540 Mosfet Sürücüsü

2.2.3 Touch Sensör

Dokunma dedektörü veya dokunsal sensör olarak da bilinen dokunma sensörü, yüzeyine fiziksel dokunmayı veya teması algılayan bir tür giriş cihazıdır. Dokunsal girişi, bir mikro denetleyici veya başka bir elektronik sistem tarafından yorumlanabilecek bir elektrik sinyaline dönüştürmek için tasarlanmıştır. **Şekil 2.3'** de gösterilmektedir.



Şekil 2.3: Touch Sensör

2.2.4 Yağmur Sensörü

Yağmur dedektörü olarak da bilinen yağmur sensörü, yağmurun varlığını algılayan ve bu bilgiyi elektrik sinyallerine dönüştüren bir cihazdır. Bu sensörler genellikle araçlar için otomatik yağmur algılama silecek sistemlerinde, bahçe sulama



sistemlerinde ve sağanak yağış tespitinin gerekli olduğu diğer uygulamalarda kullanılır. **Şekil 2.4'** de gösterilmektedir.

Şekil 2.4: Yağmur Sensörü

2.2.5 Gaz Sensörü

Gaz sensörü, çevredeki çeşitli gazların varlığını veya konsantrasyonunu tespit eden bir cihazdır. Bu sensörler, endüstriyel süreçlerin izlenmesi, güvenlik sistemleri ve iç mekan hava kalitesinin izlenmesi dahil olmak üzere çok çeşitli uygulamalarda kullanılmaktadır. **Şekil 2.5'** de gösterilmektedir.



Şekil 2.5: Gaz Sensörü

2.2.6 Buzzer

Buzzer, sürekli veya aralıklı bir ses veya ton üreten elektronik bir cihazdır; çeşitli uygulamalarda genellikle sesli uyarılar, bildirimler veya alarmlar için kullanılır. Buzzer cihazları tipik olarak bir elektrik sinyali uygulandığında titreşen bir piezoelektrik elemandan veya bir elektromanyetik bobinden oluşur. Şekil 2.6' de gösterilmektedir.



Şekil 2.6: Buzzer

2.2.7 Step Motor

Step motor, dijital darbeleri hassas mekanik dönüşe dönüştüren bir tür elektrik motorudur. Geleneksel DC veya AC motorların aksine, adım motorları ayrı adımlarla veya artışlarla hareket eder, bu da onları hassas konumlandırma veya kontrol gerektiren uygulamalar için ideal kılar. Step motorun çalışması elektromanyetizma prensibine dayanır. tipik olarak bir rotor, stator ve bobinlerden oluşur. Rotor motorun dönen kısmıdır, stator ise bobinleri veya sargıları içerir. Bobinlere belirli bir sırayla enerji verildiğinde, oluşturulan manyetik alan rotorun

küçük hareket etmesine neden olur açısal artışlar veya adımlar. Şekil 2.7' de gösterilmektedir.



Şekil 2.7: Step Motor

2.2.8 Servo Motor

Servo motor, açısal veya doğrusal konum, hız ve ivmenin hassas kontrolünü sağlayan bir tür döner aktüatördür. robotik, uzaktan kumandalı araçlar, endüstriyel otomasyon ve havacılık sistemleri gibi doğru ve kontrollü hareketler gerektiren uygulamalarda yaygın olarak kullanılır. Şekil 2.8' de gösterilmektedir.



Şekil 2.8: Servo Motor

2.2.9 Mesafe Sensörü

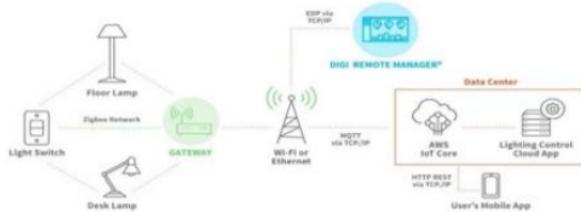
Ultrasonik mesafe sensörü, yüksek frekanslı ses dalgaları yayar ve ses dalgalarının bir nesneye çarptıktan sonra geri dönmesi için geçen süreyi ölçer. Sensör, uçuş süresini hesaplayarak nesneye olan uzaklığını belirleyebilir. Şekil 2.9' de gösterilmektedir.



