

AKVARYUM KONTROL SİSTEMİ

LİSANS TEZİ

Ali KAZANCI

Fatih İLHAN

Yusuf ÇINARCI

Danışman

Doç. Dr. İSMAİL KOYUNCU

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

Temmuz 2022

**AFYON KOCATEPE ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ**

LİSANS TEZİ

AKVARYUM KONTROL SİSTEMİ

Ali KAZANCI

Fatih İLHAN

Yusuf ÇINARCI

Danışman

DOÇ. DR. İSMAİL KOYUNCU

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

Temmuz 2022

TEZ ONAY SAYFASI

Adı SOYADI tarafından hazırlanan “Tez Onay Sayfası” adlı tez çalışması lisansüstü eğitim ve öğretim yönetmeliğinin ilgili maddeleri uyarınca GG / AA / YYYY tarihinde aşağıdaki jüri tarafından **oy birliği / oy çokluğu** ile Afyon Kocatepe Üniversitesi Fen Bilimleri Enstitüsü **Anabilim Dalı Adı Anabilim Dalı’nda YÜKSEK LİSANS TEZİ / DOKTORA TEZİ** olarak kabul edilmiştir.

Danışman : Doç. Dr. İSMAİL KOYUNCU

Afyon Kocatepe Üniversitesi
Fen Bilimleri Enstitüsü Yönetim Kurulu'nun
..... /..... /..... tarih ve
..... sayılı kararıyla onaylanmıştır.

.....

BİLİMSEL ETİK BİLDİRİM SAYFASI

Afyon Kocatepe Üniversitesi

Fen Bilimleri Enstitüsü, tez yazım kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içindeki bütün bilgi ve belgeleri akademik kurallar çerçevesinde elde ettiğimi,
- Görsel, işitsel ve yazılı tüm bilgi ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu,
- Başkalarının eserlerinden yararlanılması durumunda ilgili eserlere bilimsel normlara uygun olarak atıfta bulunduğumu,
- Atıfta bulunduğum eserlerin tümünü kaynak olarak gösterdiğim,
- Kullanılan verilerde herhangi bir tahrifat yapmadığımı,
- Ve bu tezin herhangi bir bölümünü bu üniversite veya başka bir üniversitede başka bir tez çalışması olarak sunmadığımı

beyan ederim.

05 / 07 / 2022

İmza

Ali KAZANCI Fatih İLHAN Yusuf ÇINARCI

ÖZET

Lisans Tezi

AKVAYUM KONTROL SİSTEMİ

Ali KAZANCI, Fatih İLHAN, Yusuf ÇINARCI

Afyon Kocatepe Üniversitesi

Teknoloji Fakültesi

Elektrik-Elektronik Mühendisliği Anabilim Dalı

Danışman: DOÇ.DR İsmail KOYUNCU

Bu proje esas olarak Raspberry Pi adlı tek çip kullanan bilgisayar ile bir Akvaryum İzleme Sistemi oluşturmaya yönelikdir. Bu projenin temel amacı, kapalı akvaryumlarının bakımını yapmakta güçlük çekenlere, özellikle de sık sık dışarıda kalanlara bu nedenle akvaryumlarını sürekli izleyemeyenlere yardımcı olmaktır. Bu sistem sayesinde kullanıcılar, akvaryumlarını sade ve basit bir arayüzü üzerinden kontrol edebilir ve izleyebilirler. Bu sistemin ana rolü, kullanıcıların balıkları zamanında besleme, su sıcaklığını, su seviyesini kontrol etme gibi görevleri içeren bir dizi süreci kontrol etmektir. Suyu pompalar aracılığı ile uzaktan doldurup boşaltma gibi ana işlevler bu sistem sayesinde oldukça kolay hale getirilmiştir. Akvaryumda su sızıntısı veya akvaryumun su seviyesinin normal seviyelerin altına düşmesi gibi acil durumlar olduğunda, kullanıcının arayüz üzerinde uyarı alması sağlanır. Akvaryumun durumunu sürekli olarak kontrol etmek için, Raspberry Pi 3, sensörlerden veri toplamak ve ardından verileri kaydetmek, sensörleri ve aktuatörleri kontrol etmek gibi işlevleri yerine getiren ana karttır. Kullanıcı daha sonra akvaryumlarının durumunu kontrol etmek için Raspberry Pi tarafından barındırılan özel olarak tasarlanmış arayüzüne giriş yapacaktır. Bunun yanında su sıcaklık sensörü, sistemin en iyi şekilde çalıştığından emin olmak için su sızıntı sensörü ve su seviye sensörü de gereklidir. Bunun dışında su pompa, su filtresi ve balıkları beslemek için motorlar da gereklidir. Bu proje ile akvaryum otomasyonu gerçekleştirilmiş ve belli başlı sensörler yardımıyla uzaktan kolay ve sade bir arayüz yardımıyla akvaryum kontrolü gerçekleştirilebileceği gösterilmiştir.

Anahtar Kelimeler: Akvaryum Kontrolü, Raspberry Pi 3, Python Tkinter

ABSTRACT

Sc. Thesis

AQUARIUM CONTROL SYSTEM

Ali KAZANCI, Fatih ILHAN, Yusuf CINARCI

Afyon Kocatepe University

Faculty of Technology

Department of Electrical and Electronic Engineering

Supervisor: DOÇ.DR İsmail KOYUNCU

This project is primarily intended to create an Aquarium Monitoring System with a single chip-using computer called Raspberry Pi. The main purpose of this project is to help those who have difficulty maintaining closed aquariums, especially those who are often outdoors, so those who cannot monitor their aquarium at all times. With this system, users can control and monitor their aquariums through a simple and simple interface. The main role of this system is to control a number of processes that include tasks such as user feeding fish in time, checking water temperature, water level. The main functions such as filling and draining water remotely via pumps are made very easy thanks to this system. In case of emergencies such as water leakage in the aquarium or the water level of the aquarium falls below normal levels, the user is provided with an alert on the interface. To continuously check the condition of the aquarium, the Raspberry Pi 3 is the main board that performs functions such as collecting data from sensors and then recording data, checking sensors and actuators. The user will then log into the specially designed interface hosted by Raspberry Pi to check the status of their aquarium. In addition, the water temperature sensor and the water level sensor are also required to ensure that the system is functioning optimally. In addition, engines are also required to feed the water pump, water filter and fish. This project has been used to automate the aquarium and has shown that aquarium control can be performed with the aid of a simple and simple interface from a remote distance with the help of certain sensors.

Keywords: Aquarium Control, Raspberry Pi 3, Python Tkinter

TEŞEKKÜR

Bu araştırmanın konusu, deneysel çalışmaların yönlendirilmesi, sonuçların değerlendirilmesi ve yazımı aşamasında yapmış olduğu büyük katkılarından dolayı tez danışmanım Sayın Doç.Dr İsmail Koyuncu, araştırma ve yazım süresince yardımcılarını esirgemeyen Sayın Doç.Dr İsmail Koyuncu'ya her konuda öneri ve eleştirileriyle yardımcılarını gördüğüm hocalarına ve arkadaşlarına teşekkür ederim.

Bu araştırma boyunca maddi ve manevi desteklerinden dolayı ailelerimize teşekkür ederiz.

Ali KAZANCI

Fatih İLHAN

Yusuf ÇINARCI

Afyonkarahisar 2022

İÇİNDEKİLER DİZİNİ

ÖZET	i
ABSTRACT	ii
TEŞEKKÜR	iii
İÇİNDEKİLER DİZİNİ.....	iv
SİMGELER ve KISALTMALAR DİZİNİ	v
RESİMLER DİZİNİ	vi
1.GİRİŞ	1
1.1 Akvaryum nedir?	1
1.2 Motivasyon ve Sorun Bildirimi;	1
1.3 Proje Kapsamı	3
1.4 Proje Amacı	3
1.5 Etki, Önem ve Katkı	3
2. LİTERATÜR TARAMASI	4
3. MATERİYAL ve METOT	10
3.1 Sisteme Genel Bakış	10
3.2 Tasarım Özellikleri	19
3.3 Donanım ve Yazılım	22
4. BULGULAR	77
4.1 Otomatik Mod	77
4.2 Manuel Mod	81
6. KAYNAKLAR	98

SİMGELER ve KISALTMALAR DİZİNİ

Simgeler ve Kısalmalar

Ma	Miliampere
PIC	Çevresel arabirim denetleyicisi
nm	Nanometre
mm	Milimetre
cm	Santimetre
kg	Kilogram
%	Yüzde
W	Watt
V	Volt
D	Diyot
R	Direnç
°C	Santigrat derece
IoT	Internet Of Things
IDE	Integration Development Environment
GUI	Graphical User Interface
OS	Operating System
SDLC	Software Development Life Cycle
USB	Universal Serial Bus
DC	Direct Current
LED	Light Emitting Diode
RGB	Red, Green, Blue
GPIO	General Purpose I/O pins
ADC	Analog to Digital Converter
RTC	Real Time Clock
GPRS	General Packet Radio Service
SoC	System On A Chip

RESİMLER DİZİNİ

Şekil 1 AURORA'nın Mimari Tasarımı ve Kullanım Örneği Tasarımı	6
Şekil 2 FishTalk için basitleştirilmiş bir blok diyagram.....	7
Şekil 3 Akvaryum monitör sisteminin genel blok diyagramı.....	10
Şekil 4 Akvaryum Kontrol Sistemi Kullanıcı Arayüzü (GUI)	11
Şekil 5 Akvaryum izleme sisteminin elektronik bileşenlerin fiziksel bağlantıları	12
Şekil 6 Otomatik mod için sistemin algoritma akış şeması.....	13
Şekil 7 Akvaryum Kontrol Sistemi Kullanıcı Arayüzü (GUI)	14
Şekil 8 Arayüz üzerinde manuel mod işlemlerinin gerçekleşme akış şeması	15
Şekil 9 Genel arayüz akış şeması	16
Şekil 10 Kullanıcının Otomatik mod işlemlerini gerçekleştirmeye akış şeması.....	18
Şekil 11 Kullanıcı arayüzü ile sistemin genel blok şeması	19
Şekil 12 Kullanıcı arayüzü ile sistemin genel blok şeması	20
Şekil 13 Raspberry Pi Model 3B genel görünümü	24
Şekil 14 Raspberry Pi 3 Pinli Teknik Özellikler	24
Şekil 15 Raspberry pi örnek ekran görüntüsü	25
Şekil 16 MCP3008 ADC Dönüştürücü	25
Şekil 17 MCP3008 ADC Dönüşürücünün LDR ve Raspberry Pi 3 ile Bağlantısı.....	26
Şekil 18 MCP3008 pin diyagramı	27
Şekil 19 MCP 3008 adc dönüştürücünün test kodları	28
Şekil 20 LDR sensörü ile MCP 3008 adc dönüştürücü test kodları.....	29
Şekil 21 LDR sensör test işlemleri	29

Şekil 22 HC-SR04 Ultrasonik Mesafe Sensörü.....	30
Şekil 23 HC-SR04 ve Raspberry Pi bağlantısı	31
Şekil 24 Direnç bağlantısı	32
Şekil 25 HCSR-04 sensör kodları.....	33
Şekil 26 Ultrasonik sensör deney işlemleri	34
Şekil 27 Turbidity su bulanıklık sensörü.....	35
Şekil 28 Bulanıklık sensörünün MCP3008 ADC dönüştürücü ve Raspberry Pi 3 ile bağlantısı.....	36
Şekil 29 MCP3008 Pin diyagramı	37
Şekil 30 Buraya bulanıklık sensör kod ekranı eklenecek.	38
Şekil 31 DS18B20 Su Geçirmez Sıcaklık Sensörü	39
Şekil 32 One-Wire DS18B20 sensör ile Raspberry arasındaki bağlantı devre şeması ..	40
Şekil 33 DS18B20 Sıcaklık Sensörü Test Kodları	41
Şekil 34 DS18B20 Sıcaklık sensörü test işlemleri	42
Şekil 35 28BYJ48 Step motor	43
Şekil 36 SULN2003 Step motor sürücü pcb kartı	44
Şekil 37 28BYJ48 ile ULN2003 Motor sürücü kartı ve Raspberry Pi 3 arasındaki bağlantı	45
Şekil 38 Step motor test kodları	46
Şekil 39 Step motor bağlantı şekli ve çalışma deneyi	48
Şekil 40 Yemleme sistemi görüntüsü	48
Şekil 41 SRD-05VDC-SL-C 5V 4 Kanallı 5V Röle Modülü.....	49
Şekil 42 Raspberry pi 3 ile SRD-05VDC-SL-C 5V Bağlantı Devre Şeması	50

Şekil 43 Röle test kodları	51
Şekil 44 Röle kartı test işlemleri.....	52
Şekil 45 DS3231 Saat Modülü	53
Şekil 46 Raspberry Pi 3 ile DS3231 RTC kartı arasındaki bağlantı.....	53
Şekil 47 5050 BEYAZ LED ŞERİT'in Raspberry Pi 3 ile Bağlantısı.....	56
Şekil 48 Röle ve pi kullanarak beyaz led'i kontrol etmek için kod	57
Şekil 49 3528 SMD RGB Şerit LED.....	58
Şekil 50 Raspberry Pi 3 ile RGB LED bağlantısı.....	59
Şekil 51 pigpio kitaplığını başlatmak için gereken kod	60
Şekil 52 RGB Ledlerin çalışması için gereken kodlar	61
Şekil 53 RGB LED test işlemleri	62
Şekil 54 DC12V 5W Mini Fırçasız Su Pompası (Su Geçirmez).....	63
Şekil 55 Soğutucu Fan.....	63
Şekil 56 Aktüatörler arasındaki bağlantı	64
Şekil 57 Soğutucu fan ve su pompası test kodları	65
Şekil 58 Su sızıntı test devresi.....	66
Şekil 59 Su sızıntı sensörü test kodları	67
Şekil 60 Su sızıntı sensörü test işlemleri	68
Şekil 61 Logitech C920 Hd Pro Webcam	69
Şekil 62 Web kamerasının yerleşimi	70
Şekil 63 Akvaryum 'un UV4L canlı akış sunucusu kullanılarak çekilen resmi.....	71
Şekil 64 Thonny Python IDE ekran görüntüsü.....	73

Şekil 65 UV4L.....	76
Şekil 66 Buraya canlı video yayın resmi eklenecek	76
Şekil 67 Akvaryum kontrol sistemi arayüzü	77
Şekil 68 1 dakika süre ayarlanan otomatik mod ekran görüntüsü	79
Şekil 69 “1” dakikalık sürenin ardından çalışan otomatik yemleme sisteminin ekran çıktısı	80
Şekil 70 Akvaryum üzerinde bulunan yemleme sisteminin görüntüsü	80
Şekil 71 Sıcaklık durum butonu arayüz üzerindeki görünümü	81
Şekil 72 Arayüz üzerinde bulunan sıcaklık durum bilgisi bölümü	81
Şekil 73 Su seviye butonu arayüz üzerinde görünümü	82
Şekil 74 Su seviye durum ekran görüntüsü	82
Şekil 75 Su sızıntı durum butonu arayüz ekranında görünümü	83
Şekil 76 Su sızıntısı olmadığındaki ekrana getirilen uyarı mesajı.....	83
Şekil 77 RED, GREEN, BLUE led kontrol butonları	84
Şekil 78 RGB led akvaryum sistemi üzerinde görünümü	84
Şekil 79 Kameraya bağlanma ve ekran görüntüsü alma butonlarının arayüz üzerindeki görünümleri	85
Şekil 80 UV4L sunucusu üzerinde akvaryum görüntüsü	85
Şekil 81 Işığın açılış ve kapatma butonlarının arayüz üzerinde görünümleri	86
Şekil 82 Akvaryum içerisinde bulunan beyaz şerit led çalışma görünümü.....	86
Şekil 83 Akvaryum içerisinde bulunan beyaz şerit led çalışma görünümü.....	87
Şekil 84 Akvaryum içerisinde bulunan beyaz şerit led çalışma görünümü.....	87
Şekil 85 Su doldurma ve boşaltma pompalarının kontrol butonlarının görünümü	88

Şekil 86 Su boşaltma pompası kontrol ekran uyarı mesajı.....	88
Şekil 87 Su boşaltma pompası çalışırken kullanıcının durumu takip edebilmesi amacıyla ekranda gösterilen uyarı mesajı	89
Şekil 88 Su boşaltma pompasının kapatılması durumunda ekranda gösterilecek uyarı mesajı.....	89
Şekil 89 Su pompa sistem üzerinde görünümü	90
Şekil 90 Su doldurma pompası kontrol ekran uyarı mesajı.....	90
Şekil 91 Su doldurma pompası çalışırken kullanıcının durumu takip edebilmesi amacıyla ekranda gösterilen uyarı mesajı.....	90
Şekil 92 Su doldurma pompasının kapatılması durumunda ekranda gösterilecek uyarı mesajı.....	91
Şekil 93 Su doldurma pompasının kapatılması durumunda ekranda gösterilecek uyarı mesajı.....	91
Şekil 94 Yem at butonunun arayüz üzerinde görünümü	92
Şekil 95 Akvaryum kontrol sistemi üzerine yerleştirilen yemleme sistemi resmi	92
Şekil 96 Akvaryum kontrol sistemi üzerine yerleştirilen yemleme sistemi resmi	93
Şekil 97 Sisteme kaydet butonu arayüz görünümü	93
Şekil 98 Bilgiler kayıt edildi ekran görüntüsü.....	93
Şekil 99 Kaydedilen verilerin dosya içi görüntüsü.....	94
Şekil 100 Temizle butonu arayüz görünümü.....	94
Şekil 101 Verileri temizleme onay ekranı	94
Şekil 102 Veri temizleme ekranı	95
Şekil 103 Programı kapatma sorğu ekranı.....	95

1.GİRİŞ

1999'da MIT'deki Auto-ID Laboratuvarı'nın kurucusu olan İngiliz teknoloji öncüsü Kevin Ashton, İnternet'in RFID (Radyo) dahil her yerde bulunan sensörler aracılığıyla fiziksel dünyaya bağlandığı bir sistemi tanımlamak için " Nesnelerin İnterneti" terimini icat etti.

"Nesnelerin İnterneti" ifadesi için çeşitli tanımlar olmasına rağmen, en iyi yorum, fiziksel dünyanın İnternet'in sanal dünyası ile bütünleştirilmesi olacaktır. IoT, iç veya dış ortamlarıyla iletişim kurmak, algılamak veya etkileşim kurmak için gömülü teknoloji içeren bir fiziksel nesneler ağıdır. Yıllar boyunca, örneğin akıllı telefonlar, akıllı saatler, dizüstü bilgisayarlar, ev elektronik cihazları ve diğer elektronik izleme sistemi cihazları gibi IoT konseptine dayalı olarak oluşturulmuş birçok ürün oluşturulmuştur. Bu cihazlar arasında akvaryum izleme sistemi, günümüzde çok ilgi gören bir elektronik izleme sistemi olmuştur.

1.1 Akvaryum nedir?

Akvaryum, tatlı su veya deniz suyu organizmalarının bakımı için kap veya su organizmalarının bir koleksiyonunun sergilendiği veya üzerinde çalışıldığı bir tesis olarak tanımlanmaktadır.

İlk teşhir akvaryumu 1853'te İngiltere'de Regent's Park'ta açıldı. Dünyanın belli başlı şehirlerinin çoğunda artık ticari akvaryumların yanı sıra halka açık akvaryumlar da var; diğer akvaryum tesisleri esas olarak araştırma kurumları olarak hizmet vermektedir. Büyüklüğü ne olursa olsun – ister küçük bir galonluk kavanoz ister devasa bir milyon galonluk tank – akvaryumlar özenle inşa edilmelidir; İnsanlar için toksik olmayan birçok madde, özellikle plastikler ve yapıştırıcılar, su soluyan hayvanlar için toksiktir. Suda yaşayan organizmaların bakımı için birincil gereksinim su kalitesidir.

Daha sonra akvaryum izleme sistemi akvaryumun uygun şekilde bakımının yapılması amacıyla tanıtıldı. Akvaryum izleme sistemi gibi IOT cihazlarının yardımıyla kullanıcılar, akvaryumlarını ve balıklarını internet üzerinden her zaman ve her yerden izleyebilir ve yönetebilirler. Bu durum IOT sistemlerinin ne denli geliştiğine dair çarpıcı bir örnek oluşturmaktadır.

1.2 Motivasyon ve Sorun Bildirimi;

Günümüzde günlük harcamalar önemli ölçüde arttı ve bununla başa çıkmak için birçok insan sabahın erken saatlerinden akşam geç saatlere kadar işe gidiyor. Bu da özellikle

balık akvaryumu sahipleri için, ev içerisinde kontrollerinin azalmasına yol açtı. Balıklarını zamanında besleyemez ve akvaryumu sık sık temizleyemez hale gelen kullanıcılar için akvaryum izleme sistemleri geliştirilmiş ve bu projenin de ana motivasyon kaynağı bu durum olmuştur. Akvaryum sık sık temizlenemezse, balıklardan kaynaklanan atık ürünler suda birikerek suyu bulanıklaştırır. Bu da suyun hoş olmayan bir yeşilimsi hale gelmesine ve ayrıca çevreye hoş olmayan koku salgılamasına neden olur. Sudaki yüksek kimyasal içerik ve yanlış beslenme programı nedeniyle akvaryumdaki balıklar boğulabilir veya açıktan ölebilir. Bu nedenle, aşağıdaki özelliklere sahip bir Akvaryum izleme sistemi tasarılmıştır;

- Bir Akvaryum İzleme Sistemi, akvaryumun durumunu veri tabanına sürekli olarak aktarır ve kullanıcılar internet üzerinden onları anlık olarak izleyebilir.
- Bir Akvaryum İzleme Sistemi, akvaryum suyunun sıcaklığını, suyun bulanıklğını, su seviyesini ve akvaryumdan herhangi bir su sızıntısını tespit edebilir.
- Bir Akvaryum İzleme Sistemi, balıkları uygun zamanlarda, belirli sürelerle göre otomatik olarak besler veya kullanıcılar ayrıca internet üzerinden bir düğmeye tıklayarak balıkları besleyebilir.
- Bir Akvaryum İzleme Sistemi, uygun kararları verebilecek ve su çok bulanıksa akvaryum suyunu değiştirmek gibi aksiyonları gerçekleştirebilecektir.
- Bir Akvaryum İzleme Sistemi ayrıca alarmı tetikleyecek ve akvaryumdan herhangi bir su sızıntısı olduğunda kullanıcıları uyaracaktır.

Teknolojideki ilerlemeyle birlikte, günümüzde piyasada birçok akvaryum izleme sistemi mevcuttur. Ancak, mevcut sistemler, aşağıda listelendiği gibi birkaç işlev veya özellikten yoksundur:

- Bazı akvaryum izleme sistemleri uygun kararlar veremez.
- Bazı akvaryum izleme sistemleri taşınamazır değildir.
- Bazı akvaryum izleme sistemleri IoT tabanlı sistemler değildir.
- Bazı akvaryum izleme sistemleri kullanıcı dostu değildir.
- Bazı akvaryum izleme sistemlerinin maliyeti yüksektir.

1.3 Proje Kapsamı

Bu projenin temel amacı, akvaryumun durumunu her zaman izleyebilen ve kullanıcının arayüz üzerinden akvaryum üzerinde bulunan sensör ve aktüatörleri kontrol etmesini sağlayan bir sistem geliştirmektir. Bunun yanı sıra, sistemin taşınabilir olması, düşük maliyetli olması, düşük güç kullanımı, kullanışlı ve verimli olması gereklidir. Yukarıda bahsedilen sorunların üstesinden gelmek için bu proje için birkaç önlem uygulanmıştır:

- Bu sistemin kapsamı, 10-30 Litrelilik balık akvaryumu gibi küçük boyutlu akvaryumlarla sınırlıdır.
- Proje, Python dili olan Raspberry Pi kodlamasını işlemek için programlama becerisi gerektirir.

1.4 Proje Amacı

Bu projenin amaçları aşağıdaki gibidir:

- Kullanıcılar için güvenilir bir akvaryum izleme sistemi geliştirmek.
- Kullanıcıdan girdi alarak otomatik bir şekilde akvaryum içerisindeki balıkları besleyebilmek.
- Herhangi bir ileri teknik bilgiye ihtiyaç duymadan, teknik bilgisi olmayan kullanıcıların kolayca anlayabileceği ve çalıştırabileceğii bir akvaryum izleme sistemi geliştirmek.

1.5 Etki, Önem ve Katkı

Piyasada kullanıcılar için birçok akvaryum izleme sistemi mevcuttur, ancak hala daha da geliştirilebilecek bazı sınırlamalar vardır. Aygıtın yüksek güç tüketimi de endişe nedenidir. Birçok kullanıcı, benzer şekilde çalışan ancak daha fazla güç tüketen produktlere kıyasla daha düşük güç tüketimine sahip ürünleri seçmeyi tercih eder. Bu nedenle bu sistemi tasarlarken uygun bir mikrodenetleyici kullanmak çok önemlidir.

Bunun yanında akvaryum izleme sistemi de gerçek zamanlı olmalıdır. Bunun nedeni sistemin temel amacının, akvaryumun durumu hakkında kullanıcıyı güncellemek ve kullanıcıların alınan bilgilere göre uygun aksiyonları almasını sağlamaktır. Sistem gerçek zamanlı olarak çalışmıyorsa akvaryum, su kaçağı gibi insan müdahalesine ihtiyaç duyulan herhangi bir kritik durumdayken, sistem kullanıcıyı sorunun üstesinden gelmek için

zamanında uygun eylemler gerçekleştirmesi amacıyla uyarmalıdır. Bu nedenle, sistemin gerçek zaman tabanlı olması çok önemlidir.

Son olarak, akvaryum izleme sistemi, kullanıcıların sistemdeki tüm özelliklerin nasıl kullanılacağını kolay ve net bir şekilde anlayabileceği şekilde oluşturulmalıdır. Örneğin, akvaryum izleme sisteminin web sitesinin GUI'si (grafiksel kullanıcı müdahaleleri) teknik bilgisi olmayan kullanıcıların herhangi bir zorluk yaşamadan kullanabilmeleri için uygun ve kullanıcı dostu bir şekilde oluşturulmalıdır.

2. LİTERATÜR TARAMASI

Günümüzde IOT tabanlı akvaryum izleme ve kontrol sistem uygulamaları oldukça yaygındır. Bu alana dair literatürde bulunan güncel çalışmalarla örnek olarak;

Min-Chie Chiu tarafından geliştirilen “**Ağ Uzaktan Kumanda Sistemi kullanılarak Otomatik Termal Kontrol/Yem Besleme/su Aritma ile Donatılmış Çok Fonksiyonlu Akvaryum**” (Chiu, M.C., 2010.) projesi içeriğinde tüm sensörleri yönetmek için ana kontrolör olarak PC (kişisel bilgisayar) kullanılmaktadır. Sensörler farklı modül üzerinden PC kontrol sistemine bağlanmıştır. Modül, sensörlerden gelen analog sinyali dijital sinyale dönüştüren ADC'dir (Analognan Dijitale), böylece PC'nin değerleri okuyup sınırlandırılabilir. Ardından, VB arabirimini aracılığıyla, İstemci-PC sunucu-PC ile iletişim kurabilir ve sensörlerden gelen verilere dayalı olarak akvaryumu izler ve kontrol edebilir. Projenin gücü, kontrol olarak projede PC'nin kullanılmasıdır. PC sistem olarak yüksek işleme gücüne sahiptir işlem gücü sınırlı olan mikrodenetleyiciye kıyasla sensörlerden modüler aracılığıyla elde edilen verileri daha hızlı manipüle edebilir.

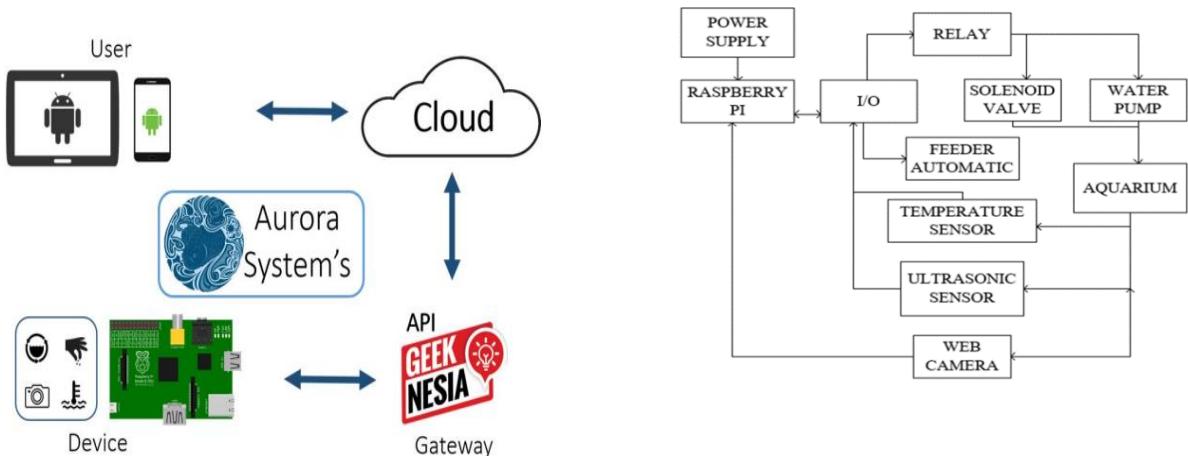
Ancak, bu üründe bazı zayıflıklar var. Bu zayıflıklarda ortaya çıkan ilk başta projenin maliyetidir. Ürün, tüm sistemi kontrol eden ve yöneten sistemin merkezi olarak bir PC'den oluşur. Sadece akvaryumu yönetmek için bütün bir PC'yi tahsis etmek gerçekten pahalıdır. İkincisi, ürünün taşınabilirliğidir. Bilgisayarın sistemin merkezi parçası olarak kullanılması, ürünün bir yerden başka bir yere taşınmasını zorlaştırmaktadır. Maliyet ve taşınabilirlik sorununun çözümü, sistemi çalıştıracak sistemin ana denetleyicisi olarak işlev görecek PC'yi değiştirmek için Raspberry pi adlı tek kartlı bir bilgisayar kullanılarak çözülebilir.

Bu proje ile ilgili bir diğer sistem ise (MZH Noor, AK Hussian, MF Saaid, MSAM Ali, M. Zolkapli, 2012) tarafından “**PIC mikrodenetleyici kullanılarak Otomatik Bahk**

Yemlik Sisteminin Tasarımı ve Geliştirilmesi” dir. Bu araştırma projesinde kullanılan mikrodenetleyici PIC16F886'dır. Geliştirilen sistem, balıkların beslenme aktivitesinin kontrolünde mekanik ve elektrik sistemini birleştirir. Bu cihaz temel olarak pelet yem depolama, kalıp, stand, DC motor ve mikrodenetleyiciden oluşmaktadır. Pelet yem deposunun altında bulunan DC motor tarafından kontrol edilir. Daha sonra bu cihaza bir kontrol sistemi eklendi ve balığın gerektiği gibi veya kullanıcı tarafından önceden tanımlandığı gibi doğru döngü zamanında beslenmesini sağlandı. Motor dönüşünü kontrol etmek için bu cihazda zamanlayıcı kullanıldı ve peletleri suya dağıtan küre oluşturucuya bu motor takıldı. Peletler, yalnızca motorun kendi dönüş hızına bağlı olarak havuzun işaretleme alanına dağıtılr. Projenin gücü, uygun çevrim süresi ile birlikte motorun dönüş hızına bağlı olarak arzu edilen alana pelet yemleri dağıtabilmesidir. Bu sistem rotor devresinde gerilmiş direnç kullanarak motor hızını kontrol etmenin geleneksel yolunun yerini alır veya elektrik makinesinin voltajını kullanılan PWM (Darbe genişlik modülasyonu) ile ayarlar. Bu sistemde bazı zayıflıklar mevcuttur. Bunlardan en önemlileri; mikrodenetleyici internete bağlı olmadığı için kullanıcı DC motor hızını uzaktan değiştiremez. Mikrodenetleyici, programda zamanda DC motor kullanarak peletleri istenen alana otomatik olarak dağıtır. Kullanıcı DC motor hızını veya program zamanını değiştirmek isterse, mikrodenetleyicinin bulunduğu yere manuel olarak gitmeli ve hex tuş takımını kullanarak değeri tuş ile değiştirmelidir. Bu, kullanıcının sistemin tam kontrolünü almasını kısıtlar. Bu, sistemin ana mikrodenetleyicisini kendi yapılandırılmış sunucusıyla düşük bütçeli Raspberry pi minibilgisayar aracılığıyla internete bağlayarak kullanıcının IP adresi aracılığıyla mikrodenetleyiciye bağlanabileceği ve DC motoru kontrol edebileceği şekilde çözülür.