Şekil 2.9: Mesafe Sensörü

3. IOT HABERLEŞME VE GÖMÜLÜ SİSTEMLER

IoT (Nesnelerin İnterneti) haberleşme ve gömülü sistemler, çeşitli haberleşme protokollerini kullanarak, sensörler ve mikrodenetleyiciler aracılığıyla çevrelerinden veri toplayan, işleyen ve paylaşan cihazlardan oluşur. Bu sistemler, Wi-Fi, Bluetooth, Zigbee, NB-IoT, LoRaWAN ve MQTT gibi protokollerle fiziksel dünyadan veri toplar ve bu verileri bulut tabanlı platformlara veya diğer cihazlara gönderir. Her protokol, farklı uygulama ihtiyaçlarına göre özelleşmiştir; örneğin, Bluetooth düşük enerji tüketimi ve kısa mesafe iletişim için idealdirken, LoRaWAN uzun mesafe ve düşük güç tüketimi gerektiren uygulamalarda tercih edilir. Gömülü sistemler, bu verileri yerel olarak işleyerek hızlı ve etkili tepki verme yeteneğine sahiptir ve genellikle enerji verimliliği ve kompakt boyutları ile öne çıkar [13]. Bu teknoloji, ev otomasyonundan endüstriyel otomasyon, sağlık izleme sistemlerine ve akıllı şehir uygulamalarına kadar geniş bir yelpazede kullanılmakta olup, cihazların birbiriley ve daha geniş bir ağ ile etkileşimde bulunmasını sağlayarak daha akıllı ve otomatize edilmiş çözümler sunar. IoT ve gömülü sistemlerin bu entegrasyonu, özellikle veri analizi ve yapay zeka ile birleştiğinde, çevresel koşullara göre kendini ayarlayabilen, öğrenen ve adapte edebilen dinamik ve etkileşimli sistemler yaratır.

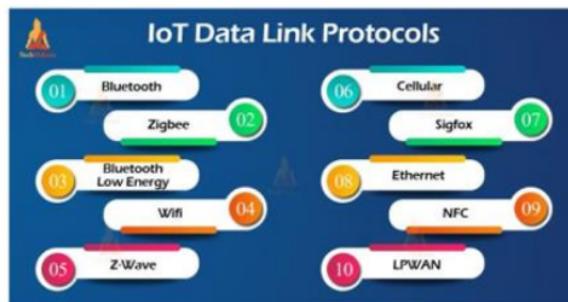


Şekil 3.1: Örnek IOT Sistemi

3.1 IoT Haberleşme Türleri

IoT (Nesnelerin İnterneti) haberleşmesi, çeşitli protokoller ve teknolojiler kullanarak geniş bir cihaz yelpazesini birbirine bağlar, bunlar arasında Wi-Fi,

Bluetooth, Zigbee, NB-IoT, ve LoRaWAN bulunur. Wi-Fi, ev ve iş yerlerinde yaygın olarak kullanılır ve yüksek hızlı veri aktarımı sağlar, ancak kısa menzil ve yüksek enerji tüketimi ile sınırlıdır. Bluetooth, özellikle düşük enerji versiyonu olan Bluetooth Low Energy (BLE) ile, kişisel alan ağları için idealdir ve düşük güç tüketimi sağlar. Zigbee, düşük güç tüketimi ve uzun batarya ömrü ile öne çıkar, genellikle ev otomasyonu ve akıllı ev cihazlarında kullanılır. NB-IoT, hücresel ağlar üzerinden düşük güç tüketimiyle uzun mesafe haberleşme sağlayarak, kentsel ve endüstriyel uygulamalar için uygundur [14]. LoRaWAN ise, uzun menzilli ve düşük güç tüketimli bir protokol olarak, kırsal ve uzak alanlarda IoT uygulamalarına olanak tanır. CoAP (Constrained Application Protocol), özellikle kaynak kısıtlı cihazlar için tasarlanmış bir web transfer protokolüdür ve IoT cihazlarının HTTP gibi daha ağır protokoller kullanmadan etkili bir şekilde iletişim kurmasını sağlar. Bu çeşitlilik, IoT'nin farklı gereksinimlere uyum sağlamasını ve geniş bir uygulama alanına hizmet etmesini mümkün kılar.



Şekil 3.2: Haberleşme Protokolleri

3.2 IoT Haberleşmesi Kapsam Alanları ve Veri Boyutları

IoT (Nesnelerin İnterneti) haberleşmesi, çeşitli kapsam alanlarına ve veri boyutlarına hitap eden, geniş bir iletişim teknolojisi yelpazesine sahiptir. Bu kapsam alanları ve ilişkili veri boyutları, IoT cihazlarının uygulama gereksinimlerine ve iletişim ortamlarına göre değişiklik gösterir:

Kişisel Alan Ağları (PAN): Bu kapsam alanı, genellikle Bluetooth ve Bluetooth Low Energy (BLE) gibi teknolojileri kullanır. PAN, bireysel kullanıcıların giyilebilir cihazlar, akıllı saatler ve sağlık izleme cihazları gibi küçük, kişisel cihazlarına bağlanmasını sağlar. Veri boyutları genellikle küçük ve düşük bant

genişliğine uygun olup, enerji verimliliği ve düşük gecikme süreleri önemlidir.

Yerel Alan Ağları (LAN) ve Ev Ağları: Wi-Fi ve Zigbee bu kategoriye örnek gösterilebilir. Ev otomasyon sistemleri, akıllı ev cihazları ve ofis ortamlarındaki IoT uygulamaları genellikle bu kapsam alanına girer. Veri boyutları, PAN'a kıyasla daha büyük olabilir ve daha yüksek veri hızları gerektirebilir.

Geniş Alan Ağları (WAN): NB-IoT, LoRaWAN ve hücresel ağlar (4G/5G) gibi teknolojiler, geniş alan IoT uygulamalarında kullanılır. Bu teknolojiler, şehir ölçüğündeki akıllı şehir uygulamaları, tarım, endüstriyel izleme ve uzaktan varlık takibi gibi uygulamalar için idealdir. Veri boyutları değişken olabilir ve genellikle düşük güç tüketimi ve uzun menzil iletişim ön planda tutar.