Ayrıca, bu proje ile ilgili bir başka araştırma projesi de “**Android Tabanlı Uygulama Kullanan Otomatik Arowana Yükseltici Kontrol Cihazı**” (Nurliani Hidayah Ritonga; Agung Nugroho Jati; Rifki Wijaya 2016). Bu sistemin adı “AURORA” dır. (Arowana'nın dünyaca pahalı bir balık olduğu biliniyor ve çok fazla estetik yönü var. Arowana'nın ayrıca benzersiz bir yaşam alanı vardır. Bu sistem sayesinde, sahibi su sıcaklığını ve su sirkülasyonunu izleyebilir, bronzlaşma lambasını kontrol edebilir, servoyu besleme için kontrol edebilir ve akvaryumdan bir görüntü yakalayabilir. Veriler, uygulama ile donanım arasında ağ geçidi işlevi gören geeknesia kullanılarak bulut hizmetinden elde edilen

uygulama tarafından gönderilir ve alınır. Mikrodenetleyici olarak Raspberry Pi kullanan bu sistem ister internet'e bağlı, ister WIFI ister 3G bağlantı kullanıyor olsun uygulama bu sistemdeki bütün parçaları birbirine bağlayabilir. Ayrıca, bu AURORA sistemi büyük arowana sorunlarına çözüm olarak uygulanacak. Arowana hobisi olan sahibinin akvaryum içine yem vermesine ve su sirkülasyonunu kontrol etmesine yardımcı olacaktır. Bu projenin zayıf yanı, akvaryumlarının durumunu kontrol etmek için sensörlere ve akvaryum aktüatörünü izlemek ve kontrol etmek için android uygulamalarına sahip olmalarına rağmen, sistemin besleme için servo motoru kontrol etmek için insan müdahalesına ihtiyacı var. Akvaryum için ışığı açar veya kapatır ve akvaryum için su pompasını değiştirir. Sistem, akvaryumun sensör değerlerine göre kendi uygun kararı verme yeteneğine sahip değildir. Örneğin, sistem balıkları zamanında beslemeli veya su bulanıklaştığında insan müdahalesi olmadan otomatik olarak akvaryum suyunu değiştirmelidir. Bu özellik çok önemlidir, çünkü kullanıcı evinden uzakta uzun bir tatil için, kullanıcının izlemesi ve yapması pratik değildir.

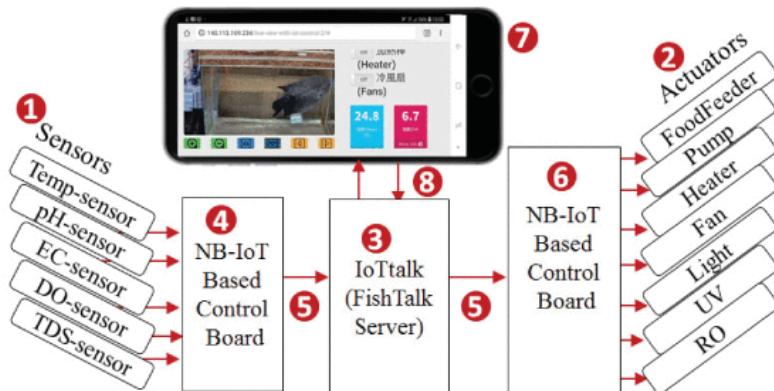


Şekil 1 AURORA'nın Mimari Tasarımı ve Kullanım Örneği Tasarımı

Bu proje ile ilgili diğer bir araştırma da (Taotao Xu; Feng Chen,2014) tarafından geliştirilen **“Gömülü Bulanık Karar Sistemi Su Ürünleri Yetiştiriciliği”** dir. Bu projede, Karar Sistemi tabanlı gömülü su kalitesi izleme platformu geliştirilmiştir. Basitçe, sensörlerden gelen verilere göre uygun karar alabilen bir sistem. Bu araştırmanın avantajı araştırmanın kendisidir. Uygun karar verebilen bir sistemdir. Bu tür bir sistem çok önemlidir çünkü insan su ürünleri yetiştirciliğini her zaman izleyemez. Ayrıca kritik

durumlarda insan panik nedeniyle doğru zamanda doğru karar veremeyebilir. Bu sistem ile kültür balıkçılığında birtakım değişiklikler olduğunda sistem bunları izleyebilir ve alabilir.

Bu projenin zayıf yanısı, sadece karar vermenin sonuçlarının GPRS veya mesajlar yoluyla yetişiricilere gönderilmesidir. Kullanıcılar, örneğin suyun sıcaklık değeri veya pH değeri gibi sensör değerleri hakkında bilgilendirilmeyecektir. Bunun yanı sıra, kullanıcı/çiftçi aynı zamanda sensör değerini gerçek zamanlı olarak göremez, örneğin bu sistem IoT tabanlı bir sistem olmadığı için internet üzerinden zaman alanı ile çizilen gerçek zamanlı bir grafik ile. Zaman alanıyla gerçek zamanlı grafik çizimi çok önemlidir, çünkü su ürünleri yetişiriciliğinde yanlış bir şey varsa, örneğin bazı balıklar belirli bir zamanda ölmüşse, kullanıcı/çiftçi bu grafiğin izini sürebilir. Sistem doğru zamanda doğru karar verebilse de onlar hala bir makinedir. Makine yanlış kararlar verebilir. Bu nedenle, her şeyin mükemmel şekilde çalışmasını sağlamak için insan müdahalesine de ihtiyaç vardır. Bu proje ile ilgili bir diğer sistem ise; (Lin, Y.-B., & Tseng, H.-C. 2019) tarafından geliştirilen **“FishTalk: Nesnelerin İnterneti Tabanlı Mini Akvaryum Sistemi”** dir. İçerisinde bulunan aktüatör ve sensörler sayesinde “FishTalk” sistemi balıkları uzaktan manuel bir biçimde beslemeye yardımcı olabilmektedir. Ayrıca üzerinde bulunan kamera sayesinde canlı olarak kullanıcıların akvaryumlarını anlık olarak takip edebilmesine olanak tanımaktadır.



Şekil 2 FishTalk için basitleştirilmiş bir blok diyagram.

Bu çalışma ile ilgili bir diğer proje ise (Pasha Mohd Daud, Ahmad Kamal; Sulaiman, Norakmar Arbain; Mohamad Yusof, Yuslinda Wati; Kassim, Murizah 2020.) Tarafından

geliştirilen “**Nesnelerin İnterneti Tabanlı Akıllı Akvaryum İzleme Sistemi**” dir. Bu araştırma balık yaşam habitatları için akvaryumda tatlı su tutma amacıyla IoT tabanlı Akıllı Akvaryum İzleme Sisteminin geliştirilmiş bir prototipini sunmaktadır. Bu sistem, balık besleme sistemi olarak çalışır ve çalışmasında bir akıllı telefon ile kontrol edilir. Arduino MEGA ve NodeMCU Tasarlanan sistemde kontrolörler kullanılmıştır. İşlemi kontrol etmek için akıllı telefon ve kontrolör arasında NodeMCU'daki Wi-Fi iletişim kullanılır. Analog pH sensörü, suyun pH değerini algılamak ve ekran için kullanılır. Değeri Liquid Crystal Display (LCD) aracılığıyla gösterir. Yazılım kısmı ise; Arduino Software IDE kullanılarak oluşturulurken, BLYNK yazılımı Android işletim sistemi için yazılım uygulamaları oluşturmak için kullanılır. Sistem, balık yaşam türüne uygun pH değerini izlemek ve Android uygulamada akıllı telefon kullanarak balık beslemesini kontrol etmek için tasarlanmıştır.

Bu çalışma ile ilgili yapılmış bir diğer proje ise; (Akila, I. S., Karthikeyan, P., Hari, H. M. V., & Hari, K. J. 2018). Tarafından gerçekleştirilen “**IoT Tabanlı Yerli Balık Besleyici**” dir. Önerilen sistemde, IoT (Nesnelerin İnterneti) ile entegre mekanik, elektrik ve iletişim bileşenlerinden oluşur. Mekanik kısım Raspberry tarafından kontrol edilen step motordan oluşur. Raspberry Pi B+, bakıcının tercihine göre gıda peletlerini içeren kabın web arayüzü aracılığıyla uygun rotasyonları tercih edip gıda peletlerini dağıtabilmektedir. Elektrik kısmı, bir Raspberry Pi B+ modülünden ve web arayüzü ve balıkların gerçek zamanlı video veri toplamasını sağlayan bir pi-kameradan oluşur. Web arayüzü kullanıcı tarafından sabitlenmiş programlama, yemleme verileri, canlı balık akışı, vb.'den oluşur. Web arayüzü ile uzaktan (manuel) veya önceden programlanmış besleme yoluyla iki besleme modu vardır. Bakıcı tarafından belirlenen süre içerisinde planlı besleme işleminde, bakıcının web sayfasındaki programı düzeltmesi gereklidir. Manuel beslemede kullanıcı, balığı web arayüzü aracılığıyla uzaktan besleyebilmektedir.

Bu proje ile ilgili yapılan bir diğer sistem ise, (Ali M. Jasim, Jafer Mohammed, Ahmed Ali, Ali Ibrahim, Mohammed J. Noori, Hazem M. Ali, 2021) tarafından yapılan “**Nesnelerin İnterneti (IoT) teknüğine dayalı bir minyatür balık çiftliği için bir kontrol ve izleme sistemi**” dir. Bu sistem, balıkları beslemek ve suyun Ph, sıcaklık ve kirliliğini kontrol etmek için geliştirilmiştir. Bu sistem, akvaryuma bırakılan yem miktarını kontrol eden otomatik bir balık besleme sistemi ile balıklar kablosuz olarak

uzaktan beslendiğinden kullanıcı çabasından, zamandan ve balık hayatından tasarruf sağlar. Bu sistemde, akvaryumda sırasıyla sıcaklık derecesi, Ph değeri ve su seviyesini ölçmek için üç adet sensör (sıcaklık, PH ve Ultrasonik sensör) bulunmaktadır. Bu bileşenler Wi-Fi mikrodenetleyici tarafından işlenmek üzere bir giriş birimine uygulanır. Çıkış olarak, için bir valf benimsenmiştir. Drenaj suyu, oksijen pompasını kontrol etmek için üç adet 5V röle, sensör verilerine göre su besleme pompası ve ısıtıcı, ayrıca bir servo motor ve sahadaki sistem parametrelerini izlemek için bir sıvı kristal ekran (LCD) kullanılmıştır. Ayrıca sistem cep telefonu uygulaması ile kontrol edilmekte ve izlenmektedir, burada algılanan veriler cep telefonu kullanıcılarına gönderilmektedir.

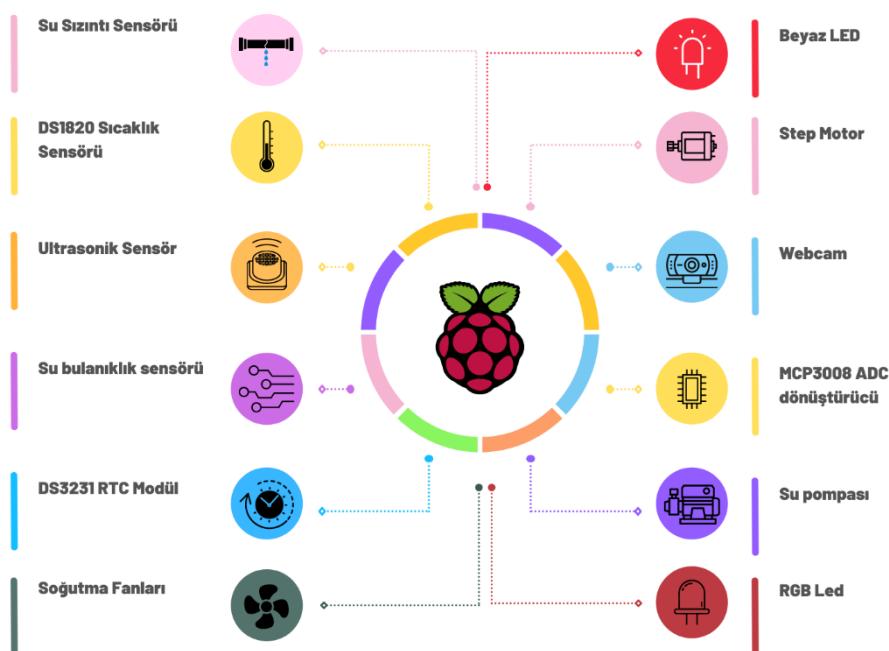
Bu çalışma ile ilgili bir diğer proje, (Saha, S., Rajib, R. H., & Kabir, S. (27-28 October 2018). Tarafından geliştirilen **“IoT Tabanlı Otomatik Balık Çiftliği Su Ürünleri İzleme Sistemi”** dir. Bu çalışmada, su kalitesinin izlenmesini Raspberry Pi, Arduino, çeşitli Sensörler, Akıllı Telefon Kamerası ve Android uygulaması ile ana hatlarıyla belirlenmiş ve gerçekleştirilmiştir. Bu çalışmada kullanılan su kalite parametreleri Sıcaklık, pH, Elektriksel İletkenlik ve Renktir. Sensör alımı Arduino tarafından yapılmakta ve Raspberry Pi, sunucunun yanı sıra veri işleme cihazı olarak da kullanılmaktadır. Suyun rengini algılamak için akıllı telefon kamerasının yardımıyla Raspberry Pi tarafından da fotoğraf çekimi gerçekleştirilmiştir. Terminal cihazı olarak Android telefon kullanılmıştır. Bir kullanıcı, bir android uygulaması kullanarak Wi-Fi aralığında ve dünyanın herhangi bir yerinden İnternet aracılığıyla su durumunu izleyebilir. Suyun genel yaklaşık durumunu ve gerekli eylemi belirlemek için dört parametre değeriyile bazı analizler yapılır. Ve kullanıcı akvaryum kontrolünü uzaktan gerçekleştirebilir. Yukarı da bahsedilen projeler verilebilir.

Projemizde, akvaryum sistemimizin izlenmesi için iki mod var, biri kullanıcı tarafından kontrol edilen manuel mod, diğeri ise sistemin kendisi tarafından kontrol edilen otomatik mod. Daha iyi bir Akvaryum İzleme Sistemi, enerji verimliliği, taşınabilirliği, güvenilirliği, kullanım kolaylığı ve düşük maliyetli, yüksek işlevselliğe dayalı olarak geliştirilmelidir.

3. MATERİYAL ve METOT

3.1 Sisteme Genel Bakış

Bu oturumda blok diyagram, aktivite diyagramı, akış şemaları, sistem resimleri gibi bazı diyagram ve çizelgeler kullanılarak akvaryum izleme sisteminin nasıl tasarılandığı ayrıntılı olarak tartışılmacaktır. Raspberry Pi 3b, bu akvaryum izleme sistemindeki sensörleri ve aktüatörleri izlemek ve kontrol etmek için ana kontrolör olarak kullanılmıştır. Pi'nin kontrol etmesi gereken yaklaşık dört sensör ve altı aktüatör bulunmaktadır. Aşağıda bu akvaryum izleme sisteminin blok şemasını ve resmi gösterilmiştir.

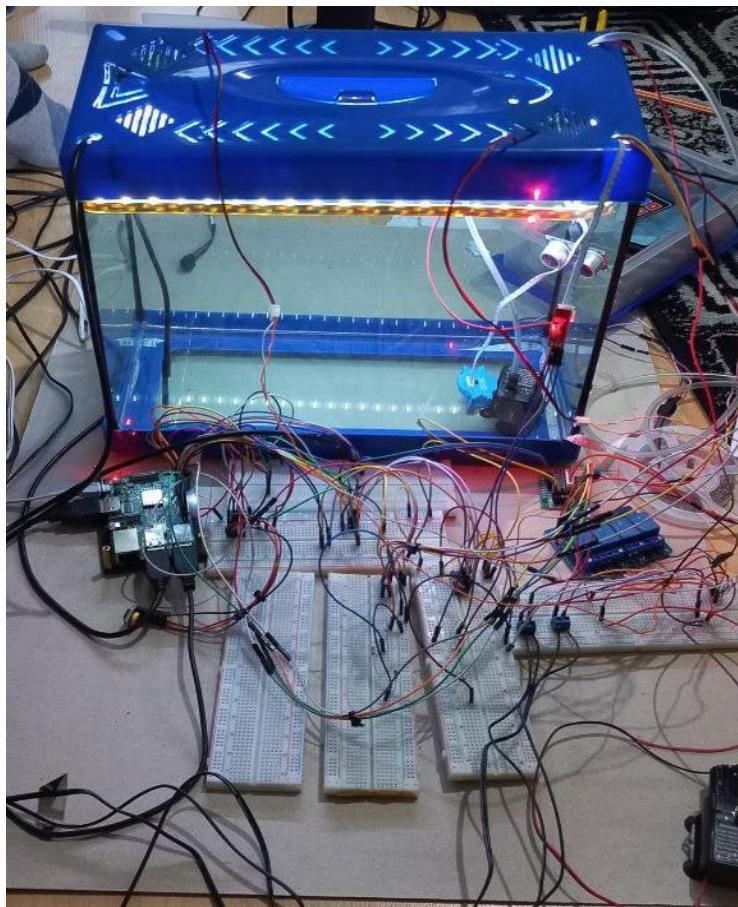


Şekil 3 Akvaryum monitör sisteminin genel blok diyagramı



Şekil 4 Akvaryum Kontrol Sistemi Kullanıcı Arayüzü (GUI)

Yukarıda verilen görselde kullanıcı arayüzü şeması gösterilmiştir. Python Tkinter arayüz tasarlama kütüphanesi kullanılarak hazırlanan bu arayüz oldukça sade ve anlaşılır bir şekilde tasarlanmıştır. Kullanıcının sistem üzerinde bulunan tüm sensörleri, ışıkları, aktuatörleri ve diğer modları uzaktan kontrol edebileceği bir arayüz tasarlanmıştır.



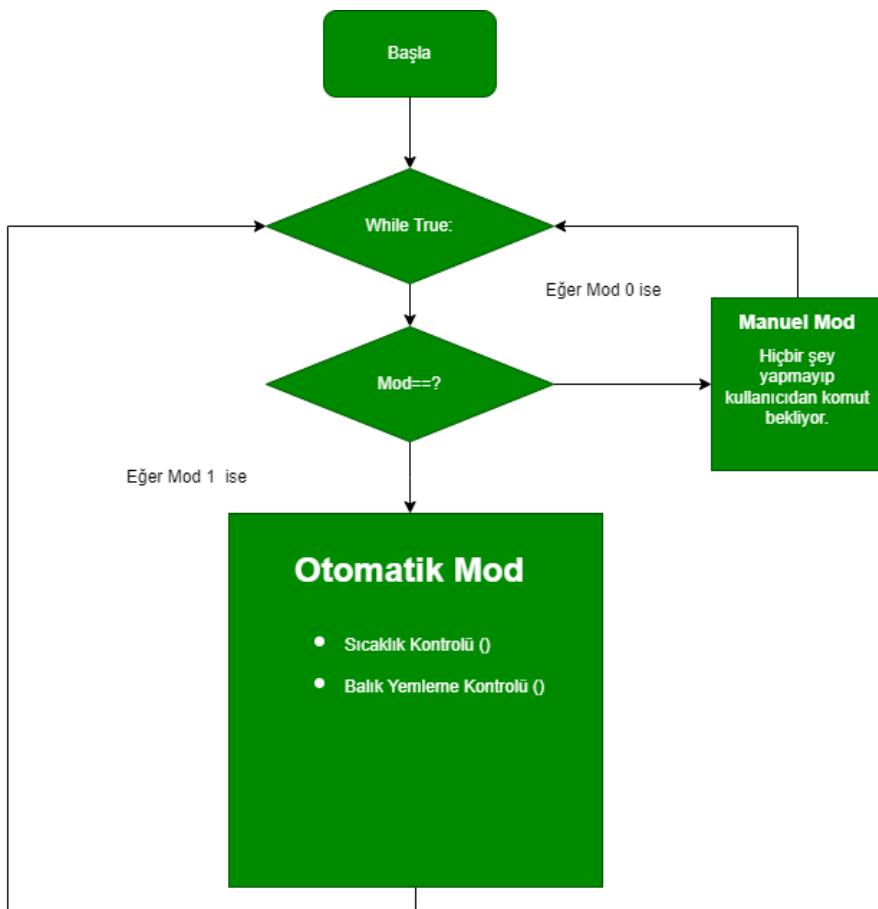
Şekil 5 Akvaryum izleme sisteminin elektronik bileşenlerin fiziksel bağlantıları

- | | |
|-----------------------------------|--------------------------------|
| 1. RGB Şerit LED
(Seviye) | 9. HC-SR04 Ultrasonik Sensör |
| 2. Soğutucu Fan | 10. Balık yemleme (Step Motor) |
| 3. Beyaz şerit LED | 11. 5V 4 Kanallı Röle |
| 4. Logitech C920 PRO Webcam | 12. RGB Kontrol devresi |
| 5. Akvaryum Su Pompası (x2) | 13. Raspberry Pi 3b |
| 6. DS18B20 su sıcaklık sensörü | 14. Sistem için ana devre |
| 7. 220V Su sızıntı sensörü | |
| 8. Turbidity (Bulanıklık) Sensörü | |

Projenin donanım ve yazılım bileşenleri, bunların spesifikasyon kurulumları ve bağlantıları, proje tezinin içerisinde ayrıntılı olarak verilmiştir.

3.1.1 Algoritma Tasarımı

Bu akvaryum izleme sistemi, biri manuel mod ve diğer otomatik mod olmak üzere iki moddan oluşur. Manuel mod, kullanıcı müdahalelerine ihtiyaç duyar. Örneğin, balığı beslemek için kullanıcının akvaryum kontrol sistem uygulamasını kullanarak Pi'ye manuel olarak komut göndermesi gereklidir. Otomatik modda ise Pi, sensöre dayalı akıllı kararlar verebilir. Örneğin, saat 21:00 ise balığı besleyin, çok kirliyse suyu değiştirin, bulanıklık sensör değerini alın vb. Yani yukarıda bahsedilen her şeyi yapmak için bir algoritmaya ihtiyaç vardır. Aşağıda bu sisteme kullanılan ana algoritma akış şeması ve otomatik mod sözde kodu gösterilmektedir.



Şekil 6 Otomatik mod için sistemin algoritma akış şeması

3.1.2 Sistemin Kontrol Modları

Kullanıcının bu akvaryum izleme sistemini kontrol etmesinin ve izlemesinin iki yolu vardır; biri Pi'de manuel mod diğeri ise yine Pi üzerinde çalışacak otomatik moddur. Bu bölümde, her bir kontrol modu tartışılmaktadır.

3.1.2.1 Arayüz Manuel Mod



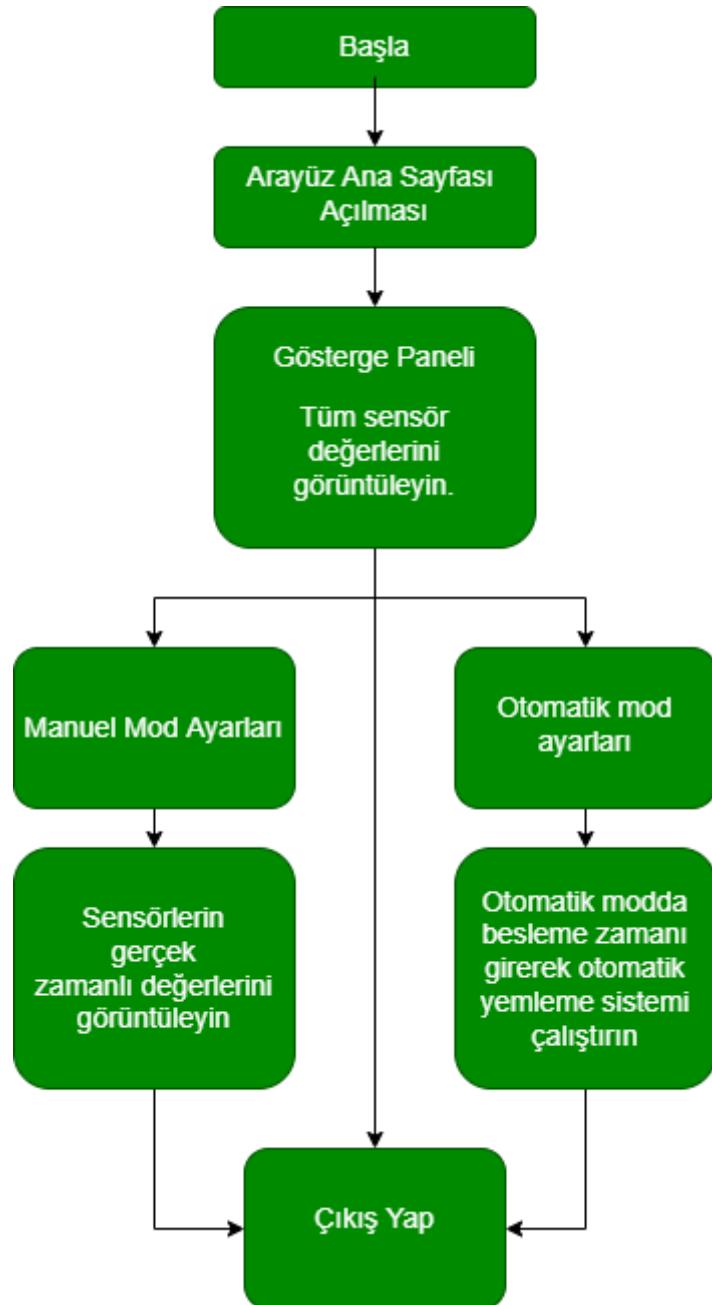
Şekil 7 Akvaryum Kontrol Sistemi Kullanıcı Arayüzü (GUI)

Yukarıda verilen arayüz kısaca tanımlanacak olursa; Otomatik mod butonu, kullanıcıdan dakika bilgisi alacak olan girdi kutusu, RGB Led kontrol butonları, yemleme butonu, kameraya bağlanma ve ekran görüntüsü alma butonu, sıcaklık durumu, su seviyesi, su sızcılık durumu gibi sensör butonları, beyaz led, soğutucu fan ve su pompa butonları ile sistemdeki bilgileri kaydetme, temizleme ve sistemi kapatma butonu yer almaktadır.



Şekil 8 Arayüz üzerinde manuel mod işlemlerinin gerçekleşme akış şeması

Yukarıdaki verilen akış şemasında sistem üzerinde kullanıcının manuel olarak gerçekleştireceği kontroller verilmiştir.



Şekil 9 Genel arayüz akış şeması

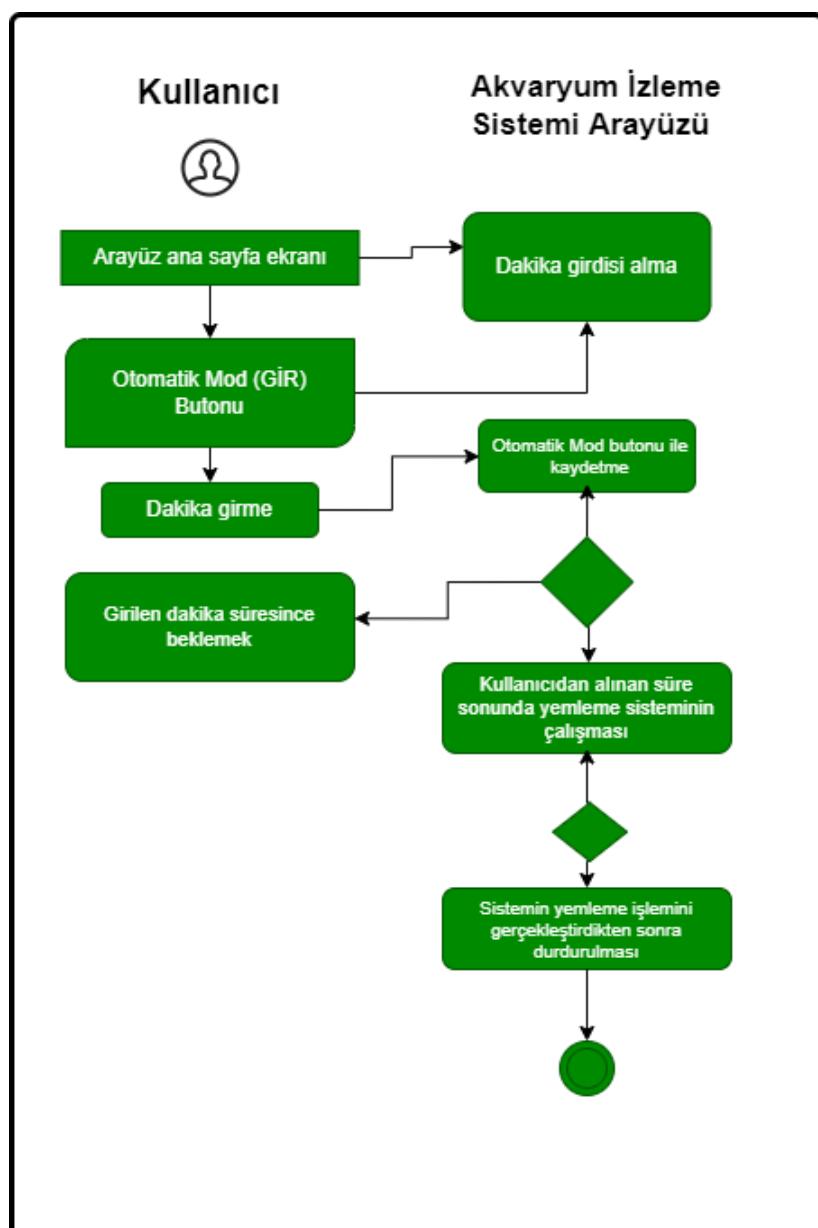
Yukarıda verilen akış şemasında kullanıcının arayüz üzerinde kontrol edebileceği modların genel şekli gösterilmiştir.

Sayılar	Buton İsmi	Fonksiyonlar
1	Sıcaklık Durumu	Su sıcaklığı akvaryum içerisinde bulunan sıcaklık sensöründen gelen verilerin ekran üzerinde gösterilmesi.
2	Su Seviyesi	Akvaryum iç kapakta bulunan ultrasonik sensörden gelen verinin ekran üzerinde gösterilmesi
3	Su sızıntı durumu	Su içerisinde bulunan su sızıntı sensöründen gelen verinin ekran üzerinde gösterilmesi
4	Kameraya bağlan	USB kameradan akvaryumun Canlı Akışını görüntüleyin.
5	Ekran görüntüsü al	USB kameradan akvaryumun canlı görüntüsünün kaydedilmesi.
6	Yem at	Kullanıcının manuel bir biçimde yemleme sistemini aktif hale getirmesi
7	Işığı aç	Akvaryum içerisine döşenmiş beyaz şerit ledlerin yanması.
	Işığı kapat	Akvaryum içerisine döşenmiş beyaz şerit ledlerin kapanması
8	Fanı aç	Akvaryum iç kapakta bulunan fanların çalıştırılması.
9	Fanı kapat	Akvaryum iç kapakta bulunan fanların kapatılması.
10	Su boşalt	Akvaryum içerisinde bulunan su pompasının çalıştırılarak akvaryum içerisinde bulunan suyu dışarı boşaltması.
11	1.pompayı kapat	Akvaryum içerisinde bulunan su pompasının durdurulması.
12	2.pompayı kapat	Akvaryum dışında bulunan su pompasının durdurulması.
13	Su doldur	Akvaryum dışında bulunan su pompasının çalıştırılarak akvaryum içerisine su doldurulması
14	RED, GREEN, BLUE	Akvaryum tabanına yerleştirilmiş olan kırmızı, yeşil ve mavi şerit ledlerin yanması.
15	RED, GREEN, BLUE (SİYAH)	Akvaryum tabanına yerleştirilmiş olan kırmızı, yeşil ve mavi şerit ledlerin kapatılması.
16	GİR	Kullanıcıdan yemleme zamanını almak için bir süre değeri istenir.
17	Otomatik Mod	Kullanıcının girdiği süre sonunda otomatik yemleme sisteminin çalıştırılması.
18	Sisteme kaydet	Sensörlerden alınan verilerin txt dosyasına kaydedilmesi.
19	Temizle	Ekranda bulunan sensör değerlerinin temizlenmesi

Tablo 1 Akvaryum fonksiyon açıklamaları

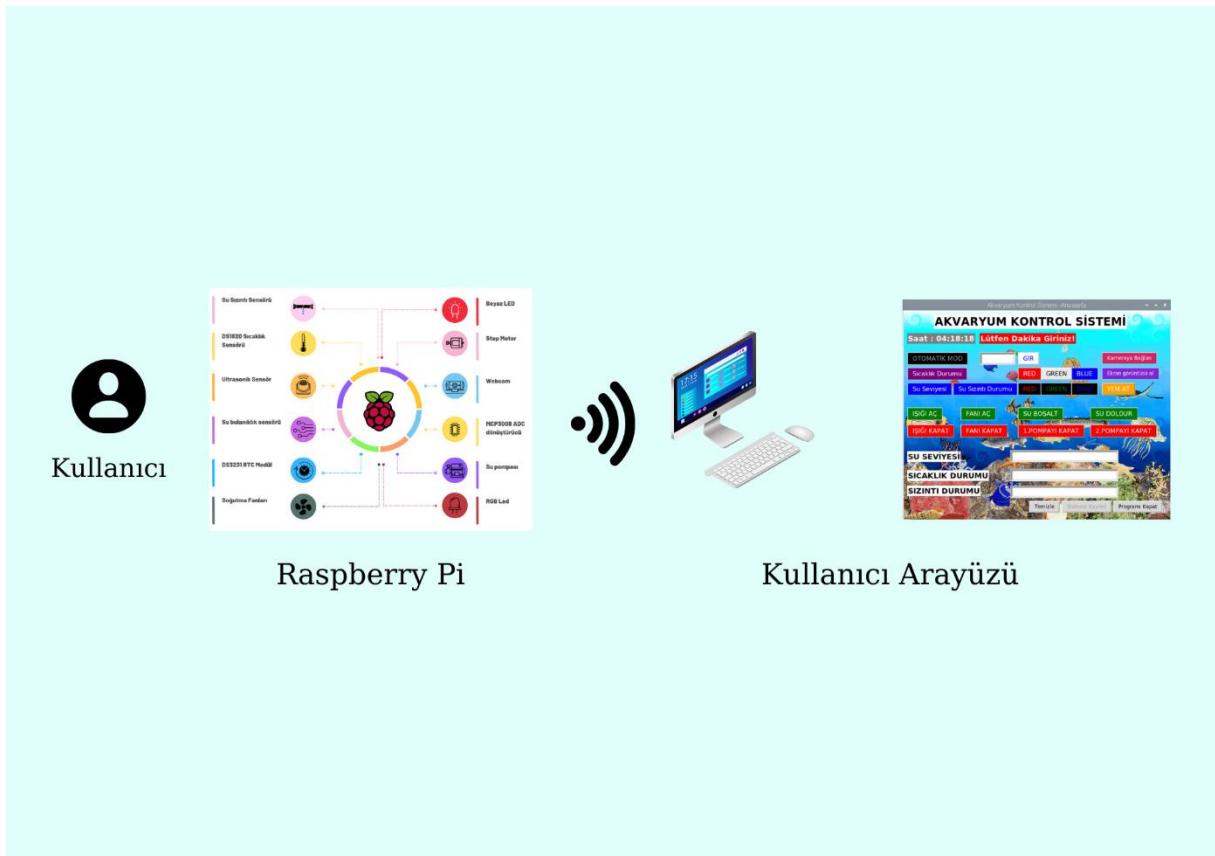
3.1.2.2 Arayüz otomatik mod

Arayüzde bulunan otomatik butonu sayesinde kullanıcı dışarıdan belirlemiş olduğu süre miktarını girerek otomatik yemleme sistemini çalıştırabilir. Bu mod sayesinde verilen değer sonunda akvaryum yemleme sistemini otomatik olarak çalıştıracak ve kullanıcının elle müdahale olmadan yemleme gerçekleştirilmiş olacaktır. Otomatik mod için akış şeması aşağıda verilmiştir.