Küresel Ölçekli İletişim: Uydu tabanlı IoT sistemleri, küresel ölçekte haberleşme sağlar. Bu sistemler, genellikle denizcilik, lojistik ve küresel varlık takibi gibi uygulamalarda kullanılır. Veri boyutları genellikle küçük ve sınırlıdır, çünkü enerji ve bant genişliği sınırlamaları göz önünde bulundurulmalıdır.

Her bir kapsam alanı, IoT cihazlarının konumuna, enerji kullanımına, iletişim frekansına ve veri boyutlarına bağlı olarak farklı iletişim teknolojileri ve protokoller gereklidir. Bu çeşitlilik, IoT'nin çok geniş bir uygulama yelpazesine hizmet etmesini sağlar ve her türden cihazın ihtiyaçlarına uygun çözümler sunar.

3.3 Haberleşme Topolojileri

IoT haberleşme topolojileri, cihazların birbirleriyle ve ağ altyapılarıyla nasıl bağlandığını ve veri alışverişi yaptığı tanımlayan yapısal düzenlemelerdir. En yaygın IoT haberleşme topolojileri arasında yıldız, ağaç, mesh, halka ve otobüs topolojileri bulunur. Yıldız topolojide, tüm cihazlar merkezi bir düğüm (genellikle bir yönlendirici veya gateway) bağlanır ve bu merkezi düğüm ağ trafigini yönetir, bu yapı özellikle basit ve kolay yönetilebilir olmasıyla tercih edilir. Ağaç topolojisi, bir merkezi düğüm bağlı hiyerarşik düğüm katmanlarından oluşur, bu yapısıyla genişletilebilir ve esnek bir ağ oluşturur. Mesh topolojisi, cihazların birbirlerine doğrudan, dinamik bağlantılar kurduğu ve her bir cihazın ağıdaki diğer cihazlara veri iletmek için bir düğüm olarak hareket ettiği bir yapıdır; bu da yüksek güvenilirlik ve kapsama alanı sağlar ancak karmaşık ve enerji yoğun olabilir. Halka topolojisinde, her cihaz yalnızca iki komşuya bağlantı kurar ve veriler bir döngü içinde iletilir, bu yapının basitliği ve düşük maliyeti avantaj sağlarken, tek bir

bağlantı hatası tüm ağı etkileyebilir. Son olarak, otobüs topolojisinde tüm cihazlar ortak bir iletim ortamına bağlıdır ve veriler bu ortam üzerinden iletilir, bu yapı kolay kurulum ve düşük maliyet avantajları sunarken, ağıın genişlemesi ile performansı düşebilir [15]. Her topoloji, IoT cihazlarının dağılımı, iletişim gereksinimleri ve uygulamanın özgünlüğüne göre avantajlar ve dezavantajlar sunar, bu da IoT sistemlerinin tasarıminda kritik bir rol oynar.

3.4 IoT Alanında Gömülü Sistemler

IoT (Nesnelerin İnterneti) alanında gömülü sistemler, çevredeki fiziksel dünyadan veri toplayıp işleyen ve bu verileri internet üzerinden diğer cihazlarla veya sunucularla paylaşan, mikrodenetleyiciler veya mikroişlemcilerle donatılmış akıllı cihazlardır. Bu sistemler, genellikle enerji verimliliği, kompakt boyut ve belirli bir görevi etkin bir şekilde yerine getirebilmek için optimize edilmiş donanım ve yazılım içerirler. Gömülü IoT cihazları, sensörler ve aktüatörler aracılığıyla çevresel parametreleri (sıcaklık, nem, hareket gibi) algılayabilir, bu verileri işleyebilir ve kablosuz haberleşme teknolojileri (Wi-Fi, Bluetooth, Zigbee, LoRaWAN gibi) kullanarak ağa bağlanabilirler. Bu cihazlar, ev otomasyonundan sağlık izleme sistemlerine, endüstriyel otomasyondan çevresel izlemeye kadar geniş bir uygulama yelpazesinde kullanılır. Gömülü IoT sistemlerinin ana avantajları arasında düşük güç tüketimi, yüksek güvenilirlik ve uzun ömürlü performans bulunur. Ayrıca, bu sistemler genellikle gerçek zamanlı işleme kapasitesine sahiptir ve özellikle sürekli izleme ve kontrol gerektiren uygulamalarda etkilidirler. IoT ekosistemi içinde, gömülü sistemler, akıllı ve bağlantılı bir dünya oluşturmanın temel taşlarından biri olarak kabul edilir.

4. AKILLI ÇİFTLİK SİSTEMİNİN GERÇEKLEŞTİRİLMESİ

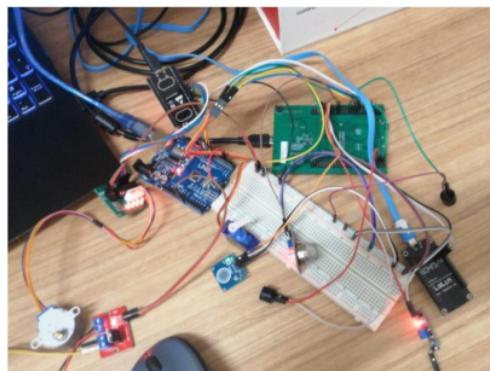
4.1 Giriş

Hayvan çiftliklerinde kas gücüne dayalı geleneksel yöntemlere son vermek ve buna teknolojik çözüm üretmek proje hedefidir. Bu doğrultuda dört ana hedef var. İlk hedef hayvanların düzenli ve ölçülü beslenmesini sağlamak. Bunun için uzaktan kontrol edilen yemlikler var. Ağ üzerinde kontrol sağlanarak yemler belirli oranlarda oluklara dökülmektedir. İkinci hedef ise sürüün bütünlüğünü sağlamak. Ahır girişindeki kapıda bulunan sensör sayesinde ahıra giren hayvanların sayımı yapılmaktadır. Mevcut sürü sayısına ulaşamaması halinde çiftlik sahiplerini uyarın alarm sistem devreye girmektedir. Üçüncü amaç hayvanları olumsuz hava koşullarından korumaktır. Yağmur veya kar yağması durumunda kullanıcı uyarılacak olup, ahırın üstü istege bağlı olarak uzaktan kontrol edilip kapatılabilecektir. Dördüncü ve son hedef ise hayvanların güvenliğini sağlamaktır. Çiftlikte bulunan sensör sayesinde olası bir yangın durumunda kullanıcı uyarılacak ve kullanıcıya uzaktan müdahale imkanı sunulacaktır. Akıllı hayvan çiftliği tasarımlı, hayvanların çevre koşullarından korunması, beslenmesi, sürü bütünlüğü, hayvanların olası kazalardan korunması gibi birçok faktörü izleyebilir ve bu bilgileri yöneticilere sunarak çiftlik yönetiminde daha bilinçli kararlar alınmasına yardımcı olabilir.

4.2 Sistemin Donanımsal Tasarımı

Projenin taskları neticesinde en uygun sensörler olarak; yangın kontrolü için gaz sensörü kullanmayı, hayvan sayılarının ağıl girişinde alınması için touch sensörü kullanmayı, yağmur kontrolü için yağmur sensörü kullanmayı, yem takibinin yapılabilmesi için ise yemlikte mesafe takibiyle yem oranını belirlemesi için mesafe sensörü kullanılmaktadır. Tüm bu sensörlerden veriler SMT32407 kullanılarak alınmış ve ESP32 ile API'ler üzerinden web sitesine iletilmektedir. Şekil 4.1'de gösterildiği gibi sistem fonksiyonları test edilmiş daha sonrasında

makete yerleştirmiştir.



Şekil 4.1: Elektronik Test Devresi

4.3 Sistemin Yazılımsal Tasarımı

Akıllı çiftlik sisteminin yazılımında birden fazla programlama dili kullanılmıştır. Projenin frontend kısmı Neighbor Blazor kullanılarak geliştirilmiştir ve kullanıcıların anlık olarak çiftlikteki hayvanların takibini, yanın takibini, yem takibini ve çati takibini yapabileceği bir sistem oluşturulmuştur. Backend tarafı .NET ile C# kullanılarak yazılmıştır. STM32F407 mikrodenetleyicisi C programlama diliyle kodlanmış olup HAL kütüphanesinden yararlanılmıştır. Sensörlerden alınan veriler anlamlandırmış ve bu veriler UART haberleşme protokolü kullanılarak ESP32 mikrodenetleyicisi tarafından alınması sağlanmış veriler anlamlandırılarak API' ler ile sunucuya iletimi sağlanmıştır.

4.4 STM32F407 Kartının Kodlanması

STM32F407 kartının öncelikler sensörlerden veri alabilmesi için çevresel birimleri açılmış ve gerekli fonksiyonlar yazılmıştır. HAL kütüphanesinden yararlanılan projede kodlama dili olarak C programlama dili kullanılmıştır. Sekil 4.2'de yer alan kod örneğinde sensörlerden veri alımı gösterilmiştir.

```

yagmur_status = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_1);
if(yagmur_status==0)
{
    data_send[0]='1';
}
else
{
    data_send[0]='0';
}

gaz_status = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_4);
if(gaz_status==0)
{
    data_send[1]='1';
}
else
{
    data_send[1]='0';
}

touch_status = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_5);
if(touch_status==1)
{
    count=count+1;

    if(count>9)
    {
        count=0;
    }
}

```

Sekil 4.2: Sensör Verileri Çekme Kodu

Alınan sensör verileri data_send adlı arraye kaydedilerek ESP32'ye yollanmaktadır. UART ile STM32F407'den ESP32'ye veri gönderme kodu [Şekil 4.3'te](#) yer almaktadır.

```
HAL_UART_Transmit(&huart2, (uint8_t*)data_send, 6, 100);
```

Sekil 4.3: STM UART Veri Gönderme Kodu

Web'den gelen veriler ESP32'ye API'ler aracılığıyla iletilmektedir. İletilen veriler UART ile STM32F407'ye gönderilmektedir. STM32F407'de veriler [Şekil 4.4'de](#) gösterilen kod ile alınmaktadır.

```
HAL_UART_Receive(&huart2, (uint8_t*)data_take, 7,2000); //pa2 tx, pa3 rx
```