Şekil 10 Kullanıcının Otomatik mod işlemlerini gerçekleştirmeye akış şeması

Yukarıda verilen akış şemasında kullanıcının akvaryum kontrol sistemi arayüzünde bulunan otomatik mod işlemlerinin sırasıyla gerçekleşmesi gösterilmiştir.



Şekil 11 Kullanıcı arayüzü ile sistemin genel blok şeması

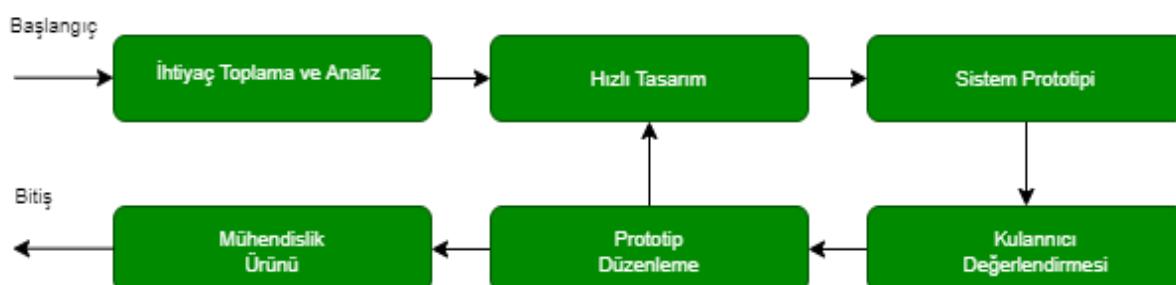
3.2 Tasarım Özellikleri

Yüksek işlevsellikte bir sistem üretmek için uygun bir tasarım metodolojisi seçmek çok önemlidir. Bunun öncesinde Yazılım Geliştirme Yaşam Döngüsü'nün (SDLC) ne olduğunu bilmek önemlidir. Yazılım Geliştirme Yaşam Döngüsü (SDLC), temel olarak bir uygulamanın veya sistemin geliştirilmesi ve yaşam döngüsü yönetimi için bir model sağlayan bir dizi adım veya fazdır. Bir SDLC sürecinin amacı, uygun maliyetli, etkili ve yüksek kaliteli bir ürün üretilmesine yardımcı olmaktadır. SDLC kapsamında pek çok model türü vardır, ancak bu projede projemizi modellemek için Prototipleme Modelini kullanılmıştır. Prototipleme modeli, mevcut gereksinimlere göre geliştirilmiştir. Prototip Modelleme Modelinin arkasındaki ana fikir, bu SDLC modelinde, tasarım veya kodlama aşaması devam

etmeden önce kullanıcının gereksinimlerinin dondurulmasıdır. Geçici prototip, müşterinin ürün gereksinimlerini anlamak için oluşturulmuştur ve bu geçici prototip, kullanıcının gereksinimine göre güncellenmeye devam edecektir. Bu model böyle bir yöntemi takip eder, böylece proje sonunda minimum arıza riski ile iyi bir fonksiyon sistemi üretilebilir.

Yazılım Geliştirme Yaşam Döngüsünün en önemli modellerinden biri olan Helezonik (Spiral) Model'dir. Projemizde kullandığımız prototipleme modelinin ana başlığı olarak bilinen Spiral yaklaşım, risk analizine öncelik verilmesi ve bir prototip geliştirilmesi bakımından diğerlerinden farklıdır. Risk analizine öncelik verildiğinden, hataları erkenden azaltma şansı verebilir. Prototipleme her aşamada gerçekleştiğinden, kullanıcı her aşamada yazılım projesinin bir bölümünü görme şansına sahip olur ve bu da zorlukların ortadan kaldırılmasına yardımcı olabilir. Dört temel aşamaya ayrılmıştır: Her aşamada, ara ürün için bir strateji geliştirilir.

Planlama: Her aşamada ara ürün için bir planlama yapılır. **Risk analizi:** Tehlikelerin araştırılmasını, tanımlanmasını ve çözülmesini gerektirir. **Üretim:** ara ürünü/ürünü yapma sürecidir. **Kullanıcı Değerlendirmesi:** Üretilen ara ürün sonucunda kullanıcılardan elde edilen girdilerin değerlendirilmesi ve bir sonraki adıma geçilmesi. Spiral modelin küçük ve düşük riskli projeler için çok pahalı bir yöntem olması, karmaşık bir yapıya sahip olması gibi birçok kusuru vardır. Uzun sürmesi ve evrak işleri gibi dezavantajları da eksik olarak bu konuda söylenebilir.



Şekil 12 Kullanıcı arayüzü ile sistemin genel blok şeması

Prototip Modelleme Modelinin farklı aşamalarını temsil eden aşamalar aşağıda verilmiştir:

- **Gereksinim toplama ve analizi:** Bu aşamada, temel kullanıcının gereksinimleri toplanır ve biriktirilir. Kullanıcı talebini tam olarak anladıktan sonra, ürün gereksinim belgeleri hazırlanır.
- **Hızlı Tasarım:** Çerçevenin altında yatan model iyileştirmedir, örneğin, temel ön koşul ve bu aşamada kullanıcı arayüzleri sağlanır. Bu hızlı tasarımında, kullanılan özellikler, orijinal anahatla tam olarak aynı şekilde çalışmayaacaktır. Ancak genel çalışma ortamı, nihai geliştirmeye daha çok benzemektedir.
- **Prototip oluşturma:** Prototip modeli, hızlı tasarımın çıktısından hızlı tasarımlı alır ve sistem tasarımını yerel olarak oluşturur. Bu yerel prototip, nihai ürünün tasarlanması gereği gibi bir görünüm ve benzer bir his verir.
- **Kullanıcı değerlendirmesi:** Sistemin prototipini oluşturuktan sonra, sistemin tüm işlevlerinin olup olmadığını kontrol etmek için prototip, değerlendirme için kullanıcıya gönderilir. Gereksinim olarak düzgün çalışır. Geliştiricinin ürünü daha da iyileştirebilmesi için kullanıcının geliştiricilere geri bildirimde bulunması gereklidir.
- **Ürünü iyileştirme:** Bu aşamada, kullanıcı geri bildirimi incelenir ve modelde herhangi bir sorun varsa geliştirici, hızlı tasarım aşamasına geri dönerek sorunu çözün ve kullanıcı gereksinimini karşılayın.
- **Mühendis ürünü:** Kullanıcıdan gelen tüm başarılı geri bildirimler ve olumlu incelemelerden sonra, asıl ürün tasarlanır ve geliştirilir.

Bu projede modelleme yöntemi olarak Prototip modeli seçmemizin ana nedeni, bu modelin esnekliği, sistemde çözülmesi gereken herhangi bir hata varsa önceki aşamaya geri dönebilme özelliğidir. Bu çok önemli çünkü birkaç denemede mükemmel bir sistem kurulması mümkün olmayacağındır. Sistemde/üründe hata olma ihtimali her zaman olacaktır ve modelin son aşaması olan Mühendis ürün aşamasına geçmeden önce bunu çözebilmeliyiz. Bunun yanı sıra, müşteri değerlendirme aşamasında geliştirici ve kullanıcı arasındaki etkileşim, geliştiricinin daha iyi kullanıcı dostu sistem veya ürün

oluşturmasına yardımcı olur çünkü kullanıcının sistem/ürün hakkındaki geri bildirimleri her zaman memnuniyetle karşılanır ve incelenir. Bir ürün geliştirirken kullanıcının gereksinimlerine önem vermek, sonuçta ürün için başarıyı getirecektir.

3.3 Donanım ve Yazılım

Proje tezinin bu bölümünde donanım ve yazılım bileşeni olmak üzere iki ana başlığa yer verilmiştir. Aşağıdaki Tablo 3.3'te, bu Akvaryum Kontrol Sisteminin oluşturulmasında kullanılan donanım bileşenlerinin ve yazılımların genel görünümünü göstermektedir.

Donanım Bileşenleri	Yazılım Bileşenleri
Raspberry Pi 3b	Python
MCP3008 Dönüştürücü	Python Tkinter GUI
HC-SR04 Ultrasonik Mesafe Sensörü	
Bulanıklık Sensörü	
DS18B20 Su Sıcaklık Sensörü	
28BYJ48 Step Motor ve ULN2003 Sürücü Kartı	
SRD-05VDC-SL-C 4 Kanal 5V Röle	UV4L Streaming Server
DS3231 Clock Modülü	
Beyaz Şerit Led	
3528 RGB Led	
3.5V~12V DC Fırçasız Su Pompası	
Soğutucu Fan	
Logitech webcam	

Tablo 2 Araçlar

Aşağıdaki bölümlerde, donanım ve yazılım için her bir bileşenin teknik özellikleri, kurulumu ve bağlantısı, kodları, birim doğrulaması ve testi tartışılmaktadır. Teknik özellikler bölümünde, bileşenlerin özellikleri hakkında bilgiler verilmiştir. Kurulum ve bağlantı bölümünde, bileşenlerin kurulum süreci ve ana kart olan Raspberry Pi ile fiziksel bağlantısı hakkında bilgiler verilmiştir. Kod bölümünde, her bileşenin kodu açıklanmış

ve birim doğrulama testinde her bileşenin beklenildiği gibi çalışıp çalışmadığı test edilmiştir.

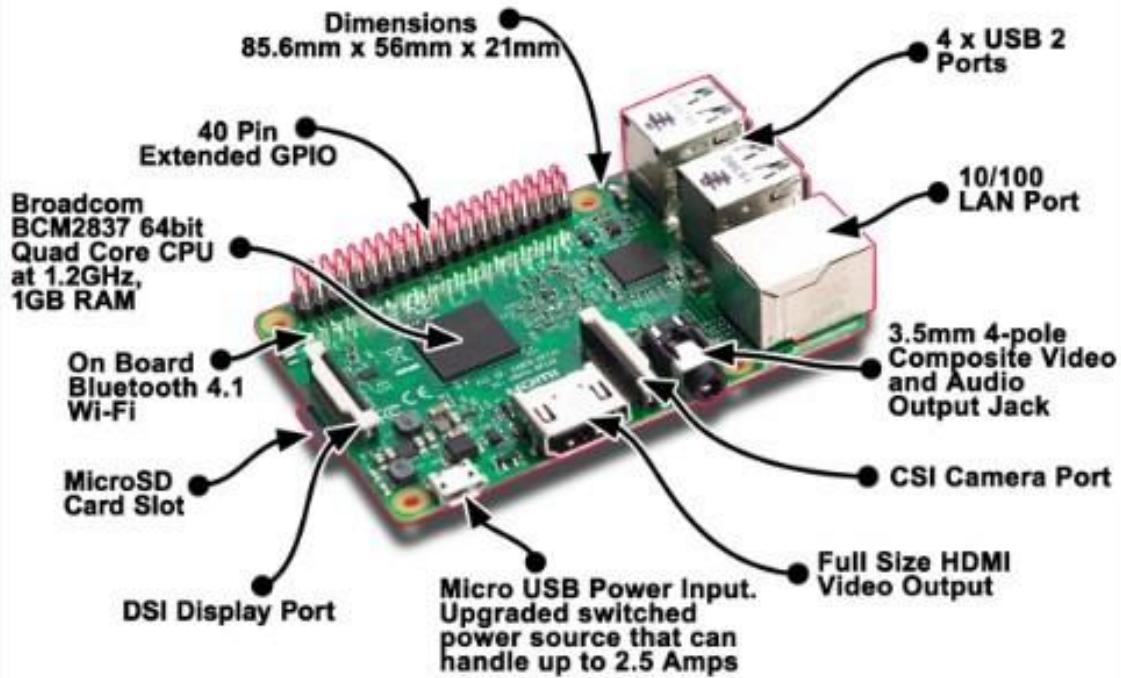
3.3.1 Donanım

3.3.1.1 Raspberry Pi Model 3B

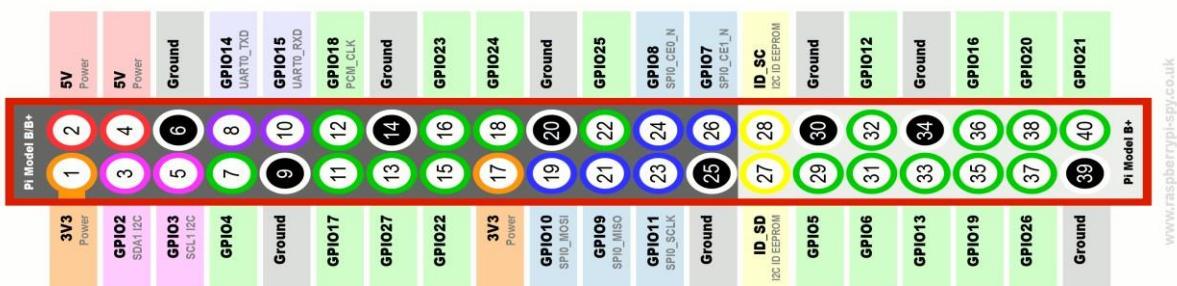
Bu projede merkezi kontrolör ve işlem birimi olarak kullanılan geliştirme kartı, Raspberry Pi 3 Model B adlı tek kartlı bir bilgisayardır. Kartın çalışması için bir İşletim Sistemine (OS) ihtiyaç duyulduğu için buna düşük maliyetli bir PC denir. Raspberry Pi 3'te kullanılan işlemci, 512KB paylaşımı L2 önbelleğe sahip 1.2 GHZ 64-bit dört çekirdekli ARM Cortex-A53 ile Broadcom BCM2837 SoC'dir. Bunun yanı sıra, Raspberry Pi 3 Model B, 40-pin Genişletilmiş GPIO'ya ve yerleşik WIFI ve Bluetooth'a sahiptir. Tablo 2, Raspberry Pi 3 Modelinin temel özelliklerini göstermektedir.

Mimari	ARMv8-A (64/32-bit)
SoC	Broadcom BCM2837
CPU	1.2 GHz 64-bit quad-core ARM Cortex-A53
Hafıza	1 GB
USB 2.0 Portları	4
Video Çıkışları	Ham LCD paneller için HDMI (rev 1.3), kompozit video (3,5 mm TRRS jakı), MIPI ekran arabirim (DSI)
Yerleşik Depolama	MicroSDHC yuvası, USB Önyükleme Modülü
Yerleşik Ağ	10/100 Mbit/s Ethernet, 802.11n wireless, Bluetooth 4.1
Pinler	24- GPIO pinleri 1- Seri UART'lar (RPi3 yalnızca mini UART'ı içerir) 2-SPI 1-I2C 2-5V Güç Pinleri 2-3.3V Güç Pinleri 8-Topraklama Pinleri
Güç Değerleri	800mA, 4 W
Güç Kaynağı	5V 2.5A MicroUSB
Boyut	85.60mm x 56.5mm x 17mm

Tablo 3 Raspberry Pi Model 3B özellikler tablosu



Şekil 13 Raspberry Pi Model 3B genel görünümü

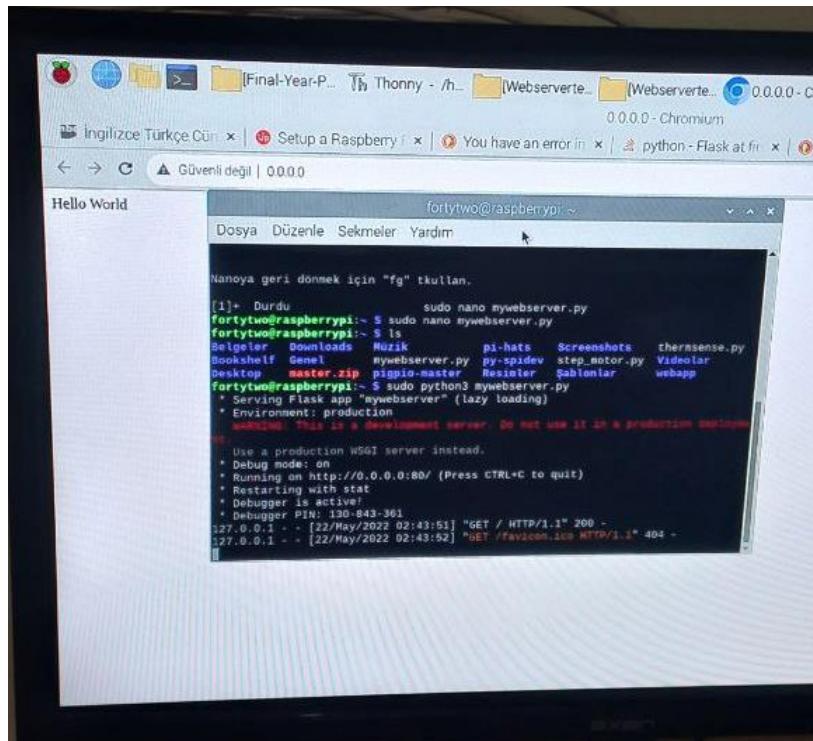


www.raspberrypi-spy.co.uk

Şekil 14 Raspberry Pi 3 Pinli Teknik Özellikler

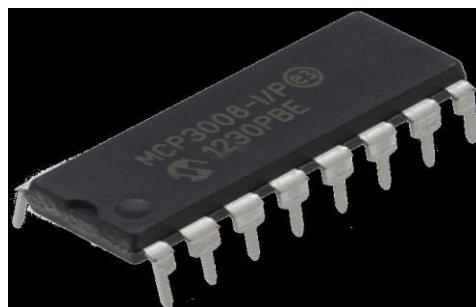
Raspberry Pi Model 3B kartının kurulum ve bağlantı aşamaları aşağıda verilmiştir. Başlangıçta, kutusundan çıkan Raspberry Pi 3'te herhangi bir işletim sistemi kurulumu bulunmuyor. Bu yüzden Rasbian işletim sistemi Raspberry Pi resmi web sitesinden indirilmiştir ve işletim sistemi görüntüsünü Raspberry Pi 3'ün Micro SD kartına Win32 Disk Imager kullanarak yazılmıştır. Ardından SD kart Raspberry Pi'ye yükleyip çalıştırılmıştır. İlk kurulum için harici bir klavye, fare ve HDMI monitör gereklidir.

Bunun nedeni Arduino'nun aksine Raspberry pi, sadece bir USB dizüstü bilgisayara takılarak kontrol edilemez. Raspberry Pi'nin çalışması için bir işletim sisteme ihtiyacı vardır. Tüm giriş ve çıkışları Raspberry pi'ye bağladıktan sonra, monitörde gösterilen Raspbian açılışını görebileceksiniz. Aşağıda Raspbian işletim sistemi kurulmuş olan kartımızın arayüz ekran resmi verilmiştir.



Şekil 15 Raspberry pi örnek ekran görüntüsü

3.3.1.2 MCP3008 ADC DÖNÜŞTÜRÜCÜ



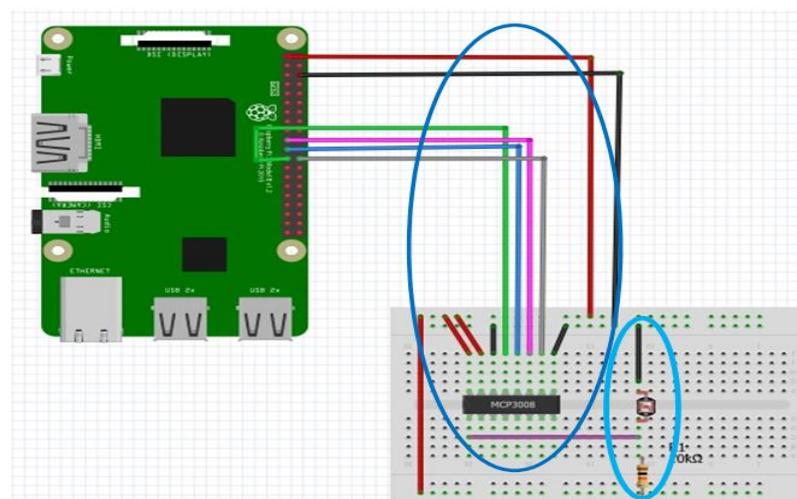
Şekil 16 MCP3008 ADC Dönüştürücü

Raspberry Pi 3 Model B, donanım açısından birçok özel yeteneğe sahip olmasına rağmen, entegre bir şekilde analog sinyali dijitalle dönüştürücü GPIO'ya sahip değildir. Dolayısıyla bu işlemi gerçekleştirmek için harici bir ADC dönüştürücüye ihtiyaç vardır. Bu projede, analog tabanlı sensörlerden gelen analog sinyali okumak ve Pi için dijital değerlere dönüştürmek için bir MCP3008 8-kanal 10bit ADC yongası kullanılmıştır. MCP3008, seri iletişim için SPI kullanır. Tablo 3.5 'te, MCP3008'in temel özelliklerini göstermektedir.

Çalışma Gerilimi	2.7-5.5 (V) aralığı	-
Çalışma Akımı	550 mA	Durum, VDD = VREF = 5V, DOUT yüksüz VDD = VREF = 2,7V, DOUT yüksüz
Çalışma Sıcaklığı	-40 ile +85(°C)	-
Yüksek Seviye Giriş Voltajı	0.7*V_dd	-
Yüksek Seviye Çıkış Voltajı	4.1 (V) Min	-

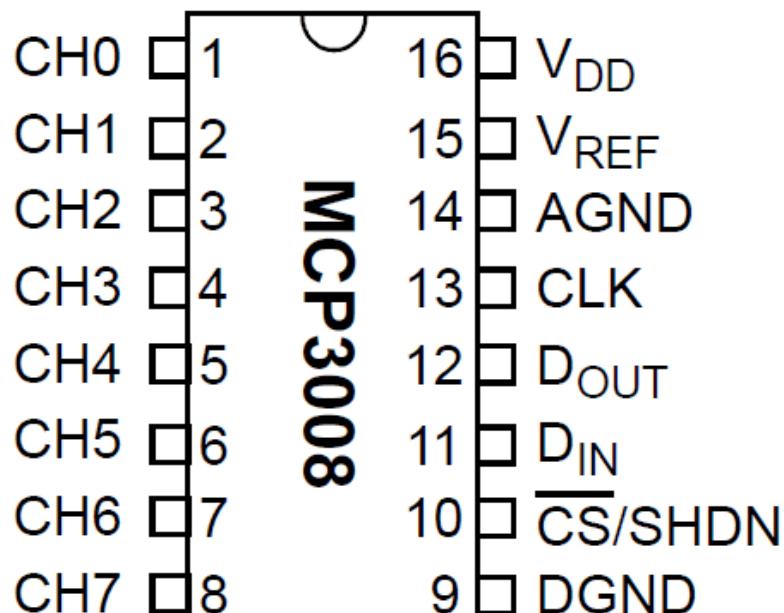
Tablo 4 MCP3008 Özellikleri

Aşağıda verilen Şekil 3.12'de fritzing devre şemasında Raspberry Pi kartı ile MCP3008 ADC mikroçipinin bağlantısı verilmiştir.



Şekil 17 MCP3008 ADC Dönüşürücünün LDR ve Raspberry Pi 3 ile Bağlantısı

Bir önceki bölümde bahsedildiği gibi, Raspberry pi'nin entegre bir ADC'si yoktur ve bu nedenle analog sensör değerlerini okumak için harici bir ADC dönüştürücü MCP3008'e ihtiyaç vardır. Bulanıklık sensörü veya LDR sensör Pi'den 3,3V kullanılarak açılır. MCP3008'in veri sayfasına göre, MCP3008 V_{dd} 'nin güç kaynağı, V_{ref} ($V_{dd} = V_{ref}$) ölçülecek voltaja eşit olmalıdır. Bu, MCP3008'in analog pinden (CH) okuyabileceği V_{ref} sinyalinin maksimum voltajının kendisinin V_{dd} değerinden daha yüksek olamayacağı anlamına gelir. Aşağıdaki şekil ve tabloda MCP3008 pin açıklaması, MCP3008 ve LDR sensörünün Pi'ye olan bağlantısı ve analog sensörü okumak için kullanılan kodu gösterilmiştir.



Şekil 18 MCP3008 pin diyagramı

Raspberry Pi 3B	MCP3008	LDR sensor
3.3V (Pin 1)	VDD and VREF (Pin 16 and 15)	Direnç ve LDR
Topraklama (Pin 6)	AGAND and DGND (Pin 14 and 9)	Topraklama Kablosu (Siyah Kablo)
GPIO11(Pin 23)	CLK (Pin 13)	-
GPIO09(Pin 21)	DOUT (Pin 12)	-

GPIO10(Pin 19)	DIN (Pin 11)	-
GPIO8(Pin 24)	CS/SHDN (Pin 10)	-
5V (Pin 2)	-	-
-	CH0 (Pin 1)	LDR verisi (Mavi Kablo)

Tablo 5 MCP3008, Raspberry Pi ve LDR sensörünün bağlantısı

```

File Edit Format Run Options Windows Help
import spidev
# Open SPI bus
spi = spidev.SpiDev()
spi.open(0,0)

# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
def ReadChannel(channel):
    adc = spi.xfer2([1,(8+channel)<<4,0])
    data = ((adc[1]&3) << 8) + adc[2]
    return data

```

Şekil 19 MCP 3008 adc dönüştürücünün test kodları

MCP3008 ADC dönüştürücünün düzgün çalışıp çalışmadığını test etmek için, Şekil 3.12'de gösterildiği gibi ikinci açık mavi daire ile çevrelenmiş 10k dirençli bir ışığa bağımlı direnç (LDR) eklenmiştir. LDR sensörü sadece MCP3008'i test etmek amacıyla kullanılmaktadır ve akvaryum izleme sistemine dahil değildir. Aşağıdaki şekil ve tablo, her bir koşulun test durumunu ve sonucunu göstermektedir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	LDR sensörünü elinle kapatmama	Okunan sensör değeri 100'den fazla olmalı ve python da "PARLAK" çıktısı vermelidir.	Sensör çıkışı gerçek sonuçla aynı olmalıdır.
2	LDR sensörünü elinle kapatma	Okunan sensör değeri 100'den az olmalı ve python da "KARANLIK" çıktısı vermelidir.	Sensör çıkışı gerçek sonuçla aynı olmalıdır.

Tablo 6 MCP3008 Test Senaryosu



```

File Edit Format Run Options Windows Help
import spidev
# Import time module
# Open SPI bus
spi = spidev.SpiDev()
spi.open(0,0)

# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
def ReadChannel(channel):
    adc = spi.xfer2([1,(8+channel)<<4,0])
    data = ((adc[1]&3) << 8) + adc[2]
    return data

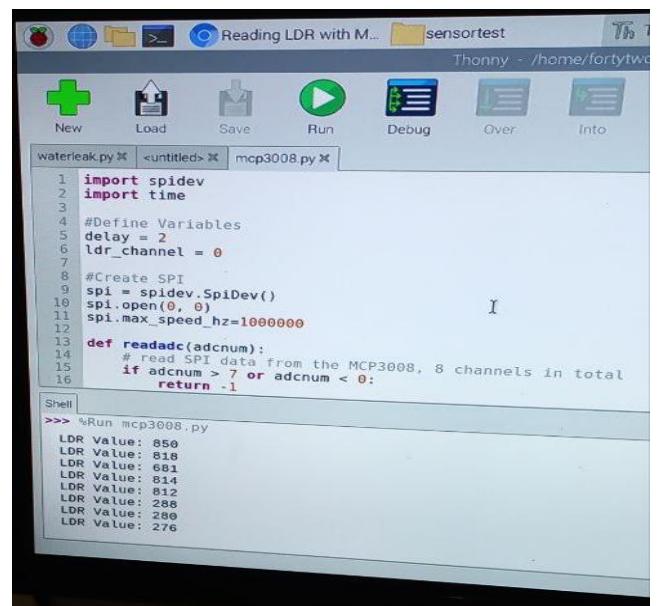
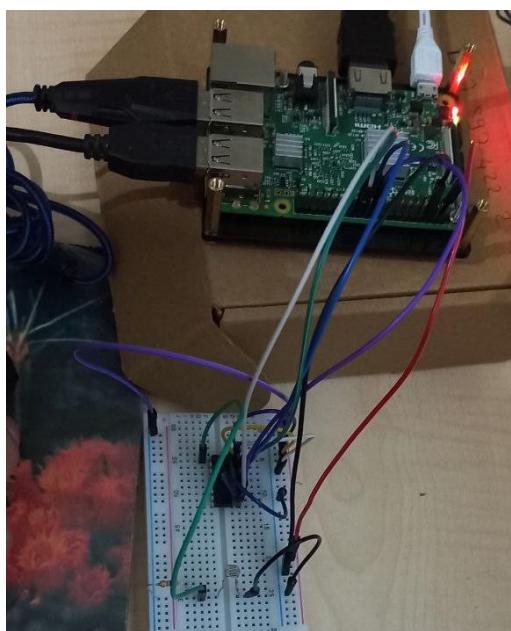
channel = 3

while True:
    LDR = ReadChannel(channel)
    if LDR <100:
        print "DARK"
    else:
        print "BRIGHT"

```

Ln: 23 Col: 0

Şekil 20.15 LDR sensörü ile MCP 3008 adc dönüştürücü test kodları



Şekil 21 LDR sensör test işlemleri

3.3.1.3 HC-SR04 (ULTRASONİK MESAFE SENSÖRÜ)



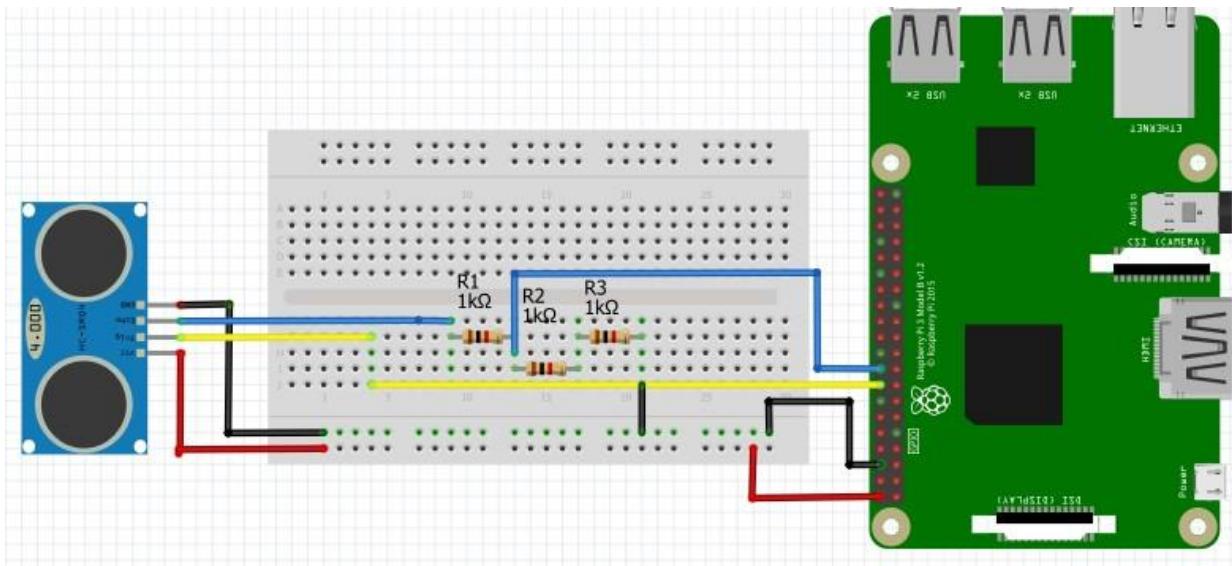
Şekil 22 HC-SR04 Ultrasonik Mesafe Sensörü

HC-SR04, bir nesneye olan mesafeyi yüksek doğruluk ve kararlı okumalarla ölçmek için kullanılan ultrasonik mesafe sensörüdür. Bir ultrasonik verici, bir alıcı ve kontrol devresinden oluşur. Verici, yakındaki herhangi bir katı nesneden yansılan yüksek frekanslı ses yayar ve sesin bir kısmı yansıtılır ve sensörün alıcısı tarafından algılanır. Yayma sinyali ve dönüş sinyali, aralarındaki farklı süreyi hesaplamak için kontrol devresi tarafından iletilecektir. Basit bir formül kullanmak yerine, sensör ile yansyan nesne arasındaki mesafe hesaplanacaktır. Bu projede akvaryumun su seviyesini ölçmek için HC-SR04 kullanılmıştır. Tablo 3.8'de, HC-SR04'ün temel özellikleri gösterilmiştir.

Çalışma Gerilimi	5.0 (V) DC
Çalışma Akımı	15(mA)
Ultrasonik Frekans	40 kHz
Maksimum Aralık	4 (m)
Minimum Aralık	2 cm
Ölçüm Açısı	15 degree
Tetik Giriş Sinyali	10uS TTL pulse

Tablo 7 HC-SR04 Genel Özellikleri

Aşağıda verilen şekil 3.17'de Ultrasonik sensör (HC-SR04) ve Raspberry Pi 3 bağlantı şeması fritzing programı üzerinden çizilerek gösterilmiştir.



Şekil 23 HC-SR04 ve Raspberry Pi bağlantısı

HC-SR04 sensöründe Vcc, Trig, Echo ve GND pini olmak üzere 4 adet pin bulunmaktadır. Burada hem mesafe hem de alınan zaman doğru orantılıdır çünkü daha fazla mesafe için geçen süre yüksektir. Tetik pimi $10 \mu\text{s}$ boyunca yüksek tutulursa, ses hızında hareket edecek ultrasonik dalgalar üretilecektir. Böylece, Echo pininde toplanacak sekiz döngü sonik patlama yaratır. Hava dolaşan $40\,000\text{ Hz}$ 'de bir ultrason yayar ve yolunda bir nesne veya engel varsa modüle geri döner. Yolculuk süresini ve sesin hızını göz önünde bulundurarak mesafeyi hesaplayabilirsiniz. Cısmın veya yansıyan darbe yoksa, Echo pini 38ms sonra zaman aşımına uğrar ve düşük duruma geri döner.

Yansıyan bir darbe alırsak, Echo (Yankı) pini 38 ms 'den daha erken inecektir. Yankı pininin yüksek olduğu süreye göre, ses dalgasının kat ettiği mesafeyi, dolayısıyla sensörden nesneye olan mesafeyi belirleyebiliriz. Bu amaçla mesafeyi hesaplamak için aşağıdaki temel formülü kullanıyoruz:

Mesafe = Hız x Zaman

Bu formülde hem hızı hem de zaman değerlerini biliyoruz. Süre, Echo pininin yüksek olduğu süredir ve hız, 340m/s olan ses hızıdır. Yapmamız gereken ek bir adım daha var ve bu da sonucu 2'ye bölmek, çünkü ses dalgasının nesneye gitmesi ve geri dönmesi için ihtiyaç duyduğu süreyi ölçüyoruz. Genel itibarıyle HC-SR04 ultrasonik mesafe sensörünün çalışma mantığı bu şekilde verilmiştir.

Aşağıdaki tablo HC-SR04 ve Raspberry Pi 3 arasındaki pin bağlantısını göstermektedir.

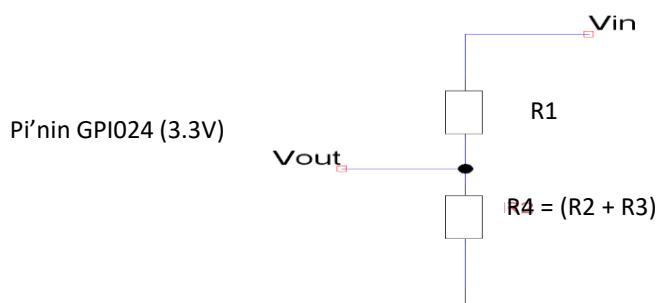
Raspberry Pi 3B Pinleri	Mesafe Sensörü Pinleri
5v pin (Pin2)	VCC
Topraklama (Pin 6)	GND
GPIO23 (Pin 16)	Trig
GPIO24 (Pin 18)	Echo

Tablo 8 Raspberry Pi ve Mesafe Sensörü pin bağlantıları

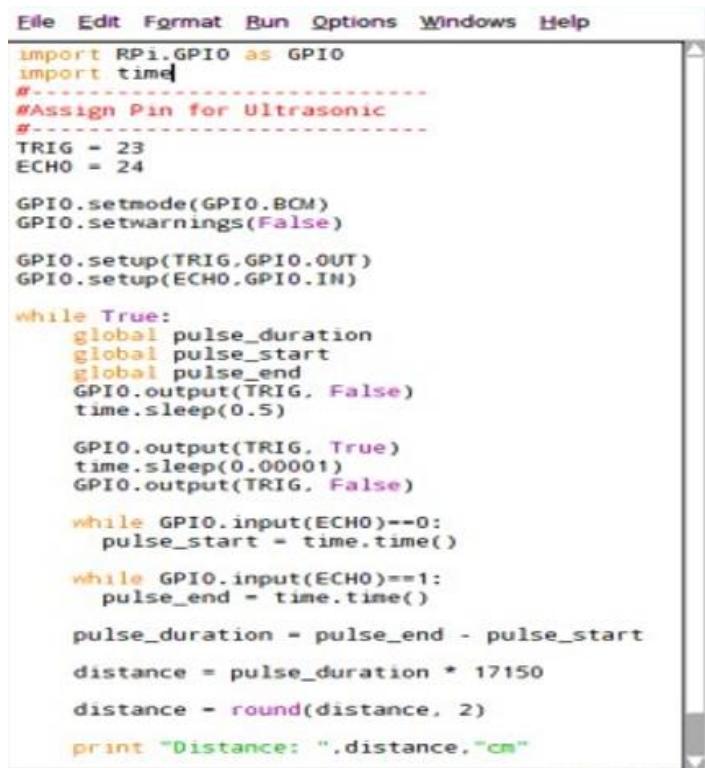
HC-SR04'ün Yankı çıkış sinyali 5V olarak derecelendirilmiştir. Ancak Pi'nin GPIO'1 yalnızca 3.3V'a kadar giriş voltajını kabul edebilir. Pi'nin korumasız 3.3V giriş portuna 5V sinyal gönderirsek, bu Pi'nin GPIO pinlerine zarar verebilir. Bu nedenle, voltaj bölücü kural devresini kullanarak, Pi'nin GPIO'1'na giriş yapıldığında HC-SR04'ün voltajını 5V'den 3.3V'a düşürebiliriz. İstenen gerilimi verecek uygun direnci bulmak için aşağıdaki denklem kullanılmıştır.

$$V_C = V_G \times \left(R_4 / (R_1 + R_4) \right)$$

1kΩ ve 2kΩ değerinde iki direnç (R1 ve R4 = (R2+R3)) kullanılır ve çıkış voltajına (V_C) düşürmesi gereken bir giriş voltajı (V_G) ile seri olarak bağlanır. V_G , 5v ile yankı olacak ve Şekil 5-F7'de gösterildiği gibi V_{out} 3.3V olacaktır. Gerilimi başarıyla düşürdükten sonra, şimdi Yankı voltajı Pi'nin GPIO'suna güvenlik girişi olabilir.



Şekil 24 Direnç bağlantısı



```

File Edit Format Run Options Windows Help
import RPi.GPIO as GPIO
import time
#-----
#Assign Pin for Ultrasonic
#-----
TRIG = 23
ECHO = 24

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

while True:
    global pulse_duration
    global pulse_start
    global pulse_end
    GPIO.output(TRIG, False)
    time.sleep(0.5)

    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)

    while GPIO.input(ECHO)==0:
        pulse_start = time.time()

    while GPIO.input(ECHO)==1:
        pulse_end = time.time()

    pulse_duration = pulse_end - pulse_start
    distance = pulse_duration * 17150
    distance = round(distance, 2)

    print "Distance: ",distance,"cm"

```

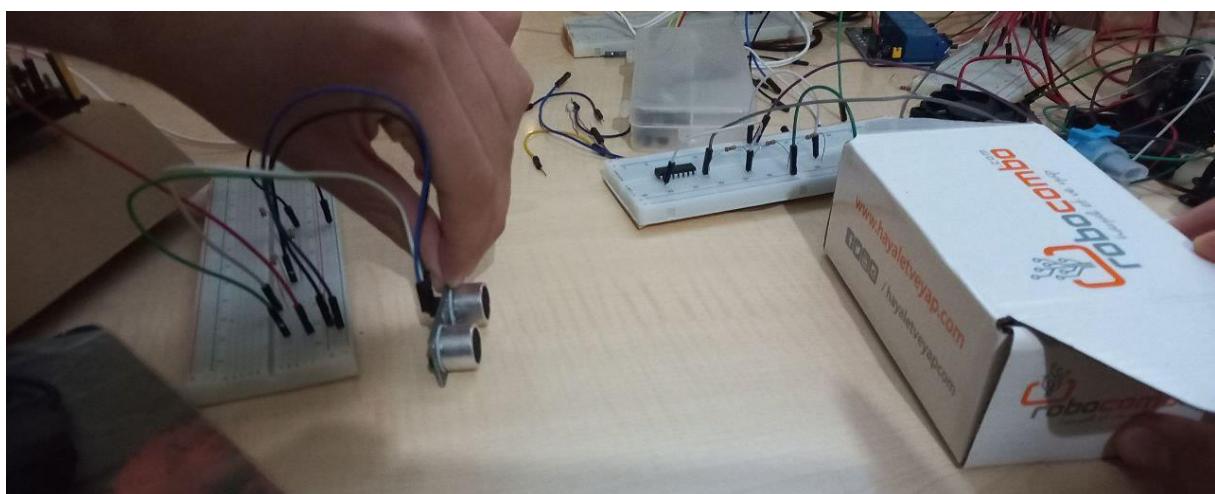
Şekil 25 HCSR-04 sensör kodları

Ultrasonik sensörün istediği gibi çalışıp çalışmadığını test etmek için birkaç test durumu tanımlanmıştır. Bir masa üzerinde bulunan kutunun sensöre belli aralıklarla yaklaştırılıp uzaklaştırılması mantığına dayanan bir dizi deneyler gerçekleştirilmiştir. Daha sonra, ultrasonik sensörün HDMI bağlantısı ile monitör üzerinden python kodlarını işaretli mesafeye birebir veya yaklaşık olarak aynı mesafe değerini verip vermeyeceğini HC-SR04 sensörü kullanılarak test edilmiştir. Aşağıdaki tablo test durumunu, gerçek sonucu ve testten elde edilen sonucu göstermektedir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	Kutuyu ilk satırdan 15cm uzağa yerleştirin (ultrasonik yerleştirildi).	Sensör, VNC görüntüleyicide python arayüzünde yaklaşık 15 cm ile aynı değeri vermelidir.	Sensör çıkışı, %1,8'lük yüzde hatasıyla gerçek sonuçla yaklaşık olarak aynı değerdedir.

2	Kutuyu ilk satırdan 25cm uzağa yerleştirin (ultrasonik yerleştirildi).	Sensör, VNC görüntüleyicide python arayüzünde yaklaşık 25 cm ile aynı değeri vermelidir.	Sensör çıkışı, %1,8'lük yüzde hatasıyla gerçek sonuçla yaklaşık olarak aynı değerdedir
---	--	--	--

Tablo 9 Test Sonuçları



```

Shell
Distance: 10.72 cm
Distance: 18.76 cm
Distance: 18.71 cm
Distance: 18.38 cm
Distance: 18.24 cm
Distance: 15.26 cm
Distance: 15.28 cm
Distance: 15.75 cm
Distance: 16.69 cm
Distance: 16.69 cm

```

Şekil 26 Ultrasonik sensör deney işlemleri

Bu akvaryum izleme sistemi projesinde akvaryumun su seviyesini ölçmek için ultrasonik mesafe sensörü kullanılmaktadır. Akvaryumun su seviyesini ölçmek için öncelikle akvaryumumuzun yüksekliğine ihtiyacımız var. Bu nedenle, arayüz'de, kullanıcının diğer şeyleri ayarlamadan önce akvaryumunun boyutunu (Uzunluk x Genişlik x Yükseklik) cm cinsinden girmesi gereklidir. **Bu projede 18.8cm yüksekliğinde bir akvaryum kullandım.** Aşağıda, HC-SR04'ün akvaryumun su seviyesini nasıl ölçüyüne ilişkin basit algoritma gösterilmektedir.

AquariumHeight = 18.8 (Defined by user)

OptimumWaterLevel = $18.8/(100/18) = 3.384\text{cm}$

HalfWaterLevel = $18.8/(100/50) = 9.4\text{ cm}$

Ultrasonik mesafe değeri yaklaşık 3.384cm ise Pi, tankın %82'sinin dolmuş olduğu ve akvaryumun su seviyesinin optimum seviyede olduğu sonucuna varacaktır. Oysa ultrasonik sensör mesafe değeri yaklaşık 9,4 cm ise Pi, akvaryumdaki suyun sadece %50'sinin kaldığı sonucuna varacaktır. Yarı su seviyesinden daha düşük veya optimum su seviyesinden daha fazla olan herhangi bir şey, sistem otomatik moddaysa su seviyesini korumak için su pompasını suyu içeri veya dışarı pompalamak için tetikleyecektir.

3.3.1.4 TURBIDITY (SU BULANIKLIK) SENSÖRÜ



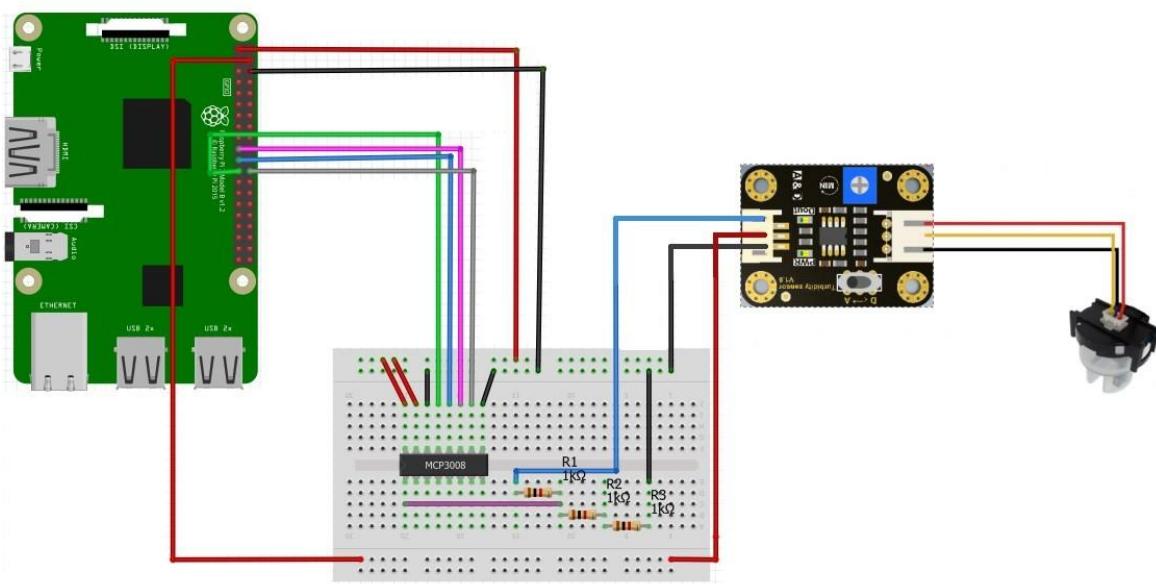
Şekil 27 Turbidity su bulanıklık sensörü

Turbidity bulanıklık sensörü, suyun bulanıklığına göre suyun kalitesini ölçer. Işık geçirgenliği ve saçılma oranını ölçerek suda asılı kalan partikülleri tespit edebilir. Bu saçılma hızı, sudaki toplam askıda katı madde (TSS) miktarı ile değişecek ve sudaki toplam askıda katı madde miktarı arttıkça bulanıklık seviyesi de düşecektir. Yani bulanıklık değerinin TTS ile ters orantılı olduğunu söyleyebiliriz. Bulanıklık sensörü, iki çıkış modu analog ve dijital sinyal sağlayan adaptör ile birlikte gelir. Tablo 5-T8, Bulanıklık sensörünün temel özelliklerini gösterir.

Çalışma Gerilimi	5.0 (V) DC
Çalışma Akımı	40(mA)max
Tepki Süresi	<500ms
Çıkış Methodu	<ul style="list-style-type: none"> Analog çıkış: 0-4.5V Dijital Çıkış: Yüksek/Düşük seviye sinyali (Potansiyometre kullanılarak ayarlanabilir)
Çalışma Sıcaklık Aralığı	5°C~90°C
Adaptör Boyutları	38mm*28mm*10mm

Tablo 10 Turbidity sensör özellikleri

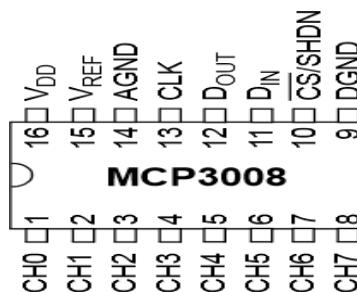
Bu projenin test aşamalarında en çok zorlandığımız sensör olarak turbidity sensörü verilebilir. Devre kurulumu ve test işlemleri gerçekleştirirken gerçek sonuç ve deney sonucu arasındaki ciddi farkları düzeltmek için bu sensör birçok defa teste tabi tutulmuştur. Aşağıda turbidity sensörün Raspberry Pi ve MCP3008 ADC konvertörü ile bağlantısı fritzing uygulaması üzerinde çizilmiş devre şeması verilmiştir.



Şekil 28 Bulanıklık sensörünün MCP3008 ADC dönüştürücü ve Raspberry Pi 3 ile bağlantısı

Daha önceki bölümde tartışıldığı gibi, bulanıklık sensörü, veri hattına (Mavi kablo) 5V çalışma voltajı ile analog değer vererek suyun bulanıklığını ölçmek için kullanılır. MCP3008 yongası, Pi'nin entegre ADC dönüştürücüsü olmadığından, bulanıklık

sensöründen gelen analog sinyali Pi'ye dönüştürmek için kullanılır. MCP3008, Pi'den 3,3V kullanılarak açılır. MCP3008'in veri sayfasına göre, MCP3008 V_{dd} 'nin güç kaynağı, V_{ref} ölçülecek voltaja eşit olmalıdır ($V_{dd} = V_{ref}$). Bu, MCP3008'in analog pinden (CH) okuyabileceğim V_{ref} sinyalinin maksimum voltajının kendisinin V_{dd} değerinden daha yüksek olamayacağı anlamına gelir. $V_{ref} > V_{dd}$ ise çipe zarar verebilir. Sorunu çözmek için, MCP3008'in V_{dd} ve V_{ref} 'ini Pi'nin 3.3V'sine bağladık ve bulanıklık sensörünü Pi'nin 5V pinile çalıştık. Daha sonra bulanıklık sensörü çıkışını (Mavi kablo), çıkış voltajını 5V'tan 3.3V'a düşüren 3 dirençli (Ultrasonik sensörün Yankı hattı kurulumuyla aynı) bir voltaj bölücü devresine bağladık. Bu yöntemi kullanarak suyun bulanıklığı MCP3008 ve Pi'ye zarar vermeden ölçülebilmiştir. Aşağıdaki şekil ve tablo, MCP3008 pin açıklamasını ve MCP3008 ile Bulanıklık sensörünün Pi'ye pin bağlantısını göstermektedir.



Şekil 29 MCP3008 Pin diyagramı

Raspberry Pi 3B	MCP3008	LDR sensor
3.3V (Pin 1)	VDD and VREF (Pin 16 and 15)	-
Topraklama (Pin 6)	AGAND and DGND (Pin 14 and 9)	Topraklama Kablosu (Siyah Kablo)
GPIO11(Pin 23)	CLK (Pin 13)	-
GPIO09(Pin 21)	DOUT (Pin 12)	-
GPIO10(Pin 19)	DIN (Pin 11)	-
GPIO8(Pin 24)	CS/SHDN (Pin 10)	-
5V (Pin 2)	-	Güç Kablosu (Kırmızı Kablo)
-	CH0 (Pin 1)	Data Kablosu (Mavi Kablo)

Tablo 11 Bulanıklı sensörü pin tanımlamaları

```

import spidev
import time
#
#-----#
#To read Analog Sensors Turbidity
#-----#
# Open SPI bus
spi = spidev.SpiDev()
spi.open(0,0)

# Function to read SPI data from MCP3008 chip
# Channel must be an integer 0-7
def ReadChannel(channel):
    adc = spi.xfer2([1,(8+channel)<<4,0])
    data = ((adc[1]&3) << 8) + adc[2]
    return data

while True:
    tur = ReadChannel(channel)
    if tur<100:
        print "The solution is very dirty"
    elif tur >280:
        print "The solution is clean"
    elif tur <270 and tur >100:
        print "Sensor out of water"
    time.sleep(1)

```

Şekil 30 Buraya bulanıklık sensör kod ekranı eklenecek.

Bulanıklık sensörünün düzgün çalışıp çalışmadığını test etmek için sırasıyla temiz su ve kirli su (kahve ile karıştırılmış su) ile iki bardak çözelti hazırlanmıştır. Ardından onları temiz ve kirli su ile etiketleyip bulanıklık sensörünü yerleştirdiğinde bulanıklık sensörünün hangi suyun temiz olup olmadığını tespit edip edemediği değerlendirilmiştir. Bu, basit (**if-if**) deyimi ile gerçekleştirilmiştir. Bulanıklık değeri 100'den küçükse kirli, 280'den büyükse temiz su, değer 100 ile 270 arasında ise sensörde su yok sayılır. Aşağıdaki tablo 3.13'de test durumu, gerçek sonucu ve testten elde edilen sonuçlar verilmiştir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	Bulanıklık sensörü değerini herhangi bir çözüme yerleştirilmedeinde.	Sensör, 100 ile 270 arasında bir değer vermelidir ve Thoony Python IDE'de "Sensör sudan çıktı" çıktısını vermelidir.	Sensör, gerçek sonuçla yaklaşık olarak aynı değeri verir ve "Sensör sudan çıktı" çıktısını vermiştir.
2	Bulanıklık sensörü değerini temiz suya yerleştirildiğinde.	Sensör, 280'den büyük bir değer vermelidir ve Thoony Python IDE'de " Temiz" çıktısını vermelidir.	Sensör, gerçek sonuçla yaklaşık olarak aynı değeri verir ve "Sensör temiz" çıktısını vermiştir.
3	Bulanıklık sensörü değerini kirli suya yerleştirildiğinde.	Sensör çıkış değeri 100'den az olmalıdır ve Thoony Python IDE'de "Kirli" çıktısını vermelidir.	Sensör, gerçek sonuçla yaklaşık olarak aynı değeri verir ve "Sensör kirli" çıktısını vermiştir.

Tablo 12 Bulanıklık Sensörü Test Senaryosu

Bu projede akvaryum suyunun kirli olup olmadığını tespit etmek ve kullanıcıya bildirmek için bulanıklık sensörü kullanılmaktadır.

3.3.1.5 DS18B20 (SU GEÇİRMEZ SICAKLIK SENSÖRÜ)



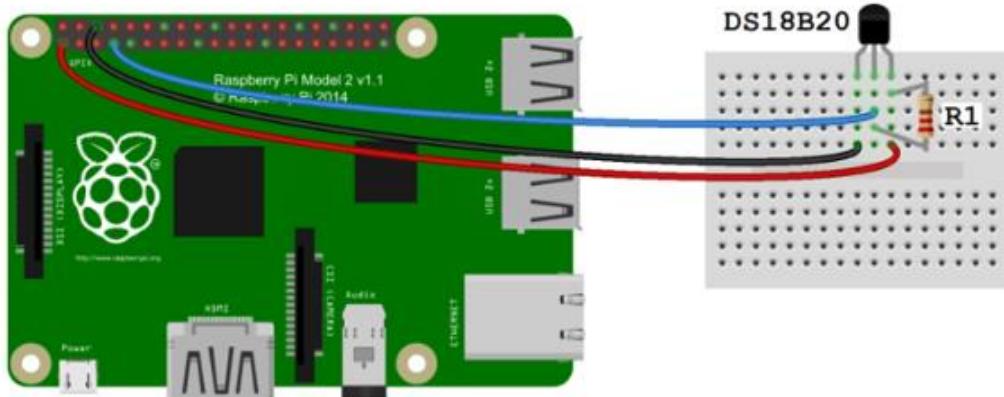
Şekil 31 DS18B20 Su Geçirmez Sıcaklık Sensörü

DS18B20, Sıvı ortamın sıcaklığını sadece basit bir One-Wire arayüzü kullanarak doğru bir şekilde ölçen dijital bir sıcaklık sensörüdür. One-Wire arayüz üzerinden 9 ila 12 bit sıcaklık okuması sağlayabilir. Dijital olduğu için uzun mesafelerde bile sinyal kaybı veya bozulması açısından sorun teşkil etmeyecektir. Tablo 3.14, DS18B20 dijital sıcaklık sensörünün temel özelliklerini gösterir.

Çalışma Gerilimi	3 ile 5.5V
Çalışma Sıcaklığı	-55°C to +125°C (-67F to +257F)
-10°C ila +85°C aralığında doğruluk	±0.5°C
Su geçirmez	Evet

Tablo 13 DS18B20 Su geçirmez sıcaklık sensör özellikleri

Su geçirmez sıcaklık sensörünün Raspberry Pi kartı ile bağlantısı fritzing programı üzerinden çizilen devre şeması aşağıda Şekil 3.22'de verilmiştir.



Şekil 32 One-Wire DS18B20 sensör ile Raspberry arasındaki bağlantı devre şeması

DS18B20 sensöründe 3 pin 1 güç hattı (Kırmızı), 1 GND (Siyah) ve 1 DATA hattı (Sarı) bulunur. DS18B20'nin veri hattı, Pi'nin GPIO4'üne beslenmeden önce bir $4.7\text{k}\Omega$ 'a bağlanır, One-Wire GPIO algılaması için özel pin çünkü bu kurulumdaki direnç veri hattı için 'çekme' görevi görecektir. One-Wire veri hattının tanımlanmış mantık seviyesinde olmasını sağlamak ve GPIO pininin serbest kalması durumunda elektriksel gürültüyü azaltmak için kullanılır. Aşağıdaki tablo, DSB1820'nin Pi ile pin bağlantısını göstermektedir.

Raspberry Pi 3B Pinleri	DS18B20 Pinleri
3.3V pin (Pin1)	Güç Kablosu (Kırmızı Kablo)
Topraklama (Pin 6)	Topraklama (Siyah Kablo)
GPIO4 (Pin 7)-OneWire algılaması için özel pin	Data Kablosu (Sarı Kablo)

Tablo 14 DS18B20 Pin Tanımlamaları

Aşağıdaki şekil 3.22'de DS18B20 Su geçirmez sıcaklık sensörünün test edilebilmesi için python'da yazılmış test kodları verilmiştir. Ayrıca aşağıda verilen Tablo 3.16'da test senaryosu, asıl sonuç ve çıktılar arasındaki farklar gösterilmiştir.

```

import os
import glob

#-----
#Initial setup for Temperature Sensor DS18B20
#-----
os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')

base_dir = '/sys/bus/w1/devices/'
device_folder = glob.glob(base_dir + '28*')[0]
device_file = device_folder + '/w1_slave'

#-----
#To read Temperature Sensor DS18B20
#-----
def read_temp_raw():
    f = open(device_file, 'r')
    lines = f.readlines()
    f.close()
    return lines

def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos+2:]
        temp_c = float(temp_string) / 1000.0
        temp_f = temp_c * 9.0 / 5.0 + 32.0
        return temp_c #,temp_f

    while True:
        temp = read_temp()
        print "Temperature :", temp, "=C"
        time.delay(1)

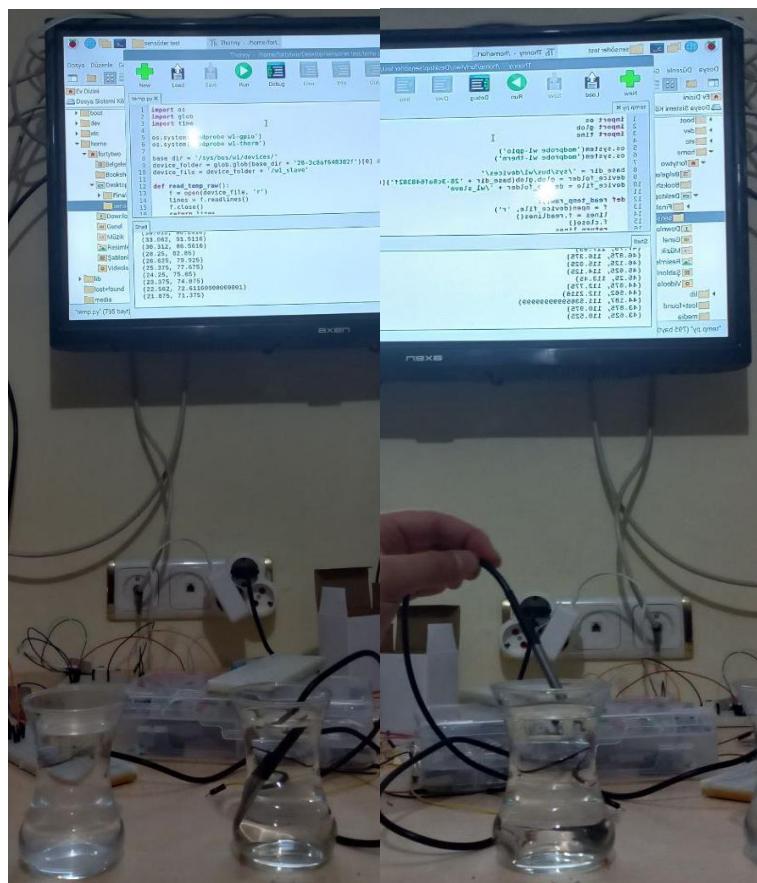
```

Şekil 33 DS18B20 Sıcaklık Sensörü Test Kodları

DS18B20 su geçirmez sıcaklık sensörünün düzgün çalışıp çalışmadığını test etmek için birkaç test durumu tanımlanmıştır. İki bardak sırasıyla sıcak su ve soğuk su ile doldurulur. Sıcak su ve soğuk suyun değeri dijital termometre ile ölçülmüştür ve bu değerler One-Wire DS18B20 su geçirmez sıcaklık sensöründen elde edilen değerle karşılaştırılarak referans olarak kullanılmıştır. Aşağıda verilen Tablo 3.16'da test durumu, gerçek sonuç ve elde edilen sonuç gösterilmektedir. Elde edilen sonuçlar ışığında DS18B20 su geçirmez sıcaklık sensörü beklenildiği gibi gerçek sonuçlara yakın değerler vererek düzgün çalışır durumda olduğunu göstermiştir. Bu projede akvaryum sıcaklığını takip etmek ve belli değerler aralığında tutmaya yardımcı olması açısından en uygun sıcaklık sensörü olarak DS18B20 su geçirmez sıcaklık sensörü seçilmiştir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	DS18B20 su geçirmez sıcaklık sensörü sıcak su dolu bir bardak içine yerleştirmek.	Dijital termometre 45 Santigrat değerini gösterir.	Sensör çıkışı, gerçek sonuçla yaklaşık olarak aynı değerdedir, yani %4,58'lik bir yüzde hatasıyla 43 C'dir.
2	DS18B20 su geçirmez sıcaklık sensörünü soğuk su dolu bardak içine yerleştirmek.	Dijital termometre 22 Santigrat değerini gösterir.	Sensör çıkışı, gerçek sonuçla yaklaşık olarak aynı değerdedir, yani %4,58'lik bir yüzde hatasıyla 25 C'dir.