Sekil 4.4: STM UART Veri Alma Kodu

4.5 ESP32 ile Veri İletimi

ESP32 mikrodenetleyicisi API'ler ile veri alışverişi sağlamak için kullanılmaktadır. STM32F407'den veri almak için yazılan kod [Şekil 4.5'te](#) gösterilmektedir.

```
bytesReceived = Serial.readBytesUntil('>', messageBuffer, 6);
```

Şekil 4.5: ESP UART Veri Gönderme Kodu

Alınan verileri API' lere gönderen örnek kod 5 aşağıda Şekil 4.6' da yer almaktadır.

```
if (messageBuffer[0] == '1')
{
    roofpostfunction();
    int httpResponseCode = http.POST("{\"isRain\":\"1\"}");
}
```

Şekil 4.6: ESP API Veri Gönderme Kodu

API' lerden veri çeken örnek kod 5 aşağıda Şekil 4.7' de yer almaktadır.

```
String owmWebrooF = "http://ipmapi.ipmsoft.com.tr/api/MoveSensor/GetBuzzerEndData" ;

String owmJson = getWebPage(owmWebrooF.c_str());
JSONVar jsonData = JSON.parse(owmJson);
if (JSON.typeof(jsonData) == "undefined")
{
    // Serial.println("Json data ayıklanamadı.");
    return;
}
// Serial.print("Roof Durumu : ");
int touchbuzzerdurum = (jsonData["buzzerActivity"]);
```

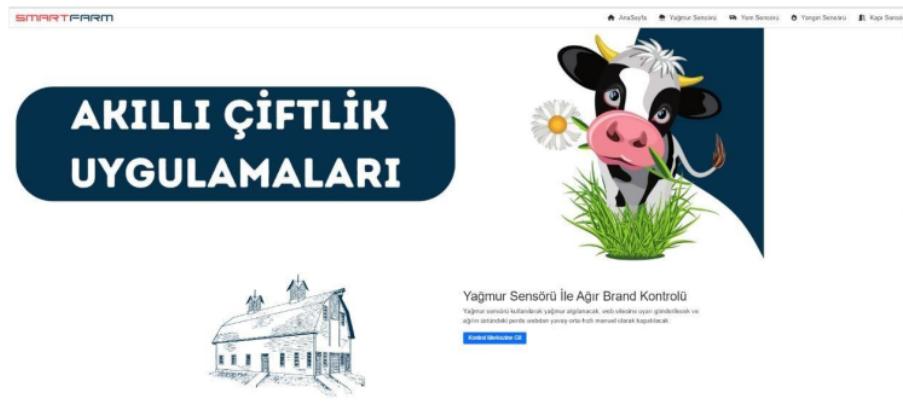
Şekil 4.7 ESP API Veri Alma Kodu

4.6 Web Arayüzünün Oluşturulması

Çiftlik sahiplerinin sistemi takip edebilmesi ve aksiyon alabilmesi için yapılan web arayüzünün Frontend kısmı Neighbor Blazor, backend kısmında .NET ve C# kullanılarak kullanıcı dostu bir arayüz geliştirilmiştir. Geliştirilen Web arayüzü 5 sayfadan oluşmaktadır. İlk olarak kullanıcı Anasayfa ile karşılaşmakta, sırası ile Yağmur Sensörü, Yem Sensörü, Yangın Sensörü ve Kapı Sensörü yer almaktadır.

4.6.1 Ana Sayfa

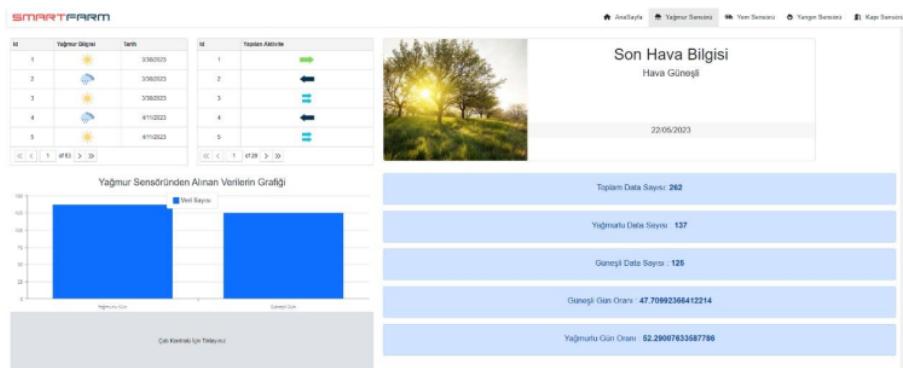
Çiftlik sahiplerinin siteye giriş yaptıklarında ilk karşılaşıkları ekran 6 Şekil 4.8' de yer almaktadır.