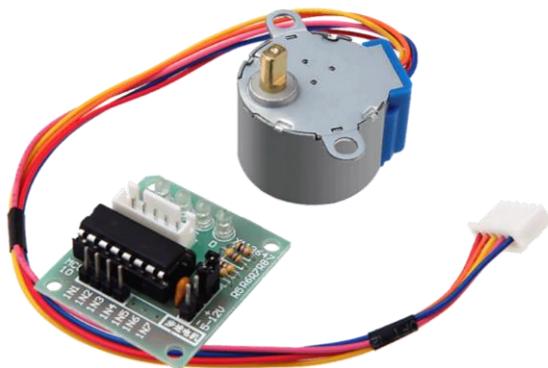
Tablo 15 DS18B20 Sıcaklık sensörünün test senaryosu



Şekil 34 DS18B20 Sıcaklık sensörü test işlemleri

Su sıcaklığının çok yüksek olması örneğin 30 Santigrattan yüksek olması durumunda sistemin soğutma fanı üzerinde otomatik olarak çalışacağı projenin otomasyon kısmında da önemli rol oynamaktadır.

3.3.1.6 28BYJ48 STEP MOTOR



Şekil 35 28BYJ48 Step motor

28BYJ48 Step Motor, motora her gelen darbe uygulandığında elektronik sinyalleri mekanik harekete dönüştüren küçük bir step motordur. Bu proje için en uygun motor olarak seçilmiştir. Step motor için yaygın olarak kullanılan iki mod vardır, tam adım ve yarımadım. Tam adım, Tek Fazlı ve İki Fazlı olmak üzere iki tipe ayrılmıştır. Bir fazda, step motor aynı anda yalnızca bir faza enerji verilerek çalıştırılır. İki fazlı step motor, bir seferde iki faz enerjili olarak çalıştırılır. Oysa yarımadım modunda, önce bir faza, ardından iki faza enerji verilir. Tam adım, yarımadım moduna kıyasla daha fazla torka sahiptir. Bu projede balıkları beslemek için ULN2003 motor sürücüsü ile step motor kullanılacaktır. Tablo 5-0-1-T14, 28BYJ48 Step Motorun temel özelliklerini gösterir.

Çalışma Gerilimi	5.0 (V) DC
Faz Sayısı	4
Hız Değişim Oranı	1/64
Adım Açısı	5.625° /64
Kendinden konumlandırılmalı Tork	>34.3Mn.m

Tablo 16 28BYJ48 Step Motor Özellikleri

3.3.1.7 SULN2003 (STEP MOTOR SÜRÜCÜ PCB KARTI)



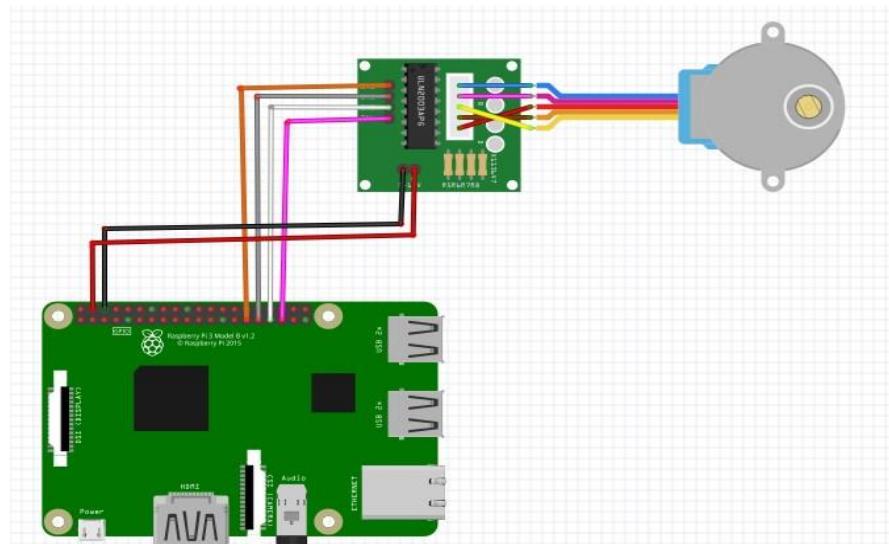
Şekil 36 SULN2003 Step motor sürücü pcb kartı

ULN2003, Raspberry pi ve 28BYJ48 step motor arasında arayüz sağlayan motor sürücüsüdür. Step motor, doğrudan Raspberry pi'den sürülemez çünkü bobinlerinin dönmesi için belirli bir sırayla etkinleştirilmesi gereklidir. Bu nedenle, Raspberry Pi pinlerinin verimli bir şekilde yapamayacağı motorun çeşitli girişlerine gücü doğru darbe dizisine dönüştürme işlemi için bir kontrolöre ihtiyaçları vardır. Ayrıca step motor da Raspberry Pi'nin verebileceği akıma göre yüksek akım çeker. Motoru doğrudan Raspberry Pi'ye bağlarsanız karta zarar verirsınız. ULN2003 step motor sürücü PCB kartı üzerinde, Raspberry Pi'ye bağlantı için 4 giriş, step motor için güç kaynağı bağlantısı ve adım durumunu belirtmek için 4 adet LED bulunmaktadır. Tablo 3.18, ULN2003 Step Motor sürücüsünün temel özelliklerini gösterir.

Çalışma Gerilimi	5.0 (V) DC
Faz Sayısı	4
Adım Açısı	5.625° 1/64
Azaltma oranı	1/64

Tablo 17 ULN2003 Özellikleri

Aşağıda verilen şekil 3.23'te 28BYJ48 Step Motor ve ULN2003 Step motor sürücü kartının bağlantı şeması fritzing programı üzerinden çizilerek verilmiştir.



Şekil 37 28BYJ48 ile ULN2003 Motor sürücü kartı ve Raspberry Pi 3 arasındaki bağlantı

Step motordan ULN2003 motor sürücüsüne 5 adet, Pi'den ULN2003 motor sürücüsüne 4 adet kablo bağlanmıştır. Daha önce tartıştığımız gibi, step motoru sormak için motor sürücüsü gereklidir çünkü Pi'nin GPIO'1 step motorun farklı fazlarına enerji vermek için yetersizdir. Aşağıda Step motor, motor sürücüsü ve Pi arasındaki pin bağlantısı gösterilmektedir.

Raspberry Pi 3 pin	ULN2003 Motor Sürücü	28BYJ48 Step Motor
5V (Pin 4)	Güç Pini	--
Topraklama (Pin 6)	Topraklama Pini	--
GPIO19 (Pin 35)	IN4	--
GPIO13 (Pin 33)	IN3	--
GPIO6 (Pin 31)	IN2	--
GPIO5 (Pin 29)	IN3	--
--	OUT1	Mavi Kablo (Pin 4)
--	OUT2	Pembe Kablo (Pin 3)

--	COM	Kırmızı Kablo (Ortak)
--	OUT4	Turuncu Kablo (Pin 1)
--	OUT3	Sarı Kablo (Pin)

Tablo 18 Raspberry Pi3, ULN2003 Motor Sürücüsü ve 28BYJ48 Step Motor Pin Tanımlamaları

Aşağıda verilen şekilde ise Python'da yazılmış Step motor test kodları verilmiştir.

```

A = 5
B = 6
C = 13
D = 19

GPIO.setup(A, GPIO.OUT) #
GPIO.setup(B, GPIO.OUT) #For Stepper Motor
GPIO.setup(C, GPIO.OUT) #
GPIO.setup(D, GPIO.OUT) #

def GPIO_SETUP(a,b,c,d):
    GPIO.output(A, a)
    GPIO.output(B, b)
    GPIO.output(C, c)
    GPIO.output(D, d)
    time.sleep(0.001)

def Fish_Feeder_RTURN(deg):
    full_circle = 510.0
    degree = full_circle/360*deg
    GPIO_SETUP(0,0,0,0)

    while degree > 0.0:
        GPIO_SETUP(1,1,0,0)
        GPIO_SETUP(1,1,0,0)
        GPIO_SETUP(0,1,0,0)
        GPIO_SETUP(0,1,1,0)
        GPIO_SETUP(0,0,1,0)
        GPIO_SETUP(0,0,1,1)
        GPIO_SETUP(0,0,0,1)
        GPIO_SETUP(1,0,0,1)
        degree -= 1
    GPIO_SETUP(0,0,0,0)

def Fish_Feeder_LTURN(deg):
    full_circle = 510.0
    degree = full_circle/360*deg
    GPIO_SETUP(0,0,0,0)

    while degree > 0.0:
        GPIO_SETUP(1,0,0,1)
        GPIO_SETUP(0,0,0,1)
        GPIO_SETUP(0,0,1,1)
        GPIO_SETUP(0,0,1,0)
        GPIO_SETUP(0,1,1,0)
        GPIO_SETUP(0,1,0,0)
        GPIO_SETUP(1,1,0,0)
        GPIO_SETUP(1,0,0,0)
        degree -= 1
    GPIO_SETUP(0,0,0,0)

```

Şekil 38 Step motor test kodları

Step motorun düzgün çalışıp çalışmadığını kontrol etmek için birkaç test durumu uygulanmıştır. Bir parça çizim kağıdına bazı açıları olan daireler çizilmiş ve step motor dairelerin ortasına yerleştirilmiştir. Step motorun üst çubuğu belirli açı değerleri ile dönüşü tamamlayıp tamamladığını kontrol etmek için bir kağıt bir ok eklenmiştir.

Aşağıdaki tablo test durumunu, gerçek sonucu ve step motor testinden elde edilen sonucu göstermektedir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	Adım motorunu dairenin ortasına yerleştirin ve adımlayıcıyı 360 derece sağa döndürmek için aşağıdaki kod satırını Fish_Feeder_RTURN(360) Çalıştırın.	Step motor 360 derece sağa dönmeli ve orijinal konumuna geri dönmelidir.	Step motor, 360 derece sağa döner ve gerçek sonuca göre orijinal konumuna geri döner. Ancak step motor, orijinal konumuna ulaşmadan önce biraz erken durur.
2	Adım motorunu dairenin ortasına yerleştirin ve adımlayıcıyı 90 derece sağa döndürmek için aşağıdaki kod satırını Fish_Feeder_RTURN(90) Çalıştırın.	Step motor 90 derece sağa dönmeli ve orijinal konumuna geri dönmelidir.	Step motor, 90 derece sağa döner ve gerçek sonuca göre orijinal konumuna geri döner. Ancak step motor, orijinal konumuna ulaşmadan önce biraz erken durur.
3	Adım motorunu dairenin ortasına yerleştirin ve adımlayıcıyı 360 derece sola döndürmek için aşağıdaki kod satırını Fish_Feeder_RTURN(360) Çalıştırın.	Step motor 360 derece sola dönmeli ve orijinal konumuna geri dönmelidir.	Step motor, 360 derece sola döner ve gerçek sonuca göre orijinal konumuna geri döner. Ancak step motor, orijinal konumuna ulaşmadan önce biraz erken durur.
4	Adım motorunu dairenin ortasına yerleştirin ve adımlayıcıyı 90 derece sola döndürmek için aşağıdaki kod satırını Fish_Feeder_RTURN(360) Çalıştırın.	Step motor 90 derece sola dönmeli ve orijinal konumuna geri dönmelidir.	Step motor, 90 derece sola döner ve gerçek sonuca göre orijinal konumuna geri döner. Ancak step motor, orijinal konumuna ulaşmadan önce biraz erken durur.

Tablo 19 Step Motor Test Senaryosu



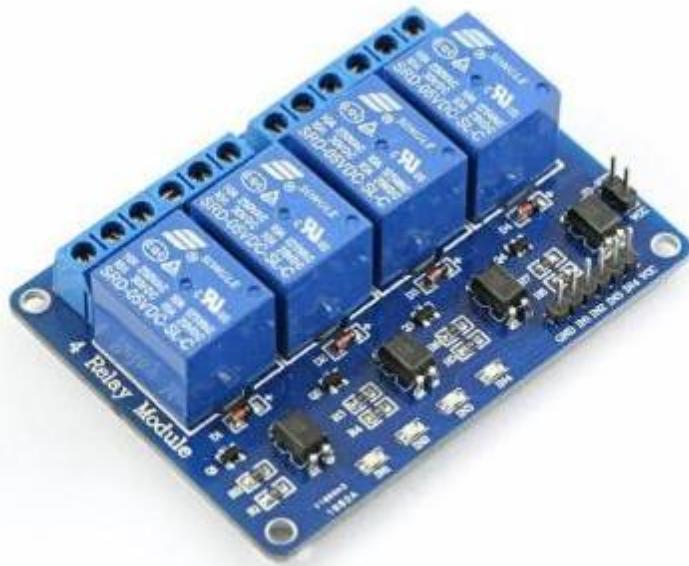
Şekil 39 Step motor bağlantı şekli ve çalışma deneyi

Bu projede küçük bir plastik kap ile bağlantı kurarak balığı beslemek için step motor kullanılmıştır. Ardından kabın dibinde küçük bir delik açılmıştır. Kap, step motor kullanılarak dönerse, balığın kuru yemi delikten düşecektir. Aşağıdaki resimde, plastik kap ile step motoru göstermektedir.



Şekil 40 Yemleme sistemi görüntüsü

3.3.1.8 SRD-05VDC-SL-C (4 KANAL 5V RÖLE MODÜLÜ)

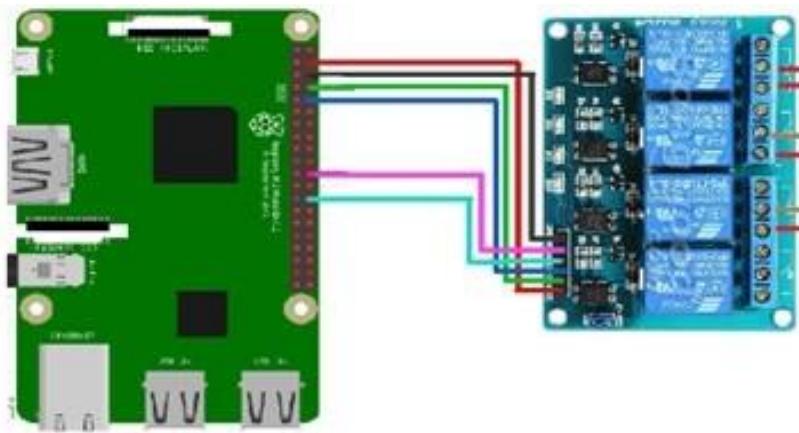


Şekil 41 SRD-05VDC-SL-C 5V 4 Kanallı 5V Röle Modülü

Röle, fan, ışık ve bazı ev aletleri gibi yüksek voltajlı (120 - 240V) cihazları kontrol etmek için kullanılır. Raspberry Pi 3.3V ile çalıştığı için yüksek voltajlı cihazları direkt olarak kontrol edemez. Dolayısıyla Raspberry Pi, röle yardımıyla yüksek gerilimli cihazları açıp kapamak için kullanılabilir. Bu projede, iki su pompası motorunu, bir çift soğutma fanını ve akvaryum için beyaz LED ışığını açıp kapatmak için 4 kanallı 5V röle kullanılmıştır. Aşağıdaki tablo SRD- 05VDC-SL-C 5V rölesinin temel özelliklerini göstermektedir.

Çalışma Gerilimi	250 VAC-110VDC
Çalışma Zaman Aralığı	10 ms Maksimum
Maksimum ON/OFF Anahtarlama	
Mekanik olarak	300 devir/dk
Elektriksel olarak	30 devir/dk
Çalışma Sıcaklığı	-25°C ile +70°C
Çalışma Nemi	45 ile 85% RH

Tablo 20 Röle Özellikleri



Şekil 42 Raspberry pi 3 ile SRD-05VDC-SL-C 5V Bağlantı Devre Şeması

Şekil 3.25, 4 Kanal 5V röle modülü ile Pi 3 arasındaki bağlantıyı göstermektedir. Tek bir 5V rölenin herhangi bir güç kaynağına ihtiyacı yoktur ancak bu projede 4 kanal 5V röle modülü kullanıldığı için 5V güç kaynağına ihtiyaç duyur. Pi'den her rölede led göstergeyi yakmak için. Röle modülündeki LED, ilgili rölenin Pi'nin GPIO pininden sinyal alıp almadığının göstergesi olarak işlev görür. Aşağıda 4 Kanal 5v röle modülü ile Pi 3 arasındaki pin bağlantısı gösterilmektedir.

Raspberry Pi 3B Pinleri	4 Kanal 5V Röle Pinleri
5v pin (Pin4)	VCC
Topraklama (Pin 14)	GND
GPIO15 (Pin 10)	IN2
GPIO7 (Pin 26)	IN3
GPIO25 (Pin 22)	IN1

Tablo 21 Raspberry pi 3 ile SRD-05VDC-SL-C 5V Bağlantı Devre Şeması

```

import RPi.GPIO as GPIO
import time

#GPIO SETUP
#
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#Set GPIO to output
#
GPIO.setup(14,GPIO.OUT) #Relay IN4 for Light
GPIO.setup(15,GPIO.OUT) #Relay IN3 for Pump_1
GPIO.setup(7,GPIO.OUT) #Relay IN2 for Pump_2
GPIO.setup(25,GPIO.OUT) #Relay IN1 for Cooler_Fan
#-----


while True:
    GPIO.output(14,GPIO.HIGH)
    GPIO.output(15,GPIO.HIGH)
    GPIO.output(7,GPIO.HIGH)
    GPIO.output(25,GPIO.HIGH)
    print "-"
    print "The relay 1 is on"
    GPIO.output(25,GPIO.LOW)
    time.sleep(1)
    print "The relay 1 is off"
    GPIO.output(25,GPIO.HIGH)
    time.sleep(1)
    print "-"
    print "The relay 2 is on"
    GPIO.output(7,GPIO.LOW)
    time.sleep(1)
    print "The relay 2 is off"
    GPIO.output(7,GPIO.HIGH)
    time.sleep(1)
    print "-"
    print "The relay 3 is on"
    GPIO.output(15,GPIO.LOW)
    time.sleep(1)
    print "The relay 3 is off"
    GPIO.output(15,GPIO.HIGH)
    time.sleep(1)
    print "-"
    print "The relay 4 is on"
    GPIO.output(14,GPIO.LOW)
    time.sleep(1)
    print "The relay 4 is off"
    GPIO.output(14,GPIO.HIGH)
    time.sleep(1)

```

Şekil 43 Röle test kodları

Röleyi test etmek ve düzgün çalıştığından emin olmak için birkaç test durumu tanımlanmıştır. Röleye Pi'den sadece düşük bir sinyal (0) gönderiyorum ve röle 0 sinyalini aldıysa röle üzerinde olacak ve ilgili rölenin LED göstergesi açılacak. Yüksek 1 sinyali alırsa röleyi kapatacak ve LED sönektir. Aşağıda gerçekleştirilmiş olan test durumu, gerçek sonuç ve elde edilen sonuç gösterilmektedir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	Röle üzerinden 1. röleye (IN1) 1 saniye boyunca düşük sinyal (0) gönder.	IN1 rölesi açık ve LED göstergesi yanıyor.	IN1 rölesi açık ve gerçek sonuç olarak LED göstergesi yanıyor.
2	Röle üzerinden 1. röleye (IN2) 1 saniye boyunca düşük sinyal (0) gönder.	IN2 rölesi açık ve LED göstergesi yanıyor.	IN2 rölesi açık ve gerçek sonuç olarak LED göstergesi yanıyor.
3	Röle üzerinden 1. röleye (IN2) 1 saniye boyunca düşük sinyal (0) gönder.	IN3 rölesi açık ve LED göstergesi yanıyor.	IN3 rölesi açık ve gerçek sonuç olarak LED göstergesi yanıyor.

4	Röle üzerinden 1. röleye (IN2) 1 saniye boyunca düşük sinyal (0) gönder.	IN4 rölesi açık ve LED göstergesi yanıyor.	IN4 rölesi açık ve gerçek sonuç olarak LED göstergesi yanıyor.
---	--	--	--

Tablo 22 Röle Test Senaryosu

Aşağıda verilen resimlerde gerçekleştirilen röle test durumları verilmiştir. Resimlerde de görüldüğü üzere röle beklenildiği gibi çalışmış ve devrede anahtarlama görevi ile kullanılmaya hazır olduğu belirlenmiştir.



Şekil 44 Röle kartı test işlemleri

3.3.1.9 DS3231 RTC KART (GERÇEK ZAMAN SAAT MODÜLÜ)



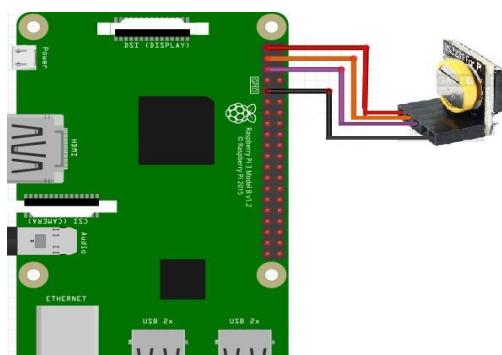
**Şekil 45 DS3231
Saat Modülü**

DS3231, Raspberry pi'nin güç kapalıken bile zamanı tutmasına yardımcı olan son derece hassas bir I2C gerçek zamanlı saattir. Varsayılan olarak, Raspberry Pi'de zamanı tutmak için bir RTC yapısı yoktur, ancak bunun yerine Pi, küresel ağ zaman protokolü (NTP) sunucularından saatini otomatik olarak güncellemek için internete bağlanmaya çalışacaktır. Ancak internet bağlantısı yoksa Pi saatini güncelleyemez. Böylece, bu düşük maliyetli DS3231 Pi'ye bağlanarak kapatıldıktan sonra da zamanı koruyabilir. Projemizde RTC çok önemli çünkü Pi'nin belirli bir zamanda kendine yapması gereken bazı otomatik adımlar var, örneğin akşam 9'da balığı beslemek veya geceyse ışığı yaktır. Aşağıdaki tablo, DS3231'in temel özelliklerini göstermektedir.

Çalışma Gerilimi	3.3 V
Çalışma Sıcaklığı Aralıkları	Ticari (0°C ila $+70^{\circ}\text{C}$) Endüstriyel (-40°C ila $+85^{\circ}\text{C}$)
Arayüz	I2C
Kullanılan $\pm 3,5$ ppm	-40°C to $+85^{\circ}\text{C}$

Tablo 23 DS3231'in temel özelliklerini

Aşağıda bulunan şekil 3.26'da fritzing programı üzerinden çizilen Raspberry Pi ve DS3231 RTC Gerçek Zaman Saat Modülünün devre bağlantı şeması gösterilmektedir.



**Şekil 46 Raspberry Pi 3 ile DS3231 RTC
karti arasındaki bağlantı**

Harici RTC olmadan, Pi'nin her açılışında internetten saatı güncellemesi gereklidir. Pi'nin saatini güncellemesi için Internet bağlantısının olmadığı durumda bu zor olabilir. Böyle bir sorunu çözmek için DS3231 RTC kullanılır. Aşağıda verilen Tablo 3.25'de RTC'nin Pi ile bağlantı pinleri gösterilmektedir.

Raspberry Pi 3B Pinleri	DS3231 RTC Pinleri
5v pin (Pin 1)	VCC
Topraklama (Pin 9)	GND
GPI02 (Pin 3) – SDA1 I2C	D(SDA)
GPIO3 (Pin 5) – SCL1 I2C	C(SCL)

Tablo 24 Pin Tanımlamaları

Bağlantının yanı sıra DS3231 RTC kartının çalışması için yapılması gereken bazı konfigürasyonlar vardır. Öncelikle, Raspberry Pi raspbian işletim sisteminde terminal ekranını açarak bu bölümde belirtilen komut satırlarının girilmesi gereklidir. İlk olarak /boot/config.txt dosyasının en altına aşağıdaki satır eklenerek düzenlenmesi gereklidir,

dtoverlay = i2c-rtc,ds3231

İkinci olarak yine terminal ekranına /lib/udev/hwclock-set komutu yazılarak açılan ekranda bulunan hwclock-set dosyasını düzenleyin ve sistem tarafından göz ardı edilmeleri için aşağıdaki satırı yorumlanmalıdır,

if [-e/run/system/system]; then

exit 0

#fi

Yukarıdaki satırları yorumlama işlemi gerçekleştirildiğinde. Aşağıdaki gibi bir kod satırı olmuş olacaktır.

#if [-e/run/system/system]; then

#exit 0

#fi

Ardından Pi'nin sistemini yeniden başlatılır ve “sudo hwclock -r” komutuyla RTC test edilir. Sağdaki sistem saatı doğrudan RTC'den okunmalıdır. Varsayılan olarak Pi, geçerli tarih ve saatı bir NTP sunucusu, internet üzerinden bir zaman sunucusu aracılığıyla alır. RTC'yi bağlayarak ve yukarıda gösterilen bazı konfigürasyonları yaparak, sistem şimdi

internet yerine RTC panosundan zaman almalıdır. Pi'nin zamanı gerçekten internetten değil de RTC'den okuyup okumadığını test etmek için Pi'yi belirlenen bir zaman aralığı miktارınca kapatmak gereklidir. Kapatılma işleminden önce pi küresel ve yerel ağdan ayrılmalıdır. Pi'yi tüm ağlardan ayırarak, NTP sunucusundan geçerli saat ve tarihi alması engellenmiş olur. Ardından, HDMI bağlantısı ile görüntü elde ettiğimiz monitör üzerinden Pi'yi açmadan önce 15 dakika bekleme süresi belirlenmiştir. Sonra pi'nin şu anki saatini kontrol edilerek arada geçen zamanın doğrulu kontrol edilmiştir. Pi'nin şu anki saatı doğru ve 15 dakika gecikmeli olmadığı gözlemlenmiştir. Böylece DS3231 RTC kartının mükemmel çalıştığı sonucuna varılmıştır. Bu projede RTC özellikle projenin otomasyon kısmında çok önemlidir, çünkü sistem kullanıcı tarafından belirlenen zamanda balıkları beslemek ve akvaryum ışığını değiştirmek zorundadır. Dolayısıyla, bu tür eylemleri gerçekleştirmek için Pi'nin geçerli saatı doğru bilmesi gereklidir ve harici RTC DS3231, internet olmadığından saat tutmaya yardımcı olabilir.

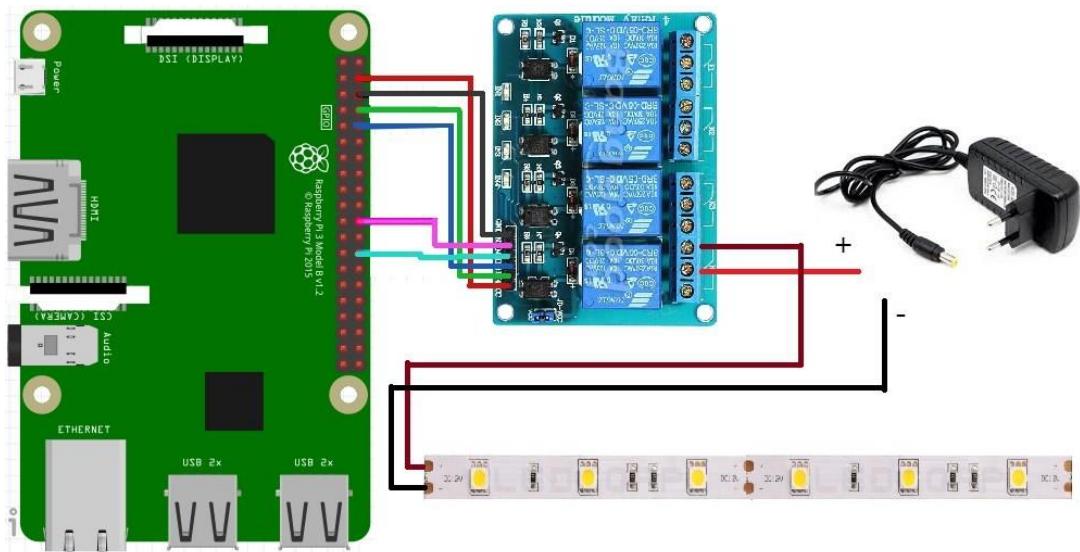
3.3.1.10 3528 Beyaz Şerit LED

3528 beyaz 3 LED şerit, genellikle iç aydınlatma, vitrin ve tavan dekorasyonu için kullanılan düşük güç tüketimi parlak beyaz LED'dir. Bu projede, akvaryumu aydınlatmak için 3528 silikonlu beyaz şerit LED kullanılmıştır. Aşağıdaki tablo, 3528 beyaz şerit LED'in temel özelliklerini göstermektedir.

Çalışma Gerilimi	12 (V) DC
Çalışma Gücü	24W/5M
Çalışma Akımı/metre	0.35-0.625A
Çalışma sıcaklığı	-20 için +50 C
Renk	Beyaz
Su geçirmez	Evet

Tablo 25 3528 Beyaz Şerit LED Özellikleri

Aşağıda verilen şekilde 3528 Beyaz şerit LED, 4 Kanallı 5V Röle modülü ve Raspberry Pi kartımızın bağlantı şeması fritzing programı üzerinden çizilerek verilmiştir.



Sekil 47 5050 BEYAZ LED ŞERİT'in Raspberry Pi 3 ile Bağlantısı

Beyaz LED, DC 12V 1A harici güç kaynağı kullanılarak açılıyor. Güç kaynağı ve LED'in güç hattı K4 rölesine bağlanır. LED'in toprağı, harici güç kaynağının topraklamasına bağlanır. 4 kanallı 5V röle kartını 5V gücüne ve Pi'nin toprağına bağlanmıştır. Ardından Pi'nin GPIO14'ünden (Pin 8) gelen kabloyu 4 kanallı röle kartının IN4 pinine bağlanır. Pi'ler LOW 'u IN4'e gönderdiğinde, röle güç kaynağını beyaz LED'e bağlayacak ve bu da ışığı açacak böylece röleyi ve ışığı açmak için HIGH sinyali gönderecektir.

```

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(14,GPIO.OUT) #Relay IN4 for Ligth
GPIO.setup(15,GPIO.OUT) #Relay IN3 for Pump_1
GPIO.setup(7,GPIO.OUT)  #Relay IN2 for Pump_2
GPIO.setup(25,GPIO.OUT) #Relay IN1 for Cooler_Fan

def isikAc():
    while True:
        #GPIO.setup(14,GPIO.OUT) #Relay IN4 for Ligth

        print("-----")
        print("IŞIK AÇILDI")

        GPIO.output(14,GPIO.HIGH)
        time.sleep(1)

        print("-----")
        break

```

Şekil 48 Röle ve pi kullanarak beyaz led'i kontrol etmek için kod

Beyaz ledin çalışıp çalışmadığını test etmek için iki test durumu tanımlanmıştır. Aşağıda test durumunu, gerçek sonucu ve elde edilen sonuçları gösteren bir tablo verilmiştir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	5050 beyaz LED için 12V güç kaynağını açın ve Actuator.py'de Light_On() işlevini çağırarak pi'den röleye 0 sinyali gönderin.	Röle devreyi tamamlamalı ve 5050 beyaz led yanmalıdır.	Gerçek sonuç olarak 5050 beyaz led yanar.
2	5050 beyaz LED için 12V güç kaynağını açın ve Actuator.py'de Light_On() işlevini çağırarak pi'den röleye 1 sinyali gönderin.	Röle devreyi kesmeli ve 5050 beyaz led sönmelidir.	Gerçek sonuç olarak 5050 beyaz led kapanır.

Tablo 26 Beyaz LED Test Senaryosu

3.3.1.11 3528 SMD RGB LED



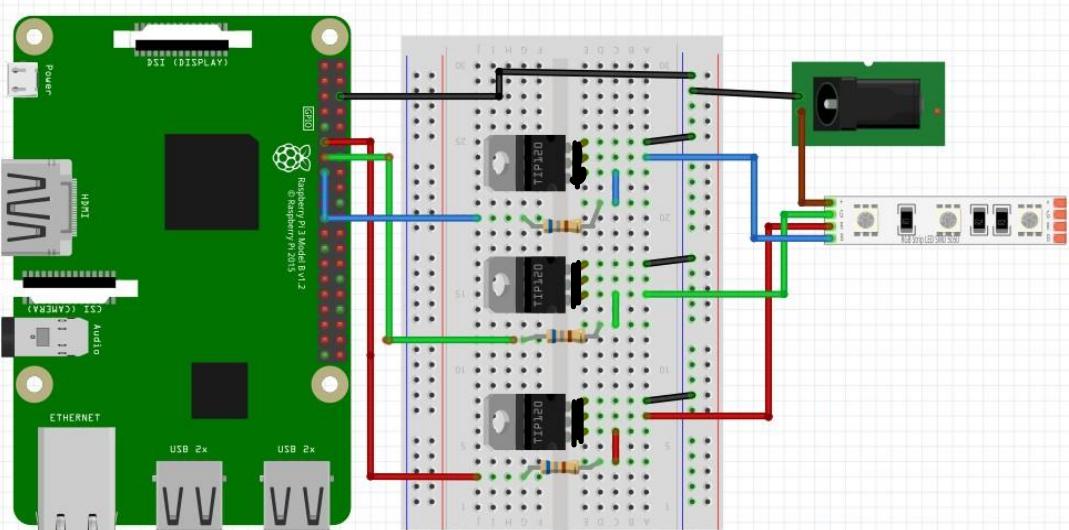
Şekil 49 3528 SMD RGB Şerit LED

3528 SMD RGB LED, Kırmızı, Yeşil ve Mavi olmak üzere 3 LED'den oluşur. Bu projede, bu RGB LED, akvaryumda su sızıntısı veya akvaryum su sıcaklığının çok yüksek olması gibi bir sorun olduğunda kullanıcıya gösterge olarak kullanılacaktır. Örneğin sistem su sızıntısını tespit ederse, RGB LED kullanıcı için bir uyarı göstergesi olarak otomatik olarak sürekli KIRMIZI renkte yanıp sönecektir. Ayrıca RGB LED, akvaryum için aydınlatma dekorasyonu olarak da kullanılır. Aşağıdaki tablo, 3528 SMD RGB LED'in temel özelliklerini göstermektedir.

Çalışma Gerilimi	12 (V) DC
Çalışma Gücü	24W/5M
Çalışma Akımı/metre	0.35-0.625A
Çalışma sıcaklığı	-20 için +50 C
Renk	RGB (Kırmızı, yeşil ve mavi)
Su geçirmez	Hayır

Tablo 27 RGB LED Özellikleri

Aşağıda bulunan şekilde fritzing üzerinden çizilen Raspberry pi ve RGB LED bağlantı devre şeması verilmiştir.



Şekil 50 Raspberry Pi 3 ile RGB LED bağlantısı

3528 SMD RGB led şerit 4 giriş pininden oluşur, R (KIRMIZI), G (YEŞİL), B (MAVİ) ve V+ güç hattı. 12V 2A harici bir güç kaynağı kullanılarak açılır. RGB LED şeridinin rengini kontrol etmek için, sırasıyla LED'in her bir R, G ve B'sini kontrol etmek için üç TIP121 güç transistörü kullanılmıştır. Bu projede TIP121 Transistörler RGB LED'i hızlı bir şekilde açıp kapatacak olan anahtar görevi göreceklere dir. Temel akımı sınırlamak ve Pi'nin GPIO'sunu korumak için transistörlerin her bir tabanı ile Pi'nin GPIO'su arasına 680Ω 'luk bir direnç bağlanır. Harici güç kaynağı topraklaması Pi'nin toprağına bağlanmalıdır. Aşağıdaki Tablo 3.29'da Pi ve TIP121 transistörü arasındaki pin bağlantısını gösterilmiştir.

Raspberry Pi 3B Pinleri	TIP121 Pinleri
GPIO27 (Pin 13)	R (Alttan 1. transistör)
GPIO22 (Pin 15)	G (Alttan 2. transistör)
GPIO17 (Pin 11)	B (Alttan 3. transistör)

Tablo 28 Raspberry Pi ve TIP121 Transistör Pin tanımlamaları

Bağlantının yanı sıra Pi, RGB LED'leri kontrol etmek için pigpio adlı önemli bir kütüphaneye ihtiyaç duyar. Pigpio, Raspberry'nin servo konumlandırmasına, motor hızını kontrol etmesine, LED parlaklığını kontrol etmesine vb. yardımcı olan bir kütüphanedir. Aşağıdaki komutu kullanarak pigpio kitaplığını Pi'ye indirilir ve kurulur,

```
rm pigpio.zip  
sudo rm -rf PIGPIO  
  
wget  
abyz.co.uk/rpi  
/pigpio/pigpio  
.zip      unzip  
pigpio.zip  
cd PIGPIO  
make  
  
sudo make install
```

Yukarıda verilen komutlar raspbian işletim sisteminde bulunan terminal ekranına yazılacak ve kütüphanenin indirme işlemleri böylelikle tamamlanacaktır. Artık RGB ışığını kontrol etmek için pigpio kitaplığını kullanabiliriz. Pigpio kitaplığını başlatmak için Pi terminalinde aşağıdaki komutun yürütülmesi gereklidir,

A screenshot of a terminal window titled 'pi@raspberrypi: ~ \$'. Inside the window, the command 'sudo pigpiod' is being typed by the user.

Şekil 51 pigpio kitaplığını başlatmak için gereken kod

(Kitaplığı kullanmak için Pi'nizin sistemini her yeniden başlattığınızda yukarıdaki komutu çalıştırmanız gerektiğini unutulmamalıdır.)

```

# RGB LED
#Kirmizi
def kirmiziAc():
    #pi.set_PWM_dutycycle(27, 0) # Red kanalı 16
    #pi.set_PWM_dutycycle(22, 0)
    pi.set_PWM_dutycycle(17, 255)
    """
    while True:
        print("Red RGB is on")
        kirmiziAc()
        break
    """
#Yesil
def yesilAc():
    pi.set_PWM_dutycycle(27, 255)
    #pi.set_PWM_dutycycle(22, 0) # Green Kanalı 20
    #pi.set_PWM_dutycycle(17, 0)
    """
    while True:
        print("Green RGB is on")
        yesilAc()
        break
    """
#Mavi
def maviAc():
    #pi.set_PWM_dutycycle(27, 0)
    pi.set_PWM_dutycycle(22, 255)
    #pi.set_PWM_dutycycle(17, 0) # Blue Kanalı 21
    """
    while True:
        print("Blue RGB is on")
        maviAc()
        break
    """
#KirmiziKapat
def kirmiziKapat():
    #pi.set_PWM_dutycycle(27, 0)
    #pi.set_PWM_dutycycle(22, 0)
    pi.set_PWM_dutycycle(17, 0)
    """
    while True:
        print("Red RGB is off")
        kirmiziKapat()
        break
    """
#Yesilkapat
def yesilKapat():
    pi.set_PWM_dutycycle(27, 0)
    #pi.set_PWM_dutycycle(22, 0)
    #pi.set_PWM_dutycycle(17, 0)
    """
    while True:
        print("Green RGB is off")
        yesilKapat()
        break
    """
#Mavikapat
def maviKapat():
    #pi.set_PWM_dutycycle(27, 0)
    pi.set_PWM_dutycycle(22, 0)
    #pi.set_PWM_dutycycle(17, 0)

```

Şekil 52 RGB Ledlerin çalışması için gereken kodlar

RGB'nin kusursuz çalışıp çalışmadığını kontrol etmek için üç test senaryolu test planı tanımlanmıştır. Aşağıda test senaryoları, gerçek sonuçlar ve test planından elde edilen sonuçlar gösterilmektedir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	RGB LED'ler için güç kaynağını açın ve RGB.py'de red() adlı işlevi çalıştırın.	RGB LED kırmızı renge dönmelidir.	RGB LED kırmızı renge döner.
2	RGB LED'ler için güç kaynağını açın ve RGB.py'de blue() adlı işlevi çalıştırın.	RGB LED mavi renge dönmelidir.	RGB LED mavi renge döner.
3	RGB LED'ler için güç kaynağını açın ve RGB.py'de green() adlı işlevi çalıştırın.	RGB LED yeşil renge dönmelidir.	RGB LED yeşil renge döner.

Tablo 29 3528 RGB LED'in Test Senaryosu



Şekil 53 RGB LED test işlemleri

Bu projede RGB LED, kullanıcı için akvaryum durumu hakkında bir göstergе olarak kullanılmıştır. Örneğin herhangi bir su kaçagi tespit edilirse veya su sıcaklığı çok yüksekse kullanıcı için uyarı göstergesi olarak RGB LED Kırmızıya döner. Ayrıca RGB, kullanıcının akvaryumu hangi renge boyamak istediğini arayüz üzerinden kontrol edebileceği akvaryum için aydınlatma dekorasyonu olarak da işlev görür. Kullanıcı, arayüzde bulunan kaydırıcı aracılığıyla sarı veya mor gibi farklı renkler oluşturmak için her bir RGB LED'in parlaklığını kontrol edebilir.

3.3.1.12 3.5V~12V DC FIRÇASIZ SU POMPASI (x2)



Şekil 54 DC12V 5W Mini Fırçasız Su Pompası (Su Geçirmez)

Bu projede, suyu akvaryuma ve akvaryumdan pompalamak için iki DC Fırçasız su pompası kullanılmıştır. Aşağıda 3.5V~12V DC Fırçasız su pompasının temel özellikleri gösterilmektedir.

Çalışma Gerilimi	3.5V ile 12V
Çalışma akımı	50mA-500mA
Saatte maksimum su pompası hacmi	350L
Sürekli çalışırsa yaşam süresi	> 20000 saat

Tablo 30 Su pompası özellikleri

3.3.1.13 Soğutucu Fan (x2)

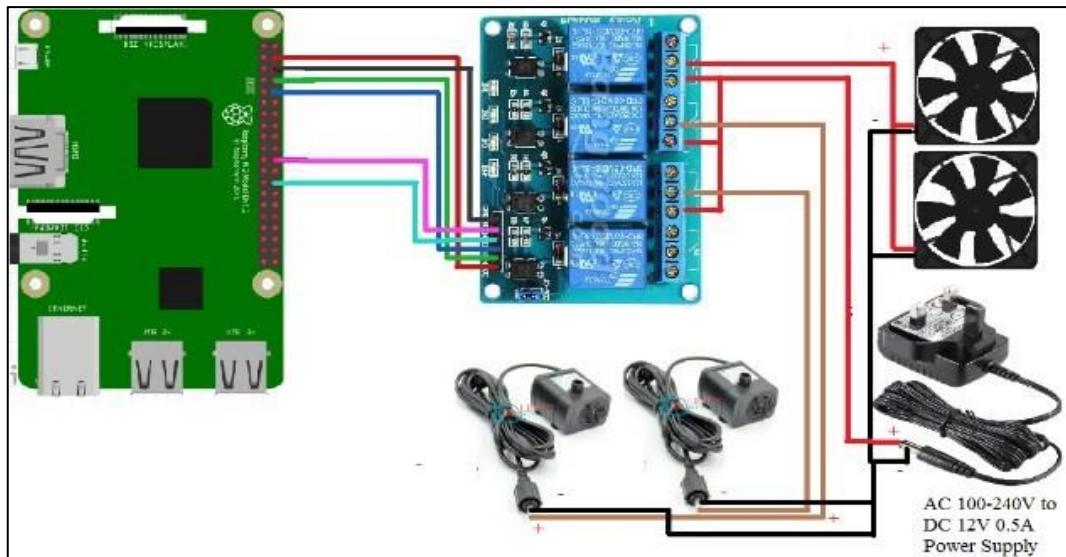


Şekil 55 Soğutucu Fan

Bu projede evaporatif soğutma yöntemi ile akvaryum su sıcaklığını düşürmek için iki adet DC motorlu soğutucu fan kullanılmıştır. DC motor soğutucu fan, akvaryum için ticari soğutma fanı fiyatı çok yüksek olduğu için laptop soğutucu fanından alınmıştır. Aşağıdaki tablo DC motorun temel özelliklerini göstermektedir. Evaporatif (Buharlaştırmalı) soğutma basit bir prensibe dayanır. Buharlaşma her sıcaklıkta olabilir. Maddeler dışarıdan ısı alarak buharlaşırlar. Dolayısıyla buharlaşmanın olduğu yerde serinleme meydana gelir. Sıcaklığın artması buharlaşmayı aynı oranda hızlandırır.

Çalışma Gerilimi	3.5V ile 12V
Çalışma akımı	50mA-1 A
Boyutu	5cm X 5cm

Tablo 31 DC Motor FAN genel özellikler



Şekil 56 Aktüatörler arasındaki bağlantı

Şekil 3.31, iki adet su pompası, bir adet 4 kanal röleli soğutma fanı ve Raspberry Pi 3'ün bağlantısını göstermektedir. Soğutma fanı ve su pompası, önceki sayfada gösterildiği gibi aynı kurulumla bağlanmıştır. Ancak soğutma fanı K1 rölesini kullanacak ve su pompası bir ve iki sırasıyla K2 ve K3 rölesini kullanacaktır. Aşağıda 4 kanallı röle kartının Pi ile pin bağlantısı gösterilmektedir.

Raspberry Pi 3B Pinleri	4 Kanal 5V Röle Pinleri	Bileşen kontrolü
5v pin (Pin4)	VCC	--
Topraklama (Pin 14)	GND	--
GPIO15 (Pin 10)	IN2	1.Pompa
GPIO7 (Pin 26)	IN3	2.Pompa
GPIO25 (Pin 22)	IN1	Soğutucu Fan

Tablo 32 Pin Tanımlamaları

```
# FAN AÇMA

def faniaC():
    while True:
        GPIO.setup(25,GPIO.OUT) #Relay IN4 for Ligth

        print("-----")
        print("FAN AÇILDI")

        GPIO.output(25,GPIO.LOW)
        time.sleep(1)

        print("-----")
        break

# POMPA AÇMA

def pompa1Ac():

    temizle_cevap=messagebox.askyesno("Su Boşalt","Su Boşaltmak\nistediğinizden Eminmisiniz ?")
    if temizle_cevap==1:
        while True:
            #GPIO.setup(7,GPIO.OUT) #Relay IN4 for Ligth

            print("-----")
            print("POMPA 1 AÇILDI")

            GPIO.output(7,GPIO.LOW)
            time.sleep(1)

            print("-----")
            messagebox.showinfo("Su Boşalt","SU BOŞALTILIYOR..")
            break
    else:
        pass|
```

Şekil 57 Soğutucu fan ve su pompası test kodları

Su pompasının ve soğutma fanının çalışıp çalışmadığını test etmek için birkaç test durumu tanımlanmıştır. Aşağıda test durumunu, gerçek sonucu ve elde edilen sonucu göstermektedir. Yapılacak testler önceden belirlenmiş olup devre şemasında gösterildiği üzere devre kurulduktan sonra test işlemleri gerçekleştirılmıştır.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	Su pompası için güç kaynağını açın ve Actuators.py'de Pump1_On() fonksiyonunu çalıştırın.	Röle devreyi tamamlamalı ve su pompası 1 açılmalıdır.	Su pompası 1 gerçek sonuç olarak açılır.
2	Su pompası için güç kaynağını açın ve Actuators.py'de Pump1_Off() fonksiyonunu çalıştırın.	Röle devreyi kesmeli ve su pompası 1 kapanmalıdır.	Su pompası 1 gerçek sonuç olarak kapanır.
3	Su pompası için güç kaynağını açın ve Actuators.py'de Pump2_On() fonksiyonunu çalıştırın.	Röle devreyi tamamlamalı ve su pompası 2 açılmalıdır.	Su pompası 2 gerçek sonuç olarak açılır.

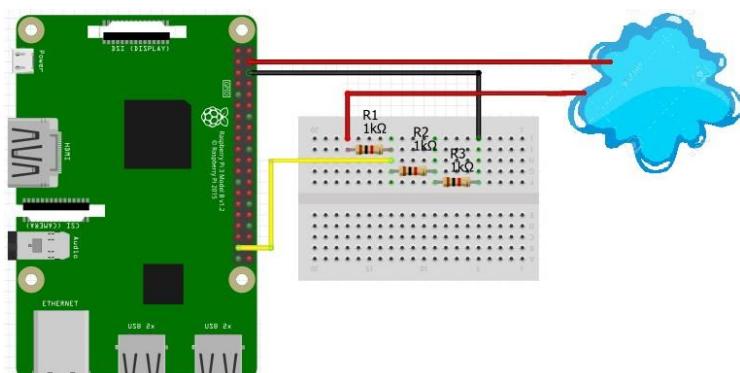
4	Su pompası için güç kaynağını açın ve Actuators.py'de Pump2_Off() fonksiyonunu çalıştırın.	Röle devreyi kesmeli ve su pompası 2 kapanmalıdır.	Su pompası 2 gerçek sonuç olarak kapanır.
5	Su pompası için güç kaynağını açın ve Actuators.py'de CoolerFan_On() fonksiyonunu çalıştırın.	Röle devreyi tamamlamalı ve soğutucu fan açılmalıdır	Soğutucu fan gerçek sonuç olarak açılır.
6	Su pompası için güç kaynağını açın ve Actuators.py'de CoolerFan _Off() fonksiyonunu çalıştırın.	Röle devreyi kesmeli ve su soğutucu fan kapanmalıdır.	Soğutucu fan gerçek sonuç olarak kapanır.

Tablo 33 Aktüatörler için test senaryosu

Bu projede, akvaryuma su pompalamak ve dışarı pompalamak için iki adet su pompası kullanılmıştır. Suyu dışarı pompalamak için akvaryumun içine bir pompa yerleştirilecek ve yeni su kaynağına başka bir pompa yerleştirilecek ve borusu akvaryuma suyu pompalamak için akvaryuma yerleştirilecektir. Soğutma fanı, evaporatif soğutma yöntemi adı verilen bir yöntemle akvaryumun su sıcaklığını düşürmek için kullanılır. Evaporatif soğutma yöntemi, yüzey boyunca hava üfleyerek suyun yüzeyinden ısını hareket ettiren bir kavramdır. Su yüzeyine hava üfleyerek buharlaşma hızı (sıvıdan gaza geçiş) artacaktır. Sıvıdan buhara geçiş hızı arttıkça su daha fazla enerji salacaktır. Bu projede kullanılan soğutma fanı, akvaryumun su yüzeyine hava üfleyerek sıcaklığını düşürecek iki DC motordan oluşmaktadır. Bu buharlaşmalı soğutma yöntemi çok yavaş bir işlemidir, bu nedenle sıcaklık düşürme hızı da daha yavaş bir oranda azalır.

3.3.1.14 Sıvı Seviye Sensörü (220V)

Şekil 3.33, Raspberry pi kullanan basit bir su kaçagi detektörünü göstermektedir. Devre şeması fritzing programı üzerinden çizilerek şekilde gösterilmiştir.



Şekil 58 Su sızıntı test devresi

Biz sadece 1 tel Pi'nin 5V pinine ve diğer Pi'nin GPIO26'sına (Pin 37) giriş olarak bağlı olan iki kablo bağlantısı kurduk. 3.3V güç kaynağı yerine 5V kullanılır, çünkü daha yüksek akımla su elektriği kolayca iletebilir. Ancak Pi'nin GPIO'ı yalnızca 3.3V giriş voltajına dayanabilir. Bu nedenle, 5V kablosu, 3 dirençli Ultrasonik ve Bulanıklık bölümünde gösterildiği gibi bir voltaj bölücü kural devresine bağlanır. Burada gerilim bölücü devresine bağlanan 220V sıvı seviye sensörü Sıvı dolu tanklardaki sıvı seviyesini algılamak için kullanılır. Bir pompa, gösterge, alarm ve bunun gibi cihazları aktive edebilir. Sıvı seviye sensörü ile depolarınızın dolduğunu veya boşaldığını görebilirsiniz. Sensör kuru kontak şeklinde çıkış verir. Yerleştirdiğiniz konuma göre normalde açık veya normalde kapalı olabilir. Sıvı üstüne yada sıvı içeresine yerleştirilebilir.

```
def suSizintiDurumu():
    entry3['state']='normal'
    entry3.delete(0,tk.END)
    entry3['state']='disabled'
    b9['state']='normal'
    while True:
        state = GPIO.input(26)
        if state == GPIO.HIGH:
            print("SIZINTI VAR")
            entry3['state']='normal'
            entry3.insert(0,"Sızıntı Var")
            entry3['state']='disabled'
            messagebox.showinfo("Sızıntı","SIZINTI VAR")
            break
        else:
            print("SIZINTI YOK")
            entry3['state']='normal'
            entry3.insert(0,"Sızıntı Yok")
            entry3['state']='disabled'
            messagebox.showinfo("Sızıntı","SIZINTI YOK")
            break
    time.sleep(1)
    print("-----")|
```

Şekil 59 Su sızıntı sensörü test kodları

Yukarıdaki şekil 3.34'te su sızıntı sensörü testi için python'da yazılmış kodlar verilmiştir. 220V su sızıntı sensörünün düzgün çalışıp çalışmadığını test etmek için iki test senaryosu ile test planı tanımlanmıştır. Su sızıntısı sensörü kodunu çalıştırıyoruz ve suyu algılayıp uyarı mesajı verip vermediğini kontrol etmek için su sızıntı sensörünü su dolu bir kabin içeresine yerleştiriyoruz. Ardından sensörü elimizle biraz yukarı kaldırdığımızda su seviyesinin azaldığını anlayarak test kodunda yer alan “su sızıntısı tespit edildi” çıktısını

elde ediyoruz. Aşağıda test durumları, gerçek sonuç ve testten elde edilen sonuç gösterilmektedir.

No	Yapılacak Test	Asıl Sonuç	Sonuç
1	Su sızıntı sensörünü su içerisinde bırakın.	Su sızıntısı sensör devresi tamamlanmadı ve GPIO26 düşük sinyal alacak ve “Su sızıntısı yok” mesajını verecek.	“Su sızıntısı yok” mesajını verdi.
2	Su sızıntı sensörünü su içerisinde biraz çıkartın.	Su sızıntısı sensör devresi tamamlanmadı ve GPIO26 düşük sinyal alacak ve “Su sızıntısı tespit edildi” mesajını verecek.	“Su sızıntısı tespit edildi” mesajını verdi.

Tablo 34 Su Sızıntı Sensörü Test Senaryosu



Şekil 60 Su sızıntı sensörü test işlemleri

3.3.1.15 Logitech C920 Hd Pro Web Kamerası



Şekil 61 Logitech C920 Hd Pro Webcam

Bu projede, akvaryum durumunu uzaktan kontrol etmek için akvaryumun videosunu web sitesine canlı olarak aktarmak için Logitech C920 HD pro web kamerası kullanılmıştır. Ayrıca, Sesle Kontrol uygulamasını kullanarak sistemi kontrol etmek için C920'deki gömülü mikrofonda kullanılmıştır. Aşağıda C920'in kamera özellikleri gösterilmektedir,

Bağlantı Tipi	Kablolu USB
Odak Türü	Otomatik Odaklama
Diyagonal görüş alanı (dFoV)	78°
Maksimum Çözünürlük	1080p/30 fps - 720p/30 fps

Tablo 35 C920 HD Pro Webcam Özellikleri



Şekil 62 Web kamerasının yerleşimi

Resim 3.39, C920 Logitech kamerasının Raspberry Pi 3'e nasıl bağlandığını gösterir. Bağlantının yanı sıra, USB kamerası test etmek için Pi'ye bir kitaplık indirip kurulması gereklidir. Önce Pi'nin terminalinde aşağıdaki komutu kullanarak fswebcam kütüphanesi indirip kurulmalıdır,

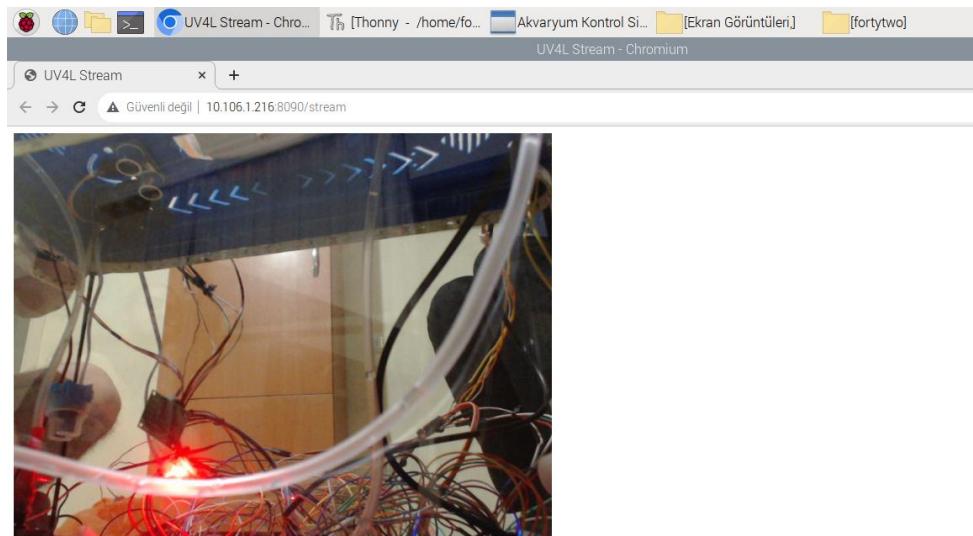
sudo apt-get install fswebcam

Kurulum tamamlandıktan sonra artık C920 USB kamera kullanılarak resim çekilebilir.

Sadece fswebcam ve ardından bir dosya adı girildiğinde görüntü çekilecektir. Örneğin,

fswebcam image.jpg

Aşağıda, yukarıdaki komut yürütüldüğünde C920 USB kamera tarafından çekilen resim gösterilmektedir.



Şekil 63 Akvaryum 'un UV4L canlı akış sunucusu kullanılarak çekilen resmi

Artık C920 USB web kamerasının fswebcam kitaplığını kullanarak resim çekerbildiği için mükemmel çalıştığı sonucuna varabiliriz. Bu projede, kullanıcının akvaryumunu uzaktan izleyebilmesi için akvaryumun videosunu web sitesine aktarmak için kamera kullanılmaktadır. Web sitesine canlı video akışı yapmak için fswebcam kitabı kullanılamaz. Bunun nedeni, fswebcam'in yalnızca USB kameralarla resim çekmek için kullanılmasıdır. Videoyu yerel web sitesine canlı yayılmak için Pi'de bir video sunucusu barındırmamız gerekiyor. Bu projede kullanıcı arayüzüne canlı akış yapmak için UV4L Akış Sunucusu kullanılmıştır.

Bu bölümde kadar projede kullanılan donanımların genel ve teknik özelliklerinden bahsedilmiş, sensörlerin ve aktüatörlerin çalışıp çalışmadığına dair yapılan testler ve sonuçları detaylı olarak aktarılmıştır. Bu bölümde ise projede kullanılan yazılım dilleri ve mimarileri, veritabanı bileşenleri aktarılmıştır. Projede kullanılan kontrol modlarını sağlayan yazılım çevrelerinin detaylı bilgileri ve bulunan bu bilgilerin elde edildiği kaynaklar kaynakça bölümünde verilmiştir.

3.3.2 Yazılım Bileşenleri

Bu bölümde akvaryum kontrol sistemimizde bulunan yazılım bileşenleri ayrı ayrı ve ayrıntılı bir biçimde açıklanmıştır. Python programlama dili kullanılan bu projede sensörler ve diğer aktüatörleri kontrol etme amacıyla kullanılan yazılım kütüphaneleri, canlı yayın sunucusu ve arayüz ekranının tasarım kütüphanesi verilmiştir.

3.3.2.1 Python 3 (IDE)

Bu projede Raspberry Pi 3, sensör değerlerini okumak, su pompası, ışık gibi aktüatörleri kontrol etmek ve Web, Telegram mobil uygulaması ve Sesli Kontrol uygulamasından alınan komutları kontrol etmek için merkezi kontrolör olarak kullanılmıştır. Bu projede kullanılan ana programlama dili Python'dur ve Python 3 kabuğu kullanılarak hata ayıklanır. Neden Python?

Python, son yıllarda dünyanın en popüler programlama dillerinden biri haline geldi. Makine öğreniminden web siteleri oluşturmaya ve yazılım testine kadar her şeye kullanılabilir. Hem geliştiriciler hem de geliştirici olmayanlar tarafından kullanılabilir. Python nedir? Python, genellikle web siteleri ve yazılımlar oluşturmak, görevleri otomatikleştirmek ve veri analizi yapmak için kullanılan bir bilgisayar programlama dilidir. Python genel amaçlı bir dildir, yani çeşitli farklı programlar oluşturmak için kullanılabilir ve herhangi bir özel sorun için uzmanlaşmamıştır. Bu çok yönlülük, başlangıç seviyesindeki dostu olmasına birlikte, onu bugün en çok kullanılan programlama dillerinden biri haline getirdi. Endüstri analisti firması RedMonk tarafından yapılan bir anket, 2021'de geliştiriciler arasında en popüler ikinci programlama dili olduğunu buldu.

Python ile neler yapabilirsiniz? Bazı şeyler şunları içerir:

- Veri analizi ve makine öğrenimi
- Web geliştirme
- Otomasyon veya komut dosyası oluşturma
- Yazılım testi ve prototip oluşturma
- Günlük görevler

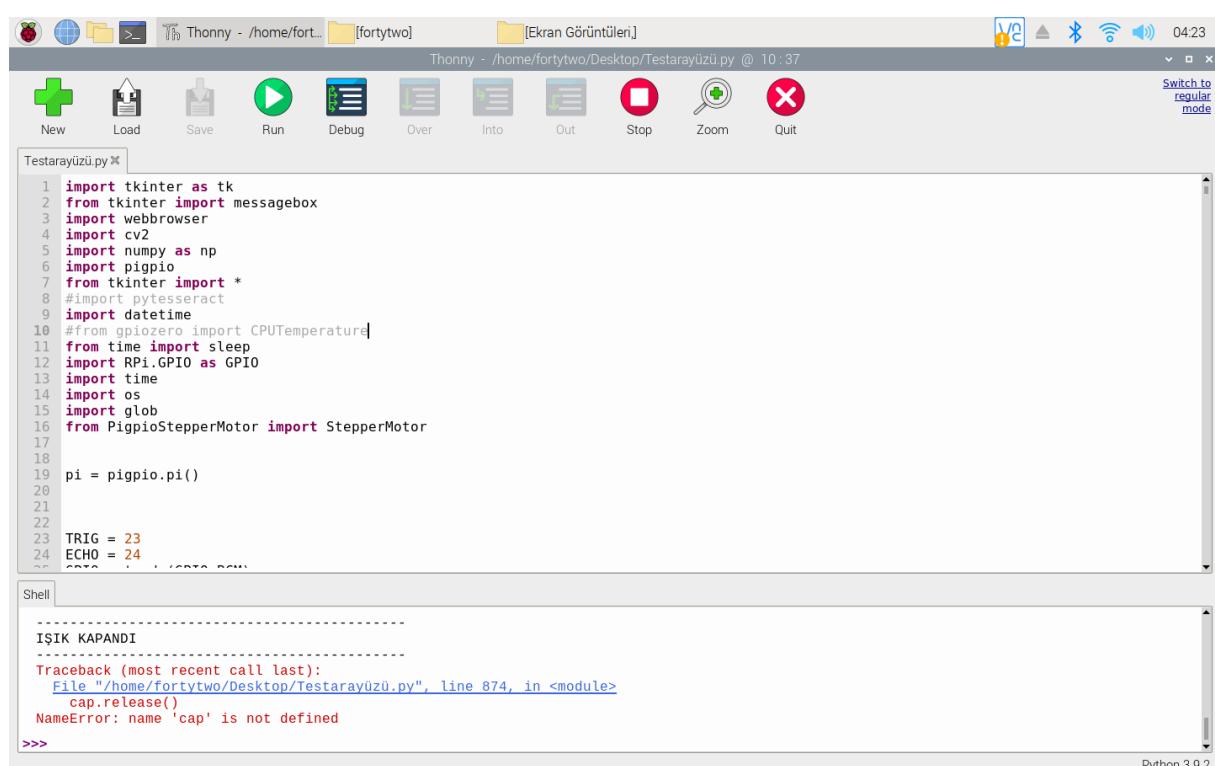
Bu projede kullanılmasının nedeni ise Raspberry Pi kartının Python yazılım dili ile muazzam bir uyum içerisinde çalışması ve çalıştırırmak istediğimiz sistemlerin Python yazılım dili üzerinden rahatça kontrol edilebilmesi örnek olarak verilebilir.

Başlangıçta Python 3 paketleri Raspberry Pi 3'ün işletim sistemi olan Raspbian ile birlikte bulunmuyor. Python'u kullanabilmek için ayrı ayrı kurmamız gerekiyor. Python 3 ide'yi almak için aşağıdaki komutu Pi'nin terminalinde yürütün.

```
sudo apt install python3 -y
```

Raspberry Pi OS'nin masaüstü sürümleri, Thonny adlı bir Python IDE ile önceden yüklenmiş olarak gelir. Bir IDE, kod yazmayı daha temiz, daha hızlı ve daha iyi bir deneyim haline getirir.

Aşağıdaki şekil 3.35'te Python arayüzü olan Thonny (IDE) ekran görüntüsü gösterilmiştir.



Sekil 64 Thonny Python IDE ekran görüntüsü

Thonny, özellikle yazılıma yeni başlayan insanlar düşünülerek tasarlanmış ücretsiz bir Python Entegre Geliştirme Ortamıdır (IDE). Spesifik olarak, kötü hatalarla

karşılaştığınızda yardımcı olabilecek yerleşik bir hata ayıklayıcıya sahiptir ve diğer harika özelliklerin yanı sıra adım adım ifade değerlendirmesi yapma yeteneği sunar.

Thonny'yi sisteminizin komut satırı aracılığıyla da yükleyebilirsiniz. Windows'ta bunu Komut İstemi adlı bir program başlatarak, macOS ve Linux'ta ise Terminal adlı bir program başlatarak yapabilirsiniz. Bunu yaptıktan sonra aşağıdaki komutu girin:

Raspberry pi kartımızın terminal ekranını açtıktan sonra aşağıdaki komutları girerek thonny'yi kurulmuş olacaktır.

\$ pip install thonny

Kullanıcı Arayüzü Thonny'nin neler sunabileceğini anladığınızdan emin olalım. Thonny'yi harika Python projeleri oluşturacağınız çalışma odası olarak düşünebiliriz. Bu projenin Python kodları Thoony IDE kullanılarak yazılmıştır.