Şekil 4.8: Hoş Geldin Ekranı

4.6.2 Yağmur Kontrol Paneli

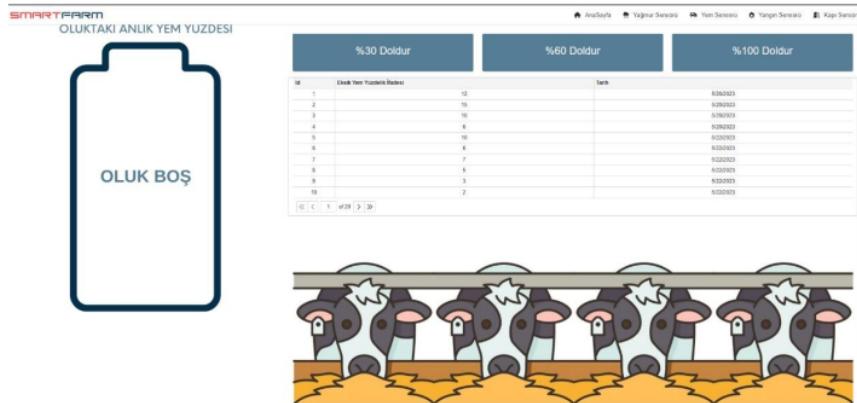
Bu ekranada yağmur sensörümüzden gelen anlık verilerle hava durumunu güncelliyor ve görsellerle desteklenmektedir. Aynı zamanda veritabanına kaydedilen verileri soldaki tablo ile listelenmektedir. Kullanıcının isteğine göre çatı kontrol paneli bulunmaktadır. Bu işlem sol alttaki tavan kontrol butonu ile gerçekleştirilir. Seçilen seçeneklere bağlı olarak çatının görüntülenme hızı da sitemin sol tarafında tablo halinde gösterilmektedir. Şekil 4.9'da detaylı olarak görülmektedir.



Şekil 4.9: Yağmur Kontrol Paneli

4.6.3 Besleme Sistemi Kontrol Paneli

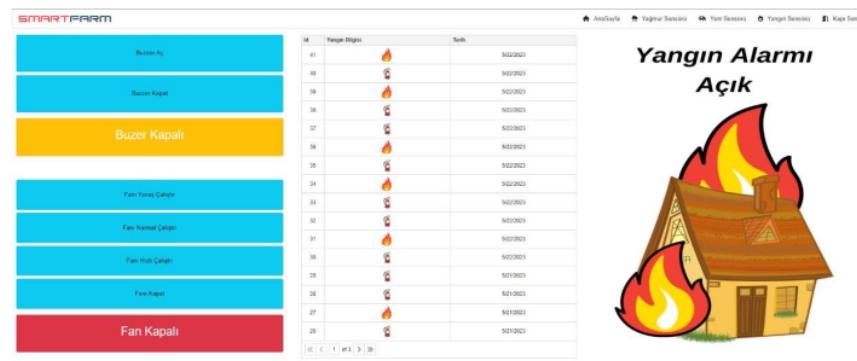
Bu ekranda ahırdaki yemliklerin yem doluluk oranı ölçülüp gösterilmektedir. Aynı zamanda yemin belirlenen oranda oluklara dökülmesi sağlanmaktadır. Ekran [Şekil 4.10](#)' da gösterilmektedir.



Şekil 4.10: Besleme Sistemi Kontrol Paneli

4.6.4 Yangın Kontrol Paneli

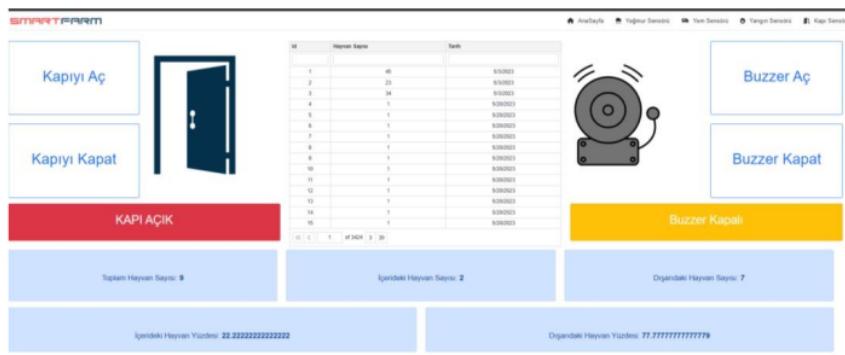
Çiftlikteki duman sensöründen gelen verilerle yangın olup olmadığı tespit edilen ve bu bilgiyi kullanıcıya aktarılan ekrandır. Aynı zamanda yangın durumunda çiftlikte bulunan buzzer ve fan kullanıcının isteğine göre çalışmaktadır. Ekran görüntüsü 4.11'de yer almaktadır.