3.3.2.2 Tkinter (Arayüz)

Grafiksel Kullanıcı Arayüzü (GUI), kullanıcıların simgeler, menüler, pencereler vb. ögeleri kullanarak görsel göstergeler aracılığıyla bilgisayarlarla etkileşim kurmasını sağlayan bir kullanıcı arayüzü biçimidir. Kullanıcıların bilgisayarlarla etkileşime girdiği Komut Satırı Arayüzü 'ne (CLI) göre avantajları vardır. Tkinter, GUI uygulamaları oluşturmak için kullanılan yerleşik python modülüdür. Çalışması basit ve kolay olduğu için Python'da GUI uygulamaları oluşturmak için en yaygın kullanılan modüllerden biridir. Zaten Python ile birlikte geldiği için Tkinter modülünün kurulumu için ayrıca endişelenmenize gerek yok. Tk GUI araç setine nesne yönelimli bir arayüz sağlar.

Aşağıdaki nedenlerden dolayı Tkinter iyi bir seçimdir:

- Öğrenmesi kolay.
- İşlevsel bir masaüstü uygulaması yapmak için çok az kod kullanın.
- Katmanlı tasarım.
- Taşınabilir Windows, macOS ve Linux dahil tüm işletim sistemlerinde kullanılabilir.
- Standart Python kitaplığı ile önceden yüklenmiştir.

Bu projede akvaryum kontrol sistemi arayüzü Python Tkinter arayüz geliştirme programı üzerinden tasarlanmıştır. Kullanıcının kolay ve anlaşılır bir arayüz ile işlem yapabilmesi için oldukça sade ve basit oluşturulan arayüz içerisinde bulunan buton, widget, label ve diğer birçok arayüz elemanı Tkinter kütüphanesinin aracılığı ile oluşturulmuştur.

3.3.2.3 UV4L Akış Sunucusu

UV4L'nin kullanıcı alanı anlamına geldiği UV4L Akış Sunucusu Video4Linux, Linux sisteminde gerçek zamanlı video yakalamayı desteklemek için bir aygit sürücüsü ve API koleksiyonudur. Şifreli canlı çift yönlü veri, ses ve video akışı ve web üzerinden konferans için kullanılabilen özelleştirilebilir bir web kullanıcı arayüzüne sahiptir. Aşağıdaki adımlar UV4L kitaplığının Pi'ye nasıl kurulacağını gösterir,

1.Adım: UV4L paketini Pi'ye kurmak için Pi'nin terminalinde aşağıdaki komutu çalıştırın;
sudo apt-get install uv4l uv4l-server uv4l-uvc uv4l-server uv4l-webrtc uv4l-xmpp-bridge

2. Adım: Yapılandırma dosyasını açın ve bir satır sürücüsünü değiştirin = uvc ve dosyayı kaydedin.

sudo vim /etc/uv4l/uv4l-raspicam.conf

Change driver = uvc

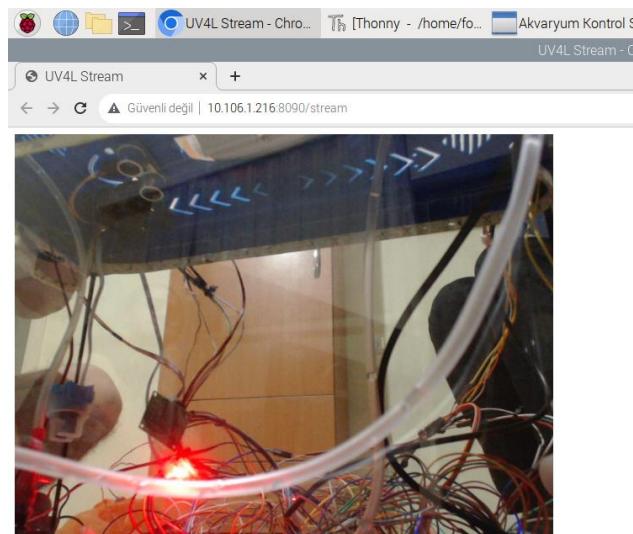
Şimdi sisteminizi yeniden başlatın ve yeniden başlatıldıktan sonra tarayıcınızı açın ve bağlantı noktası numarası 8090 olan IP yerel adresinize gidin, örneğin bizim durumumuzda **10.106.1.216:8090**. Artık UV4L Akış Sunucusunun web sayfasını görebilirsiniz. Sadece **http:// 10.106.1.216:8090/stream** adresine gidin, şimdi usb kamerasınızı kullanarak canlı akış videosunu görebilirsiniz. Aşağıda UV4L Akış Sunucusunun web sayfası gösterilmektedir.

UV4L Streaming Server



Şekil 65 UV4L

Aşağıda canlı akış videosunun ekran görüntüsü verilmiştir.



Şekil 66 Buraya canlı video yayın resmi eklenecek

Bu akvaryum kontrol sisteminde kullanıcıların akvaryumlarını canlı olarak izleyebilmeleri için kontrol sistemi arayüzüne bir kamera 'ya bağlan butonu eklenmiştir. Az önce bahsedilen UV4L canlı akış video sunucu kütüphanesinin kullanıldığı bu sistemde donanım kısmında ayrıntılı olarak açıklanmış olan web kamera aracılığı ile

kullanıcının akvaryumdan canlı görüntü alması sağlanmıştır. Ayrıca kullanıcının anlık olarak kamera ile akvaryumun fotoğrafını çekmesi amacıyla ekran görüntüsü al butonu tasarım arayüzüne eklenmiştir. Bu butonun aktif edilmesiyle kullanıcı yine UV4L video sunucusunun ekran görüntüsünü alma uzantısına gidecek ve anlık olarak akvaryumun fotoğrafını çekecektir. Çekilen bu fotoğraflar Raspberry pi'nin ana dizin ekranına kaydedilmiştir.

4. BULGULAR

Bir önceki bölümde anlatılan akvaryum kontrol projesinin kontrol modlarında çalışma çıktıları bu bölümde ayrıntılı olarak incelenmiş ve aktarılmıştır. Kullanıcıların sade ve anlaşılır bir arayüz üzerinden akvaryumlarını kontrol edebilmesi amacıyla tasarlanmış bu sistemde Python Tkinter arayüz geliştirme kütüphanesi kullanılarak geliştirilen arayüz aşağıdaki görselde verilmiştir.

4.1 Otomatik Mod



Şekil 67 Akvaryum kontrol sistemi arayüzü

Sade ve anlaşılır olması için çok fazla ayrıntı ve detaya yer verilmeyen arayüz tasarımları üzerinde bulunan butonlar yardımıyla akvaryumun uzaktan rahatlıkla kontrol edilebileceği gözlemlenmiştir.

Kullanıcının akvaryum kontrol sisteminde kullanabileceği iki kontrol modu vardır. Bunlar manuel mod ve otomatik moddur. Otomatik mod sayesinde kullanıcının dışarıdan gireceği süre sonunda akvaryum üzerinde bulunan ve step motor kullanılarak yapılan yemleme sistemi çalışacak ve akvaryum içerisinde bulunan balıklar otomatik olarak yemlenmiş olacaktır. Yapılan test ve deneylerde otomatik modun sorunsuz ve gerektiği gibi çalıştığı gözlemlenmiştir.



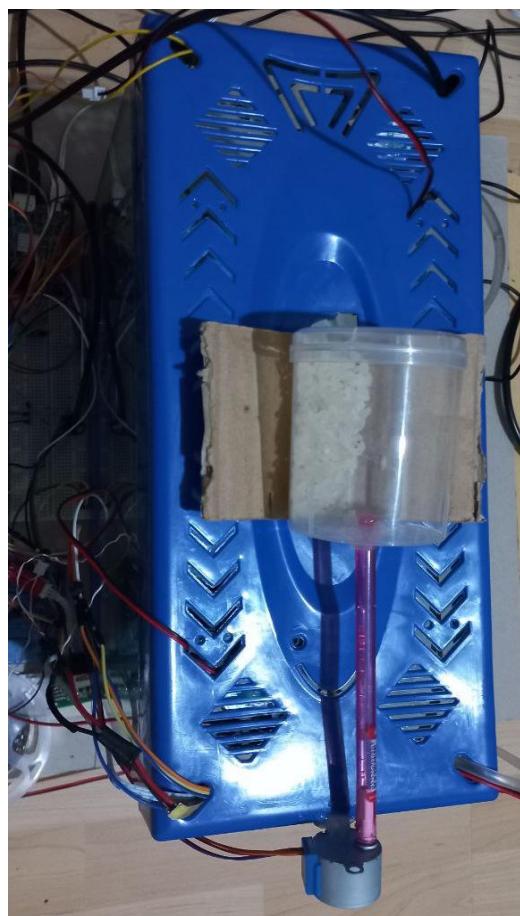
Şekil 68 1 dakika süre ayarlanan otomatik mod ekran görüntüsü

Yukarıda verilen görselde otomatik mod seçeneği tercih edilmiş ve yemleme sisteminin çalışması için “1” dakikalık süre ayarlanmıştır.

```
Shell
Dakika: 37
Sıcaklık: 22.625 Derece
-----
YEM ATILIYOR...
-----
YEM ATILDI...
```

Şekil 69 “1” dakikahk sürenin ardından çalışan otomatik yemleme sisteminin ekran çıktısı

Yukarıdaki görselde görüleceği üzere sistem dışarıdan aldığı süre değeri sonunda otomatik olarak çalışmış ve yemleme işlemi gerçekleştirılmıştır.



Şekil 70 Akvaryum üzerinde bulunan yemleme sisteminin görüntüsü

Ayrıca kullanıcıdan herhangi bir girdi almadan otomatik mod içerisinde çalışan bir diğer sistem ise sıcaklık durum kontrolüdür. Akvaryum içerisinde bulunan suyun sıcaklık değeri 30 santigrat dereceyi aştığı anda akvaryum içerisinde bulunan fanlar devreye girecek ve beş dakikalık bir periyot halinde suyu soğutmaya çalışacaktır. Bu sistemin de çalışması test edilmiş ve sistemin beklenildiği gibi çalıştığı gözlemlenmiştir.

4.2 Manuel Mod

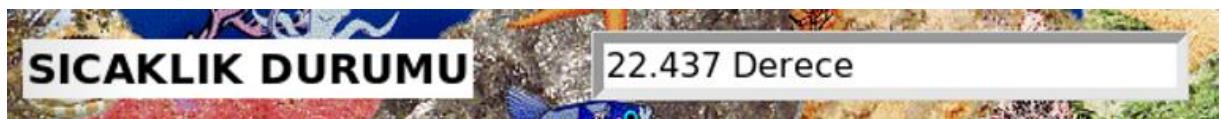
Akvaryum kontrol sisteminde bulunan bir diğer kontrol modu ise manuel moddur. Bu modda kullanıcı akvaryum üzerinde gerçekleştirmesini istediği özellikleri arayüz yardımıyla gerçekleştirecek ve çalışmasını deneyimleyecektir. Bu modun özellikleri bölümde detaylı olarak verilmiştir.

4.2.1 Sıcaklık Durumu

Bir önceki bölümde detaylı olarak bahsedilen sıcaklık sensörü tarafından alınan sıcaklık verisinin ve durum takibinin arayüz ekranı üzerinde gerçekleştirilmesi bu bölümde detaylı olarak açıklanmıştır. Akvaryum kontrol sisteminde sıcaklık takibi için DS18B20 su sıcaklık sensörü kullanılmıştır. Materyal metot bölümünde donanım ve yazılım özellikleri bahsedilen sıcaklık sensörü akvaryumun içerisinde yerleştirilmiştir. One-wire haberleşme protokolü kullanılan bu sensör üzerinden anlık olarak çekilen sıcaklık verisi akvaryum kontrol sistemi arayüzünde kullanıcının sıcaklık durumu butonuna tıklamasıyla birlikte görülecektir.



Şekil 71 Sıcaklık durum butonu arayüz üzerindeki görünümü



Şekil 72 Arayüz üzerinde bulunan sıcaklık durum bilgisi bölümü

Yukarıda verilen görseller kullanıcının sıcaklık durumu butonuna tıklaması ile birlikte arayüz ekranında bulunan sıcaklık durumu bölümü üzerine akvaryum içerisinde bulunan suyun sıcaklık verisi santigrat cinsi şeklinde geldiği gözlemlenmiştir.

4.2.2 Su seviyesi

Materyal metot bölümünde yazılım ve donanım kısımları detaylı bir biçimde anlatılan HCSR-04 ultrasonik sensör yardımıyla alınan su seviyesi değerinin arayüz üzerinden görüntülenme işlemi bu bölümde detaylı olarak verilmiştir. HCSR-04 ultrasonik mesafe sensörünün ölçüdüğü değer sensörün yerleştirilmiş olduğu akvaryum iç kapağı ile su arasında bulunan mesafedir. Bu mesafenin artması durumunda akvaryum içerisinde bulunan suyun seviyesinde bir azalma olduğu anlaşılacak, mesafenin azalması durumunda su seviyesinin yükselmiş olduğu anlaşılmuş olacaktır.

Su Seviyesi

Şekil 73 Su seviye butonu arayüz üzerinde görünümü



Şekil 74 Su seviye durum ekran görüntüsü

Yukarıda verilen görsellerde su seviye durumunun izlenilebilmesi amacıyla kullanılan buton ve durum bölümünün ekran görüntüleri görülmektedir. Test aşamasında akvaryum içerisinde su bulunmadığından dolayı 28.16 cm'lik boş akvaryum yüksekliği ölçülerek ekran çıktısı olarak gözlemlenmiştir.

4.2.3 Su sızıntı durumu

Akvaryum içerisinde bulunan su sızıntı sensörünün kullanıcı tarafından tetiklenen su sızıntı durumu butonunun aktif duruma geçmesiyle birlikte ekran uyarı arayüz üzerinde

belirecektir. Bu ekran uyarısında akvaryumda herhangi bir su sızıntısı söz konusu olduğunda su sızıntısı var veya su sızıntısı yok mesajları gösterilecektir.

Su Sızıntı Durumu

Şekil 75 Su sızıntı durum butonu arayüz ekranında görünümü



Şekil 76 Su sızıntısı olmadığından ekrana getirilen uyarı mesajı

Yukarıda verilen görselde görüleceği üzere akvaryum içerisindeki suyun azalmaması durumunda kullanıcının su sızıntı durumu butonunu aktif hale getirmesiyle ekrana gelen sızıntı yok uyarısı gözlemlenmiştir. Yapılan testler sonucunda su sızıntı sensörünün beklenildiği gibi çalıştığı gözlemlenmiş ve notlara kaydedilmiştir.

4.2.3 RGB Led kontrolü

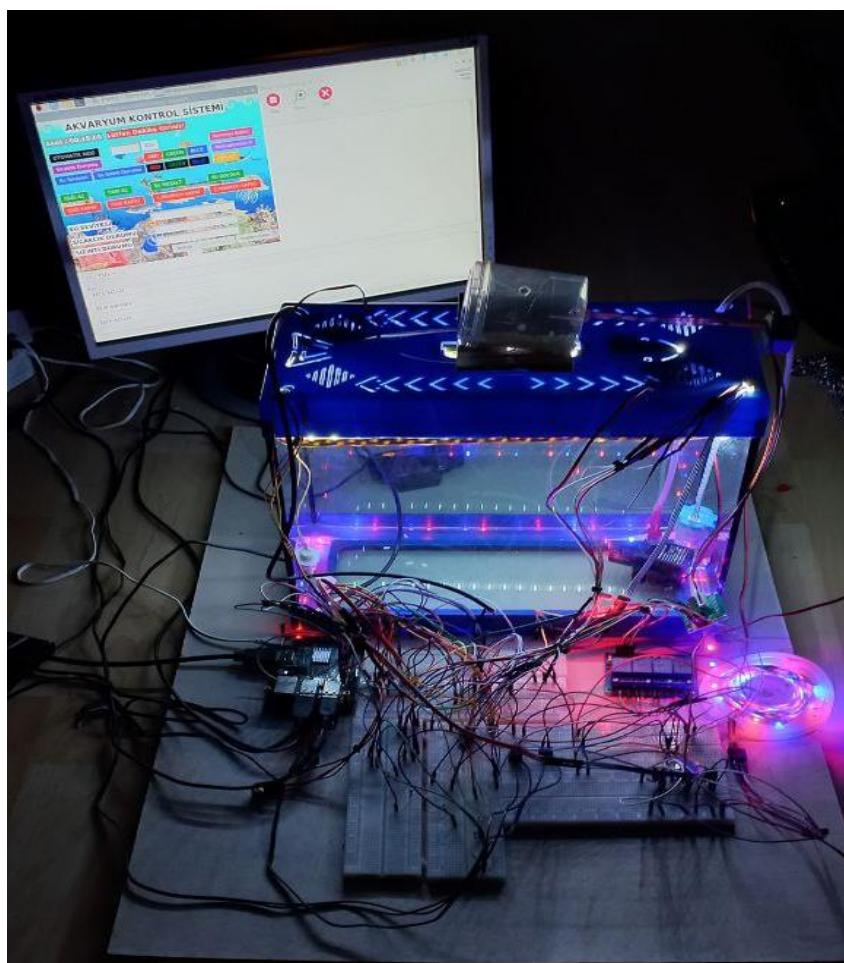
Akvaryum kontrol sistemi üzerinde görsel bir güzellik sağlaması amacıyla akvaryum tabanına yerleştirilen RGB ledlerin kontrolü bu bölümde detaylı olarak açıklanmıştır.

Akvaryum kontrol sistemi arayüzü üzerinde bulunan RED, GREEN, BLUE butonları ve bu butonların tam tersi işlemlerini gerçekleştiren siyah butonlar sayesinde kullanıcı akvaryum tabanında bulunan RGB ledleri kontrol edecektir.



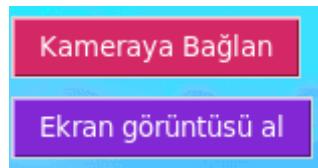
Şekil 77 RED, GREEN, BLUE led kontrol butonları

Yukarıda verilen görselde RGB kontrol butonlarının arayüz üzerinde görünümleri verilmiştir. Siyah arka plan ile oluşturulmuş butonlar ise RGB ledlerin kapatılmasını sağlayacaktır.



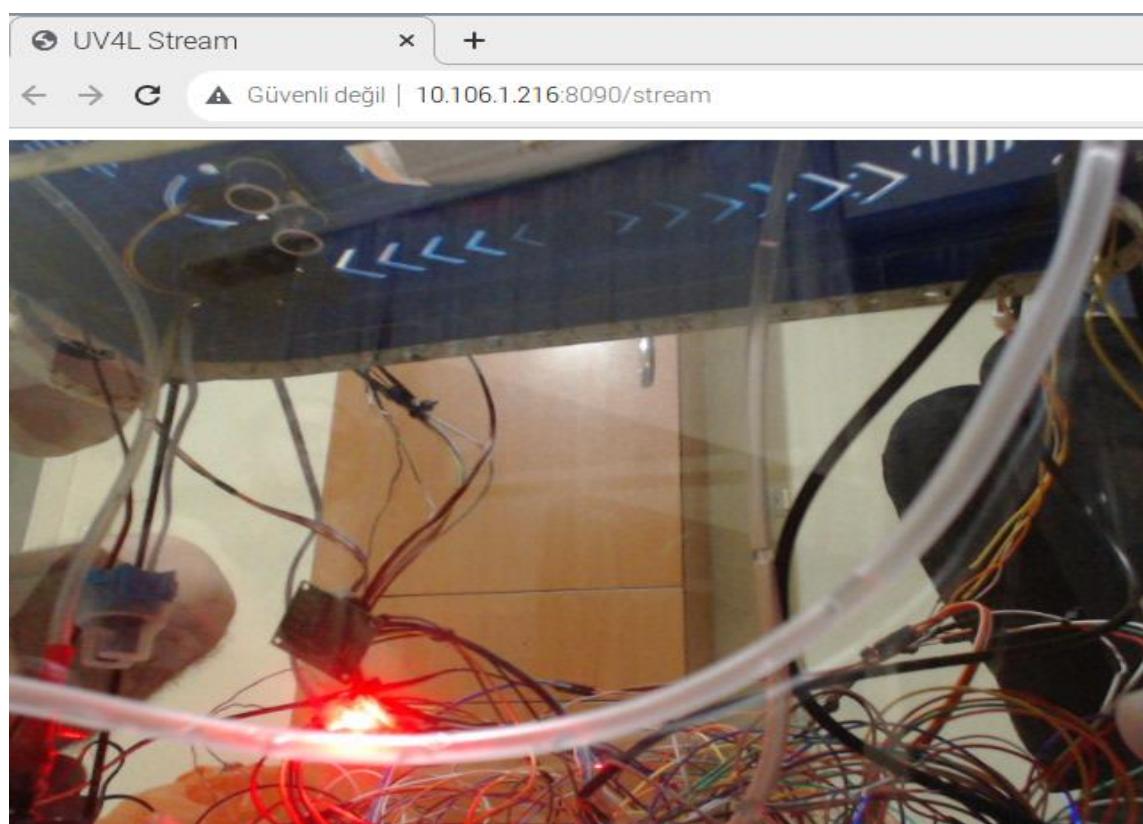
Şekil 78 RGB led akvaryum sistemi üzerinde görünümü

4.2.4 Kameraya bağlanma ve ekran görüntüsü alma



Şekil 79 Kameraya bağlanma ve ekran görüntüsü alma butonlarının arayüz üzerindeki görünümleri

Bu bölümde Raspberry pi ana kartımıza taktığımız USB kameramıza bağlanma işlemleri ayrıntılı olarak anlatılmıştır. UV4L canlı akış sunucusu kullanılan kamera işlevinde akvaryumu canlı bir şekilde izlemek isteyen kullanıcı arayüz üzerinden kameraya bağlan butonuna tıklayacak ve sunucu üzerinde akvaryumun canlı görüntüsünü izleyebilecektir. Bu işlevin eklenmesinin amacı kullanıcılarının uzaktan gece veya gündüz saatlerinde akvaryumlarını görüntüülü olarak takip edebilmelerini sağlamaktır.



Şekil 80 UV4L sunucusu üzerinde akvaryum görüntüsü

Yukarıda verilen görselde görüldüğü üzere kameraya bağlanma ve ekran görüntüsü alma işlevleri beklenildiği gibi çalışmış ve akvaryum sisteminin canlı görüntüsünü sunucu üzerinden ekranımıza getirilmiştir. Sistem beklenildiği ve amaçlandığı gibi çalışmıştır.

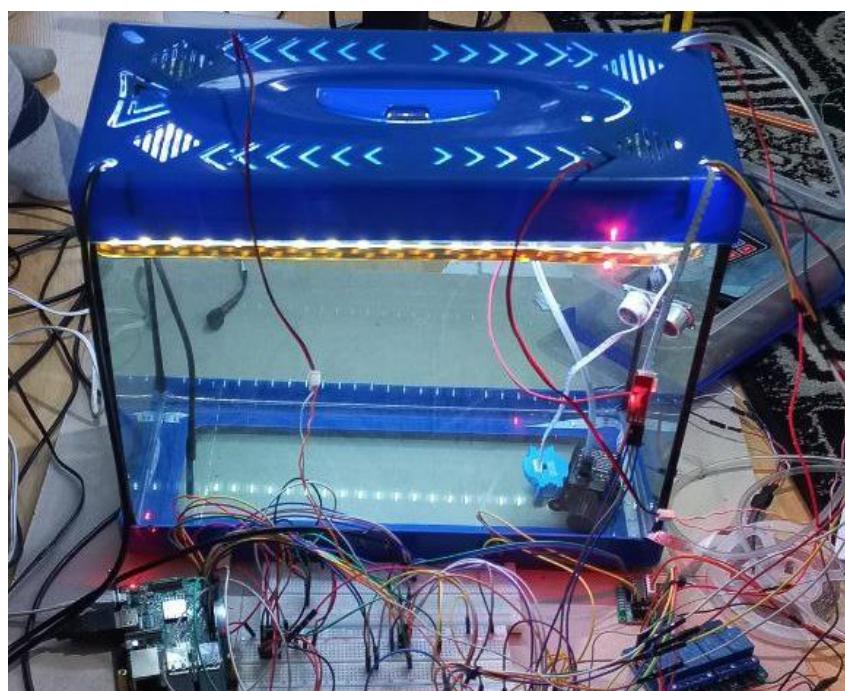
4.2.5 Beyaz ışık kontrolü

Akvaryum kontrol sistemi üzerine yerleştirilen beyaz şerit ledlerin kontrol edilebilmesi amacıyla arayüze eklenen bu işlev detaylı bir biçimde bu bölümde verilmiştir.



Şekil 81 Işığın aç ve kapat butonlarının arayüz üzerinde görünümü

Akvaryum içerisinde yerleştirilen beyaz şerit ledler kullanıcının ışığı aç ve ışığı kapat butonlarını kullanması ile açılıp kapatılacaktır.



Şekil 82 Akvaryum içerisinde bulunan beyaz şerit led çalışma görünümü

Yukarıda verilen akvaryum kontrol sistemi üzerinde beyaz şerit led çalışma görünümü görselinde de görüleceği üzere sistem beklenildiği gibi çalışmış ve kullanıcı uzaktan ışığı kontrol edebilmiştir.

4.2.6 Soğutucu Fan kontrolü

Akvaryum sistemi üzerinde bulunan ve su sıcaklığını kontrol etmesi amacıyla kullanılan soğutucu fanların kullanımı bu bölümde anlatılmıştır. Akvaryum kapağının iç kısmına yerleştirilen iki adet soğutucu fan arayüz üzerinde bulunan fani aç, fani kapat butonları kullanılarak kontrol edilecektir.



Şekil 83 Akvaryum içerisinde bulunan beyaz şerit led çalışma görünümü



Şekil 84 Akvaryum içerisinde bulunan beyaz şerit led çalışma görünümü

Kullanıcı tarafından arayüz üzerinde bulunan fani açma işlevleri gerçekleştirilmiş ve sistemin beklenildiği gibi çalıştığı gözlemlenmiştir.

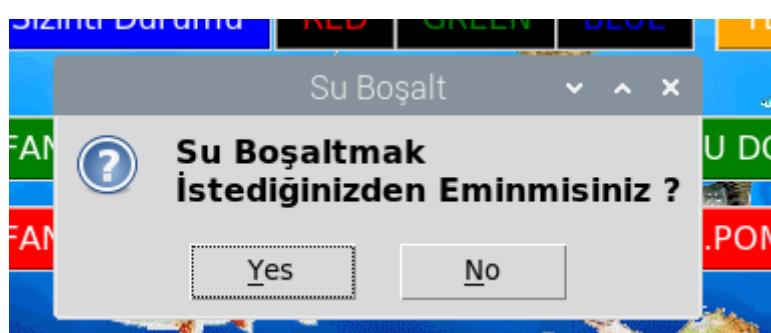
4.2.7 Su boşaltma ve doldurma pompa kontrolü

Akvaryum kontrol sisteminin en önemli özelliklerinden birisi de akvaryum içerisindeki suyu dışarıya boşaltma ve dışarıdan içeriye temiz su aktarımı işlemlerinin uzaktan kontrol edilebilmesidir. Bu amaca yönelik materyal metot kısmında bağlantıları ve yazılımı gösterilen su pompalarının çalışma şekilleri bu bölümde aktarılmıştır.



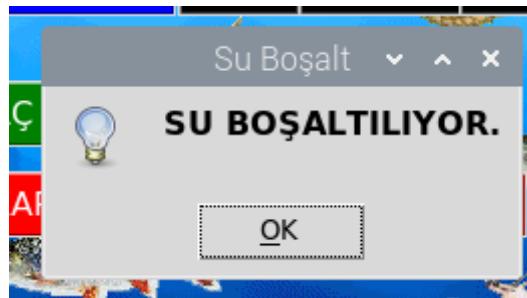
Şekil 85 Su doldurma ve boşaltma pompalarının kontrol butonlarının görünümü

Yukarıda verilen görsellerde bulunan butonların kullanıcı tarafından aktif edilmesi ile su doldurma ve boşaltma işlemleri gerçekleştirilecektir. Bu işlemler sırasıyla açıklanacak olursa;

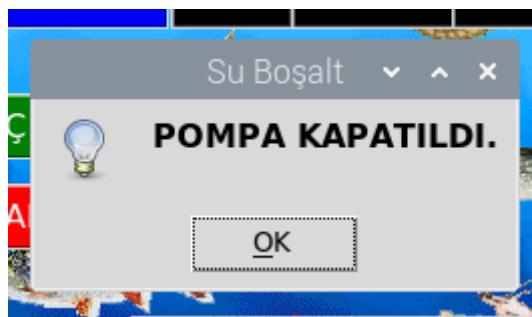


Şekil 86 Su boşaltma pompa kontrol ekran uyarı mesajı

Yukarıda verilen görsel de görüleceği üzere su boşaltma pompasının kullanıcı tarafından çalıştırılıp çalıştırılmayacağının kontrolü için ekrana bir uyarı mesajı gönderilecektir. Kullanıcının onayı alındıktan sonra sistem devreye girecek ve akvaryum içerisinde bulunan su pompa çalışmaya başlayarak akvaryum içerisindeki suyu tahliye edecektir.

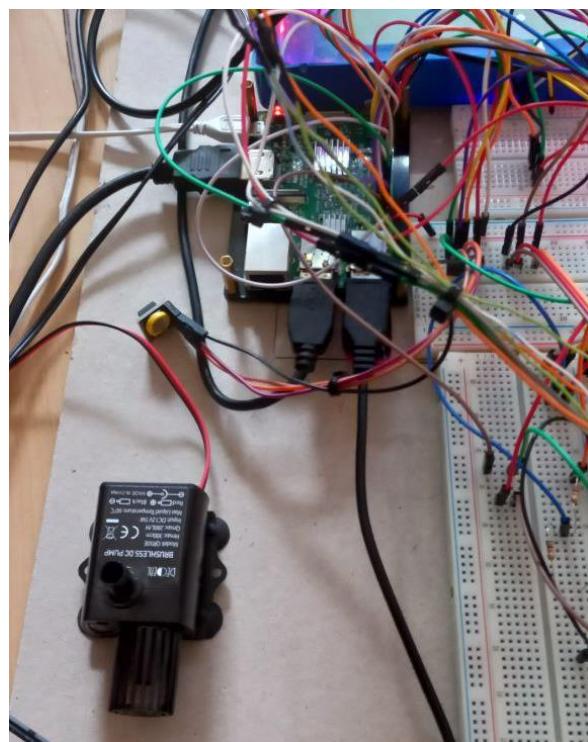


Şekil 87 Su boşaltma pompası çalışırken kullanıcının durumu takip edebilmesi amacıyla ekranda gösterilen uyarı mesajı

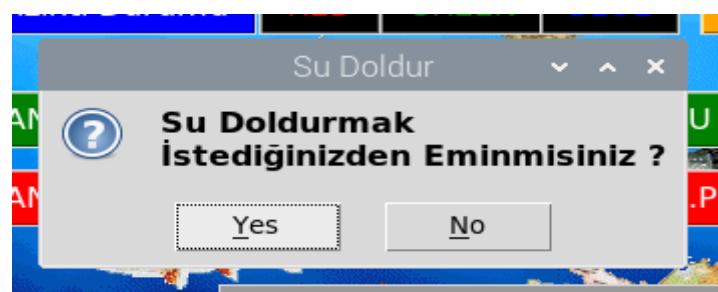


Şekil 88 Su boşaltma pompasının kapatılması durumunda ekranda gösterilecek uyarı mesajı

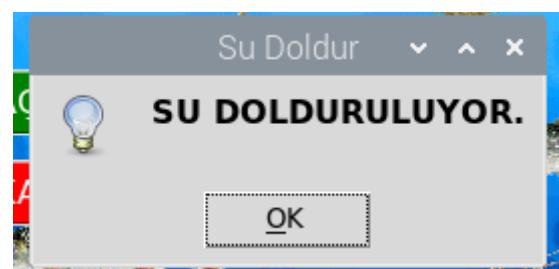
Yukarıda verilen uyarı mesajları kullanıcının pompa çalışırken durum kontrolü yapabilmesi amacıyla ekranda gösterilmiştir.