Şekil 4.11: Yangın Kontrol Paneli

4.6.5 Kapı Kontrol Paneli

Çiftlik kapısı isteğe göre açılıp kapanabilmektedir. Aynı zamanda touch sensör sayesinde içerisindeki hayvan sayısını sayılabilmektedir. Böylece çiftliğin içindeki ve dışındaki hayvan sayısını sayılabilmektedir. Hayvanların kaybolması durumunda alarm devreye sokulmaktadır. Tüm detaylar [Şekil 4.12' de gösterilmektedir.](#)



Şekil 4.12: Kapı Kontrol Paneli

4.6.6 Veri tabanı Tasarım

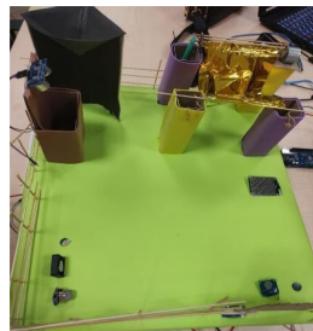
Sensörlerden gelen verilerin web uygulamasında kullanılmak üzere kaydedilmesine yönelik bir veri tabanı tasarımı yapılmıştır. Toplanan tüm bilgiler [Şekil 7.1'de görüldüğü gibi tasarlanan bir tabloya aktarıldı](#). Bu tablo tek başına oluşturulduğunda içindeki bilgilerle birlikte oldukça büyük bir boyuta sahiptir. Erişimini daha hızlı ve kolay olması için tek tablo yerine birden fazla tablonun yer aldığı bir yapı oluşturulmasına karar verildi. Tablodan örnek [Şekil 4.13' te yer almaktadır.](#)

Id	BuzzerActivity	Date	Id	isFire	Date	Id	AnimalNumber	Date
1	1	2023-06-21	1	1	2023-06-21	1	45	2023-05-03
2	0	2023-06-21	2	0	2023-06-21	2	23	2023-05-03
3	1	2023-06-21	3	1	2023-06-21	3	34	2023-05-03
4	2	2023-05-08	4	0	2023-06-21	4	1	2023-05-20
5	1	2023-05-08	5	1	2023-06-21	5	1	2023-05-20

Şekil 4.13: Veri Tabanı

4.7 Deneysel Sonuçlar

Bu bitirme tezi, IOT tabanlı bir akıllı çiftlik sistemi geliştirmek amacıyla gerçekleştirilmiştir. Projede, sensör verilerinin web tabanlı bir arayüz aracılığıyla izlenmesi ve kontrol edilmesi sağlanmıştır. Sistem dört tasktan oluşmaktadır: yem seviyesi kontrolü, yağmur varlık kontrolü, yangın kontrolü ve hayvan sayısı kontrolü. Sensörlerden alınan bu veriler API'ler aracılığıyla web sunucusuna yönlendirilmekte ve kullanıcının aksiyon almasına imkan vermektedir. Çift yönlü haberleşme sayesinde kullanıcı anlık veri takibi yapabilmekte ve anlık tepki verebilmektedir. Test süreci boyunca ekranlardan veri kontrolü yapılmış; yem eklenmiş, çatı açılıp kapatılmış, fan hızı kademeli olarak ayarlanmış ve hayvan sayısına göre kapı kapatılmış ya da alarm çalıştırılmıştır. Sistemin modellemesi Şekil 5.1' de gösterilmektedir.



Şekil 4.14: Akıllı Çiftlik Modellemesi

5. SONUÇLAR

Bu proje, IOT teknolojisi kullanılarak oluşturulan bir smart farm sistemidir. Bu sistem, çiftliğin web tabanlı bir arayüz aracılığıyla gerçek zamanlı olarak izlenmesi ve kontrol edilmesini sağlamaktadır.

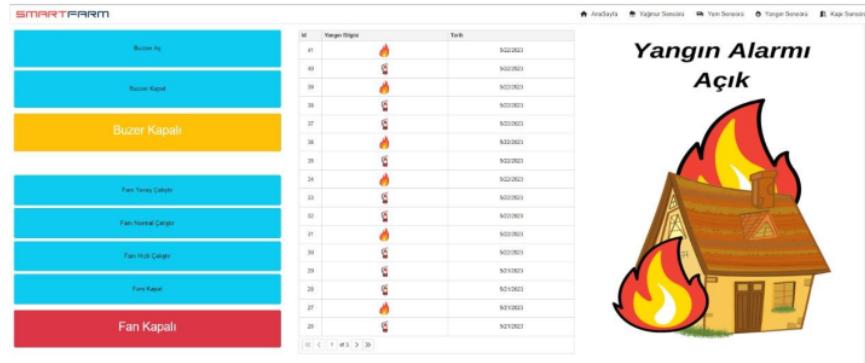
Çiftliklerin izlenmesi ve takibi günümüzde çiftlik sahipleri için bir gereklilik haline gelmiştir. Çiftliklerdeki hayvanların takibi ve güvenliği çiftlik sahipleri oldukça önemlidir. Sistem tüm bu imkanları sensörlerden aldığı bilgiler sayesinde sağlamakta ve kullanıcı için oluşturulan arayüz sayesinde kontrol edilebilir hale getirmektedir.

Proje, frontend kısmında React kullanılarak ¹⁷ kullanıcı dostu bir arayüz geliştirilmiştir. Bu arayüz aracılığıyla kullanıcılar, anlık olarak çiftlikteki yem durumunu, çati konumunu, yanın durumunu ve hayvan sayısını görebilmektedir. Aynı zamanda yemin bitmesi durumunda yem eklemesi yapabilmekte, yağmur yağması durumunda çatıyı kapatabilmekte, hayvan sayısına göre; hayvan sayısının tam olması durumunda kapıyı kapatarak hayvanların içerisinde kalmasını sağlamakta, eksik olması durumunda çalışanları uyarmak için alarmı çalıştırabilmekte ve aynı şekilde yanın durumunda çalışanları uyarmak için yanın alarmını aktif hale getirebilmektedir.

Bu proje gömülü taraf ve web tarafı olmak üzere 2 farklı sistemden oluşmaktadır. Projede kullanılan STM32F407 Discovery ile sensörlerden ADC, I2C kullanılarak veriler alınır. Alınan veriler ESP32 ile API aracılığıyla web sitesine ilettilir. Gönderilen veriler doğrultusunda kullanıcı tasklar neticesinde aksiyon alır. Bu tasklar; yağmur sensöründen alınan veri neticesinde ağılmın üstündeki perde kontrolü, gaz sensöründen alınan veriye göre fan kontrolü, touch sensörden alınan veriye göre hayvan sayısının kontrol edilmesi gelen buna göre kapı kontrolü ve mesafe sensörü kullanılarak yem oranının izlenmesi ve yem sağlanmasına yöneliktir.

Arayüz ile yapılan testlerde gaz sensörüne duman verilerek yanın alarmı aktif edilir. Yanın alarmının aktif olmasıyla arayüzün yanın sensörü panelinde yer alan evin etrafında yanın çıkmaktadır. Tüm tasklar bu mantıkta çalışmaktadır ve sistem

stabilitesi sağlanmaktadır.



Sekil 5.1: Yangın Alarmı

6. KAYNAK

- [1] Ali Kaan, Mehmetcan Aşık, Halis Seçme, Murat Karaer "Tarımda Nesnelerin Interneti (IoT) ve Uygulamaları", 2018
- [2] Tuncay Ercan, Mahir Kutay "Endüstride Nesnelerin Interneti (IoT) Uygulamaları", 2016
- [3] Yeliz Durgun "Nesnelerin İnterneti Teknolojisinin Kümes Ortamında Uyulanması ve Etkileri", 2019
- [4] C(programlama dili), Wikipedia
- [5] C Kullanımı ve Fonksiyonları , C_Lab
- [6] C Sharp Programlama Dili, VikiKitap
- [7] C# Kullanarak Mesafe Giderme, İrfan Duman, Resul Kara, Erkan Çetiner
- [8] Şahin İnanç "C# Programlama Dilinde Gerçekleştirilen Program ile Euler Sayısının Rasgeleliğinin Sınanması", 2021
- [9] Javascript, Mozilla
- [10] API, WebDevelopers
- [11] Ferat Aktaş "Hastane Otomasyon Projesi", 2014
- [12] STM32F407 Discovery, EmbeddedCore
- [13] Hayati Memur, Zeynep Dicle, Süleyman Erdener "İç Ortam Bitki Takibi için IoT Tabanlı Akıllı Gömülü Sistem Tasarımı", 2022
- [14] Metmet Ali Ebleme "Nesnelerin İnterneti Uygulama Katmanı Haberleşme Protokollerinin Başarım Analizi", 2019

[15] Temel Ağ Topolojileri, Çözümpark

% **6**
BENZERLİK ENDEKSİ

% **5**
İNTERNET KAYNAKLARI

% **0**
YAYINLAR

% **3**
ÖĞRENCİ ÖDEVLERİ

BİRİNCİL KAYNAKLAR

- | | | | |
|--|----------|----------------------------------------------------|-------------|
| | 1 | acikbilim.yok.gov.tr | % 1 |
| | | Internet Kaynağı | |
| | 2 | openaccess.maltepe.edu.tr | % 1 |
| | | Internet Kaynağı | |
| | 3 | Submitted to Istanbul Aydin University | <% 1 |
| | | Öğrenci Ödevi | |
| | 4 | Submitted to Düzce Üniversitesi | <% 1 |
| | | Öğrenci Ödevi | |
| | 5 | Submitted to Eskisehir Osmangazi University | <% 1 |
| | | Öğrenci Ödevi | |
| | 6 | dspace.yildiz.edu.tr | <% 1 |
| | | Internet Kaynağı | |
| | 7 | Submitted to Afyon Kocatepe University | <% 1 |
| | | Öğrenci Ödevi | |
| | 8 | Submitted to Southampton Solent University | <% 1 |
| | | Öğrenci Ödevi | |
| | 9 | campustechnology.com | <% 1 |
| | | Internet Kaynağı | |

10	ehm.kocaeli.edu.tr Internet Kaynağı	<% 1
11	www.slideshare.net Internet Kaynağı	<% 1
12	vdocuments.net Internet Kaynağı	<% 1
13	9lib.net Internet Kaynağı	<% 1
14	askabiologist.asu.edu Internet Kaynağı	<% 1
15	openaccess.hacettepe.edu.tr Internet Kaynağı	<% 1
16	www.cz.endress.com Internet Kaynağı	<% 1
17	www.masfor.ee.boun.edu.tr Internet Kaynağı	<% 1
18	tr.wikipedia.org Internet Kaynağı	<% 1

Alıntıları çıkart

üzerinde

Eşleşmeleri çıkar

Kapat

Bibliyografyayı Çıkart üzerinde