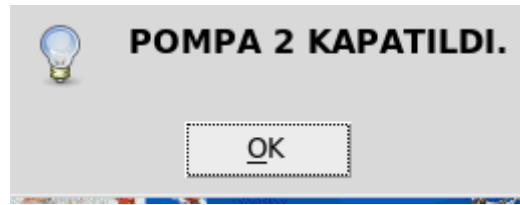
Şekil 89 Su pompa sistem üzerinde görünümü



Şekil 90 Su doldurma pompası kontrol ekranı uyarı mesajı

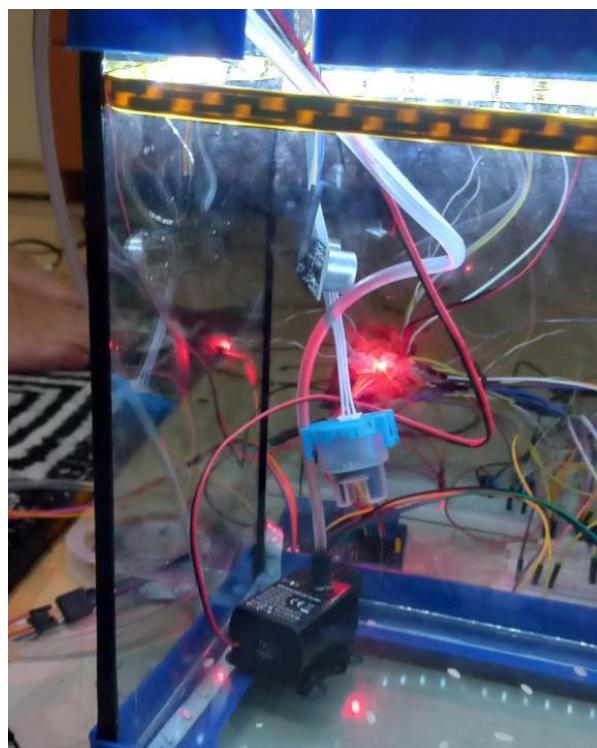


Şekil 91 Su doldurma pompası çalışırken kullanıcının durumu takip edebilmesi amacıyla ekranda gösterilen uyarı mesajı



Şekil 92 Su doldurma pompasının kapatılması durumunda ekranda gösterilecek uyarı mesajı

Yukarıda verilen görsel de görüleceği üzere su doldurma pompasının kullanıcı tarafından çalıştırılıp çalıştırılmayacağının kontrolü için ekrana bir uyarı mesajı gönderilecektir. Kullanıcının onayı alındıktan sonra sistem devreye girecek ve akvaryum içerisinde bulunan su pompa çalışmaya başlayarak akvaryum içerisinde dışarıdan su girişi sağlayacaktır.



Şekil 93 Su doldurma pompasının kapatılması durumunda ekranda gösterilecek uyarı mesajı

Yukarıda verilen işlem adımları gerçekleştirilmiş ve sistem beklenildiği gibi çalışmıştır. Su tahliye ve su doldurma işlemlerinin uzaktan pompa kontrolü ile yapılması bu projenin

en önemli özelliklerinden biridir. Bu sistemin düzgün bir biçimde çalışma kontrolü notlara kaydedilmiştir.

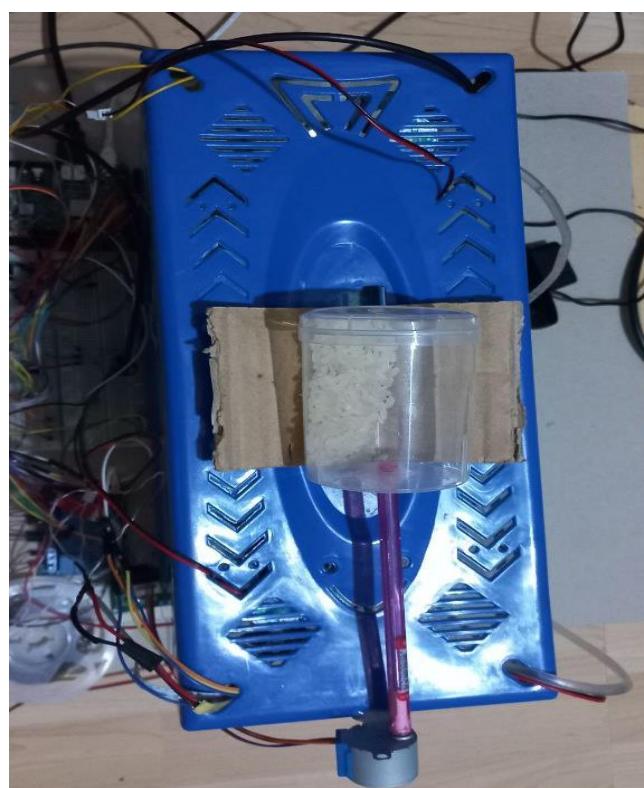
4.2.8 Manuel mod yem atma

Kullanıcının akvaryum kontrol sistemi arayüzü üzerinden akvaryumda bulunan balıkları besleyebilmesi amacıyla eklenen yem atma işlevinin özellikleri bu bölümde ayrıntılı olarak verilmiştir.



Şekil 94 Yem at butonunun arayüz üzerinde görünümü

Kullanıcı yukarıda görseli verilen butonu aktif etmesiyle birlikte akvaryum kontrol sistemi üzerinde bulunan yemleme sistemi çalışacak ve balıklar uzaktan beslenmiş olacaktır. Akvaryum kontrol sistemi üzerine yerleştirilen step motor kontrollü yemleme sistemi bu projenin en önemli özelliklerinden birisidir.



Şekil 95 Akvaryum kontrol sistemi üzerine yerleştirilen yemleme sistemi resmi

YEM ATILIYOR...

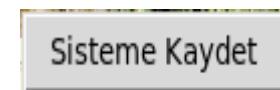
YEM ATILDI...

Şekil 96 Akvaryum kontrol sistemi üzerine yerleştirilen yemleme sistemi resmi

Yukarıda verilen işlemler gerçekleştirildiğinde kullanıcının yem at butonunu aktif etmesi sonucunda step motor yemleme sistemini döndürmüşt ve akvaryum içerisine otomatik yem atılmıştır. Sistemin beklenildiği gibi çalıştığı gözlemlenmiş ve notlara kaydedilmiştir.

4.2.9 Sisteme verileri kaydetme

Akvaryum kontrol sistemi üzerinde bulunan sıcaklık, su sızıntı ve su seviye sensörlerinden gelen verilerin gerçek zamanlı bir şekilde sisteme kaydedilmesi amacıyla sisteme kaydet butonu ve işlevi eklenmiştir.



Şekil 97 Sisteme kaydet butonu arayüz görünümü

Yukarıda görseli verilen butonun kullanıcı tarafından aktif edilmesi ile birlikte sistem üzerinde gerçek zamanlı olarak gelen veriler bir text dosyasına kaydedilecek ve ekrana bilgiler kayıt edildi şeklinde bir bildirim gelecektir. Kaydedilen bu veriler sisteme tanımlanan klasör içerisinde akvaryum_bilgileri_.txt dosyası olarak kaydedilecektir.



Şekil 98 Bilgiler kayıt edildi ekran görüntüsü

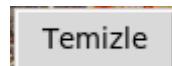


```
Testarayüzü.py akvaryum_bilgileri.txt
1 ****
2 Tarih: 03.07.2022 Saat 22.45.18
3 Su Seviyesi: 28.16 cm
4 Sicaklık Durumu: 22.437 Derece
5 Sızıntı Durumu: Sızıntı Yok
6 ****
7 ****
8 Tarih: 03.07.2022 Saat 22.56.50
9 Su Seviyesi: 29.83 cm
10 Sicaklık Durumu: 22.312 Derece
11 Sızıntı Durumu: Sızıntı Yok
12 ****
13 |
```

Şekil 99 Kaydedilen verilerin dosya içi görüntüsü

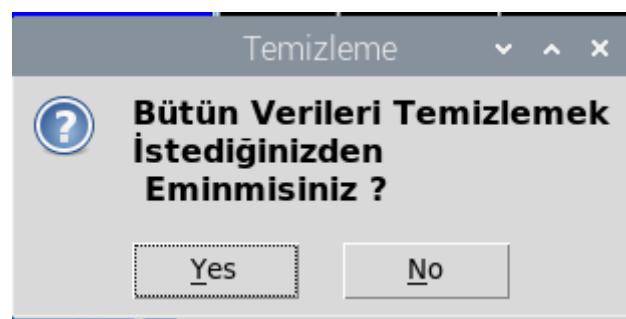
Yukarıdaki görselde de görüldüğü üzere işlev sorunsuz bir şekilde çalışmış ve sistemde bulunan veriler gerçek zamanlı şekilde kaydedilmiştir.

4.2.10 Verileri temizleme



Şekil 100 Temizle butonu arayüz görünümü

Yukarıda görseli verilen butonun kullanıcı tarafından aktif edilmesi ile birlikte sistem üzerinde verilerin temizlenecek olduğu bilgisinin kullanıcı onaya sunulduğu bir onay mesajı belirecek ardından kullanıcı onayıyla birlikte ekran üzerinde bulunan veriler temizlenecektir.



Şekil 101 Verileri temizleme onay ekranı

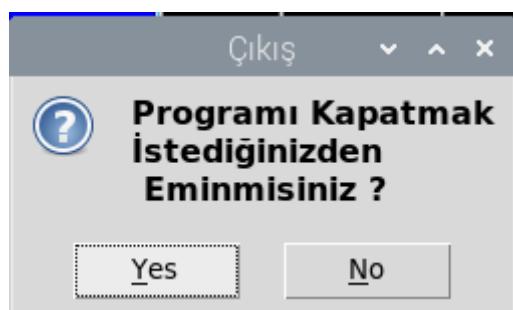


Şekil 102 Veri temizleme ekranı

Yukarıdaki işlemler gerçekleştirildiğinde ekrandaki verilerin temizlendiği görülmüş ve sistemin sorunsuz bir şekilde çalıştığı notlara kaydedilmiştir.

4.2.11 Programı Kapat

Akvaryum kontrol sisteminden çıkış yapma amacıyla eklenen programı kapat işlevinin gerçekleştirilmesi ile birlikte kullanıcı sistemden çıkış yapacaktır.



Şekil 103 Programı kapatma soru ekranı

Yukarıda verilen soru ekranında kullanıcının onayının alınması ile birlikte sistem sorunsuz bir biçimde kapatılmış ve uygulamadan çıkış yapılmıştır.

5. TARTIŞMA VE SONUÇ

Bilinen en eski akvaryumcular, en az 4500 yıl önce yapay havuzlarda balık tutan Sümerlerdi; balık tutma kayıtları da eski Mısır ve Asur'a aittir. 1000 BCE kadar erken bir tarihte yiyecek için sazan yetiştiren Çinliler, muhtemelen herhangi bir başarı ile balık yetiştiren ilk kişilerdi. Süs akvaryum balıklarının seçici üremeleri daha sonra süs sazanlarının üremesinin mükemmelleştirildiği Japonya'ya tanıtıldı. Balıkları yemek ve eğlence için besleyen antik Romalılar, bilinen ilk deniz akvaryumcularıydı; okyanustan gelen taze deniz suyuyla beslenen göletler inşa ettiler. Japon balıkları 1700'lerin ortalarında İngiltere'de cam kaplarda başarılı bir şekilde tutulmuş olsa da oksijen, hayvanlar ve bitkiler arasındaki ilişki bir yüzyıl sonra keşfedilene kadar akvaryum bakımını tam olarak kurulmamıştı. Büyüklük bir akvaryumun tasarıımı, özellikle modern akvaryumlardaki sergiler her tür su organizmasını içerdiginden, örneklerin gereksinimlerini dikkate almalıdır: memeliler, kuşlar, sürüngenler, amfibiler ve omurgasızlar ve balıklar. Dikkate alınması gereken birçok faktör arasında, ziyaretçilerin trafik akış düzenleri, camdan yansımalar, akustik ve su berraklısı, çözünmüş atıklar, sıcaklık, tank dekoru, hastalık tedavisi ve beslenme gibi tank bakım sorunları yer almaktadır.

Giriş bölümünde ifade edildiği gibi insanların günümüz dünyasında vakitlerini oldukça verimli kullanması önemlidir. Gün içerisinde birçok yoğun telaşın ve koşturmacanın arasında evlerinde hobi olarak besledikleri balıklarının bakımlarını kolaylaşturma amacıyla bu sistem geliştirilmiştir.

Sistemde kullanılan donanım ve yazılımlar materyal ve metod bölümünde detaylı olarak verilmiştir. Sistemin çalışma mantığı, ekran görüntüleri, test görüntüleri ve kullanıcı kontrol durumları bulgular bölümünde detaylı olarak verilmiştir.

Literatür taraması bölümünde verilen benzer projeler göz önüne alındığında bu sistemde birçok geliştirme yapılabılır ve sistem hem donanım hem yazılım olarak üst düzey bir görünümme kavuşabilir.

Bu çalışma ile kullanıcıların akvaryumlardaki sensörleri ve aktüatörleri uzaktan sade ve basit bir arayüzü aracılığı ile kontrol edebileceği gösterilmiştir. Kullanılan ana kart Raspberry pi model 3B bu çalışmada önemli bir rol oynamış ve sistemin sorunsuz bir şekilde çalışmasını sağlamıştır.

6. KAYNAKLAR

- A. K. Pasha Mohd Daud, N. A. Sulaiman, Y. W. Mohamad Yusof and M. Kassim, "An IoT-Based Smart Aquarium Monitoring System," 2020 IEEE 10th Symposium on Computer Applications & Industrial Electronics (ISCAIE), Malaysia, 2020, pp. 277-282, doi: 10.1109/ISCAIE47305.2020.9108823.
- Abdullah, Nur Dalila, et al. Smart Feeder Monitoring Devices with Mobile Application. Journal of Design for Sustainable and Environment, 2019, 1.1.
- AFIAH, Yasmine; ROSADI, Rizal Abdulrozaq; HAFIZ, Mohammad Raihan. The smart monitoring and automation control system for fish aquarium based on internet of things technology. In: AIP Conference Proceedings. AIP Publishing LLC, 2019. p. 030018.
- Akila, I. S., Karthikeyan, P., Hari, H. M. V., & Hari, K. J. (2018). IoT Based Domestic Fish Feeder. 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA). doi:10.1109/iceca.2018.8474829
- Ali M. Jasim, Jafer Mohammed, Ahmed Ali, Ali Ibrahim, Mohammed J. Noori, Hazem M. Ali, 2021 Implementation of an Automated Fish Control and Monitoing System Based on Internet of Things Technique. Journal of Global Scientific Research, 1643-1651.
- Chi, M.C., 2010. A Multi-functional Aquarium Equipped with Automatic Thermal Control/Fodder-Feeding/water Treatment using a Network Remote Control System. Information Technology Journal, 9(7), pp.1458-1466
- Chi, M.C., 2010. A Multi-functional Aquarium Equipped with Automatic Thermal Control/Fodder-Feeding/water Treatment using a Network Remote Control System. Information Technology Journal, 9(7), pp.1458-1466
- F. Budiman, M. Rivai and M. A. Nugroho, "Monitoring and Control System for Ammonia and pH Levels for Fish Cultivation Implemented on Raspberry Pi 3B," 2019 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, 2019, pp. 68-73, doi: 10.1109/ISITIA.2019.8937217.
- Fatmawati, Kiki, et al. Rancang Bangun Tempat Sampah Pintar Menggunakan Sensor Jarak Berbasis Mikrokontroler Arduino. Riau Journal Of Computer Science, 2020, 6.2: 124-134.

- H. N. Vishwas and S. Ullas, "Remotely Monitored Energy EfficientMethod for Aquaculture Using Smart Devices," 2019 International Conference on Advances in Computing, Communication and Control (ICAC3), Mumbai, India, 2019, pp. 1-6, doi: 10.1109/ICAC347590.2019.9036810.
- I. Daut, M. Irwanto and S. Hardi, "Photovoltaic powered uninterruptible power supply using smart relay," 2010 4th International Power Engineering and Optimization Conference (PEOCO), Shah Alam, 2010, pp. 453-457, doi: 10.1109/PEOCO.2010.5559175.
- Irawan, Y., Fernando, Y., & Wahyuni, R. Detecting Heart Rate Using Pulse Sensor As Alternative Knowing Heart Condition. Journal of Applied Engineering and Technological Science (JAETS),2019, 1(1), pp 30-42.
- Irawan, Yuda. "Implementation Of Data Mining For Determining Majors Using K-Means Algorithm In Students Of SMA Negeri 1 Pangkalan Kerinci." Journal of Applied Engineering and Technological Science (JAETS) 1.1 (2019): 17-29.
- Irawan, Yuda. Moving Load Robot Using Wifi Network and Android Based. Journal of Robotics and Control (JRC), 2020, 2.3: 217-220.
- Irawan, Yuda. Penerapan Data Mining Untuk Evaluasi Data Penjualan Menggunakan Metode Clustering Dan Algoritma Hirarki Divisive Di Perusahaan Media World Pekanbaru. Jurnal Teknologi Informasi Universitas Lambung Mangkurat (JTIULM), 2019, 4.1: 13-20.
- Irawan, Yuda; Wahyuni, Refni; Fonda, Hendry. Folding Clothes Tool Using Arduino Uno Microcontroller And Gear Servo. Journal of Robotics and Control (JRC), 2020, 2.3: 170-174.
- K. J. Shin, A. V. Angani and M. Akbar, "Fully automatic fluid flow control system for smart vertical aquarium," 2017 International Conference on Applied System Innovation (ICASI), Sapporo, 2017, pp. 424-427, doi: 10.1109/ICASI.2017.7988443.
- KIM, Yuhwan, et al. Realization of IoT based fish farm control using mobile app. In: 2018 International Symposium on Computer, Consumer and Control (IS3C). IEEE, 2018. p. 189-192.
- Lin, Y.-B., & Tseng, H.-C. (2019). FishTalk: An IoT-based Mini Aquarium System. IEEE Access, 1–1. doi:10.1109/access.2019.2905017

- Muhardi, Muhardi, et al. "Design Of Web Based LMS (Learning Management System) in SMAN 1 Kampar Kiri Hilir." Journal of Applied Engineering and Technological Science (JAETS) 1.2 (2020): 70-76.
- N. E. Cater, P. Eng. and T. O'Reilly, "Promoting interoperable ocean sensors the Smart Ocean Sensors Consortium," OCEANS 2009, Biloxi, MS, 2009, pp. 1-6, doi: 10.23919/OCEANS.2009.5422448.
- N. H. Harani, A. S. Sadiah and A. Nurbasari, "Smart Fish Feeder Using Arduino Uno With Fuzzy Logic Controller," 2019 5th International Conference on Computing Engineering and Design (ICCED), Singapore, Singapore, 2019, pp. 1-6, doi: 10.1109/ICCED46541.2019.9161114.
- Noor, M.Z.H., Hussian, A.K., Saaid, M.F., Ali, M.S.A.M. and Zolkapli, M., 2012, July.
- Noor, M.Z.H., Hussian, A.K., Saaid, M.F., Ali, M.S.A.M. and Zolkapli, M., 2012, July. The design and development of automatic fish feeder system using PIC microcontroller. In Control and System Graduate Research Colloquium (ICSGRC), 2012 IEEE (pp. 343- 347). IEEE
- Nurliani Hidayah Ritonga; Agung Nugroho Jati; Rifki Wijaya, 2016, May. Automatic Arowana Raiser Controller Using Mobile Application Based on Android. In 2016 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob) (pp.86-87). IEEE.
- Ordila, Rian, et al. Penerapan Alat Kendali Kipas Angin Menggunakan Microcontroller Arduino Mega 2560 Dan Sensor Dht22 Berbasis Android. Riau Journal Of Computer Science, 2020, 6.2: 101-106.
- Pankhurst, N.W., King, H.R. (2010). Temperature and salmonid reproduction: Implications for aquaculture. Journal of Fish Biology, pp:69-85.
- Pasha Mohd Daud, Ahmad Kamal; Sulaiman, Norakmar Arbain; Mohamad Yusof, Yuslinda Wati; Kassim, Murizah (2020). [IEEE 2020 IEEE 10th Symposium on Computer Applications & Industrial Electronics (ISCAIE) - Malaysia (2020.4.18-2020.4.19)] 2020 IEEE 10th Symposium on Computer Applications & Industrial Electronics (ISCAIE) - An IoT-Based Smart Aquarium Monitoring System. , (), 277–282. doi:10.1109/ISCAIE47305.2020.9108823
- Prasad, A.N., Mamun, K.A., Islam, F.R. and Haqva, H., 2015, December. Smart water

- quality monitoring system. In 2015 2nd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE) (pp. 1-6). IEEE.
- Saha, S., Rajib, R. H., & Kabir, S. (27-28 October 2018). IoT Based Automated Fish Farm Aquaculture Monitoring System. Int. Conf. on Innovations in Science, Engineering and Technology (ICISET).
- Sarkar, B., B.C. Mohapatra, S.K. Singh, D. Mahji, N.Sarangi and G.N. Tiwari. (2007). Impact on energy consumption in greenhouse fish production. Asian Journal of Agriculture Res., 1:74-79.
- Sohor, Suherman, Et Al. Rancang Bangun Tempat Sampah Otomatis Menggunakan Mikrokontroler Dan Sensor Ultasonik Dengan Notifikasi Telegram. Jurnal Ilmu Komputer, 2020, 9.2: 154-160.
- System. Information Technology Journal, 9(7), pp.1458-1466
- T. I. Salim, T. Haiyunnisa and H. S. Alam, "Design and implementation of water quality monitoring for eel fish aquaculture," 2016 International Symposium on Electronics and Smart Devices (ISESD), Bandung, 2016, pp. 208-213, doi: 10.1109/ISESD.2016.7886720.
- Taotao Xu, ; Feng Chen, (2014). [IEEE 2014 IEEE Workshop on Electronics, Computer and Applications (IWECA) - Ottawa, ON, Canada (2014.5.8-2014.5.9)] 2014 IEEE Workshop on Electronics, Computer and Applications - An embedded fuzzy decision system for aquaculture. , (), 351–353. doi:10.1109/IWECA.2014.6845629
- The design and development of automatic fish feeder system using PIC microcontroller.In Control and System Graduate Research Colloquium (ICSGRC), 2012 IEEE (pp. 343-347). IEEE.
- Tolentino, Lean Karlo S., et al. Development of an IoT-based Intensive Aquaculture Monitoring System with Automatic Water Correction. International Journal of Computing and Digital Systems, 2020, 9: 1-11.
- Ulum, M., et al. Smart aquaponic system based Internet of Things (IoT). In: Journal of Physics: Conference Series. IOP Publishing, 2019. p. 012047.
- Vijayakumar, N. and Ramya, R., 2015, March. The real time monitoring of water qualityin IoT environment. In Innovations in Information, Embedded and

Communication Systems (ICIIIECS), 2015 International Conference on(pp. 1-5).
IEEE

Wahyuni, Refni; Irawan, Yuda. Web-Based Heart Disease Diagnosis System With Forward Chaining Method (Case Study Of Ibnu Sina Islamic Hospital). Journal Of Applied Engineering And Technological Science (Jaets), 2019, 1.1: 43-50.

Internet Kaynakları

- 1- <https://www.elprocus.com/hc-sr04-ultrasonic-sensor-working-and-its-applications/#:~:text=The%20HC%20DSR04%20Ultrasonic%20sensor%20comes%20with%20four%20pins%20namely,works%20on%20the%20sound%20waves>.
- 2- <https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/>
- 3- <https://www.electronics-tutorials.ws/dccircuits/voltage-divider.html>
- 4- <https://www.klimexs.com/evaporatif-sogutucu-nedir-nasil-calisir/>
- 5- <https://redmonk.com/sogrady/2021/08/05/language-rankings-6-21/>
- 6- <https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python>
- 7- <https://raspberry-valley.azurewebsites.net/UV4L/>
- 8- <https://realpython.com/python-thonny/>
- 9- <https://www.britannica.com/science/aquarium/Maintenance-problems>
- 10-<https://journal.umy.ac.id/index.php/jrc/article/view/10187>
- 11-<http://www.merriam-webster.com/dictionary/aquarium>
- 12-<http://er.yuvayana.org/sdlc-prototype-model-design-advantagesdisadvantages-and-applications/>
- 13-<http://searchsoftwarequality.techtarget.com/definition/systems-development-life-cycle>
- 14-<http://www.libelium.com/products/wasp mote/>
- 15-<http://theindianeye.net/west-coast-news/divine-vastu-tips-for-fishaquariuma-good-remedial-measure-for-vastu-defect>
- 16-http://fengshui.lovetoknow.com/Feng_Shui_Lucky_Number_of_Fish_in_a_Tank

17-http://www.myduino.com/index.php?route=product/product&product_id=869&search=t

18-http://www.myduino.com/index.php?route=product/product&product_id=496&search=

19-<http://www.lelong.com.my/dc12v-4-8w-mini-brushless-submersible-water-pump-fishtank-thestarbuy-174271482-2017-02-Sale-P.html>

20-<http://www.gartner.com/it-glossary/internet-of-things/> (

21-<http://dordnung.de/raspberrypi-ledstrip/>

EK 1:

Projenin Python kodları bu bölümde verilmiştir.

```
import tkinter as tk  
  
from tkinter import messagebox  
  
import webbrowser  
  
import cv2  
  
import numpy as np  
  
import pigpio  
  
from tkinter import *  
  
#import pytesseract  
  
import datetime  
  
from time import sleep  
  
import RPi.GPIO as GPIO  
  
import time  
  
import os  
  
import glob  
  
from PigpioStepperMotor import StepperMotor
```

```
pi = pigpio.pi()
```

TRIG = 23

ECHO = 24

```
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setwarnings(False)
```

```
GPIO.setup(TRIG,GPIO.OUT)
```

```
GPIO.setup(ECHO,GPIO.IN)
```

```
A = 5
```

```
B = 6
```

```
C = 13
```

```
D = 19
```

```
GPIO.setup(14,GPIO.OUT) #Relay IN4 for Ligh
```

```
GPIO.setup(15,GPIO.OUT) #Relay IN3 for Pump_1
```

```
GPIO.setup(7,GPIO.OUT) #Relay IN2 for Pump_2
```

```
GPIO.setup(25,GPIO.OUT) #Relay IN1 for Cooler_Fan
```

```
GPIO.setup(26,GPIO.IN) #For water leak
```

```
#To read Aquarium WAtter Level
```

```
#-----
```

```
#def read_AquaWaterLevel():
```

```
GPIO.output(14,GPIO.HIGH)
```

```
GPIO.output(15,GPIO.HIGH)
```

```
GPIO.output(7,GPIO.HIGH)  
GPIO.output(25,GPIO.HIGH)  
  
GPIO.setup(A, GPIO.OUT) #  
GPIO.setup(B, GPIO.OUT) #For Stepper Motor  
GPIO.setup(C, GPIO.OUT) #  
GPIO.setup(D, GPIO.OUT) #
```

```
def GPIO_SETUP(a,b,c,d):  
    GPIO.output(A, a)  
    GPIO.output(B, b)  
    GPIO.output(C, c)  
    GPIO.output(D, d)  
    time.sleep(0.001)
```

```
def Fish_Feeder_RTURN(deg):  
  
    full_circle = 510.0  
    degree = full_circle/360*deg  
    GPIO_SETUP(0,0,0,0)
```

```
while degree > 0.0:  
    GPIO_SETUP(1,0,0,0)
```

```
GPIO_SETUP(1,1,0,0)  
GPIO_SETUP(0,1,0,0)  
GPIO_SETUP(0,1,1,0)  
GPIO_SETUP(0,0,1,0)  
GPIO_SETUP(0,0,1,1)  
GPIO_SETUP(0,0,0,1)  
GPIO_SETUP(1,0,0,1)  
degree -= 1  
GPIO_SETUP(0,0,0,0)
```

```
def Fish_Feeder_LTURN(deg):
```

```
    full_circle = 510.0  
    degree = full_circle/360*deg  
    GPIO_SETUP(0,0,0,0)
```

```
while degree > 0.0:
```

```
    GPIO_SETUP(1,0,0,1)  
    GPIO_SETUP(0,0,0,1)  
    GPIO_SETUP(0,0,1,1)  
    GPIO_SETUP(0,0,1,0)  
    GPIO_SETUP(0,1,1,0)  
    GPIO_SETUP(0,1,0,0)  
    GPIO_SETUP(1,1,0,0)  
    GPIO_SETUP(1,0,0,0)
```

```

degree -= 1

GPIO_SETUP(0,0,0,0)

#motor = StepperMotor(29,31,33,35)

os.system('modprobe w1-gpio')
os.system('modprobe w1-therm')

base_dir = '/sys/bus/w1/devices/'

device_folder = glob.glob(base_dir + '28-3c8af648382f')[0] #serial number of my sensor
device_file = device_folder + '/w1_slave'

pi.set_PWM_dutycycle(17, 255)
pi.set_PWM_dutycycle(22, 255)

pencere = tk.Tk()

img = PhotoImage(file="/home/fortytwo/Desktop/image3")
fotograf = Label(pencere,image=img)
fotograf.place(x=0, y=0)

entry =
tk.Entry(state='disabled',bd=5,disabledbackground="white",disabledforeground="black",width=27,font="bold")
entry.place(x=280, y=366)

```

```
entry2 =  
tk.Entry(state='disabled',bd=5,disabledbackground="white",disabledforeground="black",width=27,font="bold")  
  
entry2.place(x=280, y=416)
```

```
entry3 =  
tk.Entry(state='disabled',bd=5,disabledbackground="white",disabledforeground="black",width=27,font="bold")  
  
entry3.place(x=280, y=460)
```

```
entry4 =  
tk.Entry(bd=5,disabledbackground="white",disabledforeground="black",width=8,font="bold")  
  
entry4.place(x=200, y=110)
```

```
#####
#####
```

```
#####
#####
```

```
#Kamera
```

```
new = 1  
  
url = "http://10.106.1.216:8090/stream"  
  
def openweb():  
  
    webbrowser.open(url, new=new)
```

```
#####
url2 = "http://10.106.1.216:8090/stream/snapshot.jpeg?delay_s=0"
new = 2
def ekran_goruntusu():
    webbrowser.open(url2, new=new)
```

```
#####
#YEMLEME SİSTEMİ
```

```
def yem_at():
    """
    GPIO.setup(A, GPIO.OUT) #
    GPIO.setup(B, GPIO.OUT) #For Stepper Motor
    GPIO.setup(C, GPIO.OUT) #
    GPIO.setup(D, GPIO.OUT) #
```

```
def GPIO_SETUP(a,b,c,d):
    GPIO.output(A, a)
    GPIO.output(B, b)
    GPIO.output(C, c)
    GPIO.output(D, d)
    time.sleep(0.001)
```

```
def Fish_Feeder_RTURN(deg):
```

```
    full_circle = 510.0
```

```
    degree = full_circle/360*deg
```

```
    GPIO_SETUP(0,0,0,0)
```

```
    while degree > 0.0:
```

```
        GPIO_SETUP(1,0,0,0)
```

```
        GPIO_SETUP(1,1,0,0)
```

```
        GPIO_SETUP(0,1,0,0)
```

```
        GPIO_SETUP(0,1,1,0)
```

```
        GPIO_SETUP(0,0,1,0)
```

```
        GPIO_SETUP(0,0,1,1)
```

```
        GPIO_SETUP(0,0,0,1)
```

```
        GPIO_SETUP(1,0,0,1)
```

```
        degree -= 1
```

```
    GPIO_SETUP(0,0,0,0)
```

```
def Fish_Feeder_LTURN(deg):
```

```
    full_circle = 510.0
```

```
    degree = full_circle/360*deg
```

```
    GPIO_SETUP(0,0,0,0)
```

```
while degree > 0.0:  
    GPIO_SETUP(1,0,0,1)  
    GPIO_SETUP(0,0,0,1)  
    GPIO_SETUP(0,0,1,1)  
    GPIO_SETUP(0,0,1,0)  
    GPIO_SETUP(0,1,1,0)  
    GPIO_SETUP(0,1,0,0)  
    GPIO_SETUP(1,1,0,0)  
    GPIO_SETUP(1,0,0,0)  
    degree -= 1  
    GPIO_SETUP(0,0,0,0)
```

""""

```
while True:
```

```
    print("-----")  
    print("YEM ATILYOR...")  
    print("-----")  
    Fish_Feeder_LTURN(90)  
    time.sleep(0.1)  
    print("YEM ATILDI...")  
    print("-----")  
    Fish_Feeder_LTURN(0)  
    time.sleep(0.1)
```

```
Fish_Feeder_RTURN(120)
```

```
time.sleep(0.1)

break

#####
def gir():

    b16['state']='normal'

#####

#####

def otoMod():

    dakika_degeri=int(entry4.get())

    na = datetime.datetime.now()

    yeni = int(na.strftime("%M"))

    while 1:

        x = read_temp()

        an = datetime.datetime.now()

        saat = int(an.strftime("%M"))

        #print(type(an))

        #print(type(saat))

        print("-----")

        print("Saat: ",(an))

        print("Dakika: ",(saat))

        print("Sicaklik: "+str(x)+" Derece")

        print("-----")

        time.sleep(2)
```

```

if(x > 30.0):

    print("Çok Sıcak.. Fan Açılıyor...")

    while True:

        GPIO.setup(25,GPIO.OUT) #Relay IN4 for Ligh

        print("-----")
        print("FAN AÇILDI")

        GPIO.output(25,GPIO.LOW)

        time.sleep(60)

        GPIO.output(25,GPIO.HIGH)

        print("FAN KAPANDI")

        print("-----")

        break

    break

elif(saat == yeni + dakika_degeri):

    while True:

        print("-----")
        print("YEM ATILIYOR...")
        print("-----")

        Fish_Feeder_LTURN(90)

        time.sleep(0.1)

        print("YEM ATILDI...")

```

```

print("-----")
Fish_Feeder_LTURN(0)
time.sleep(0.1)

Fish_Feeder_RTURN(120)
time.sleep(0.1)
break

messagebox.showinfo("Otomod","OTOMOD KAPATILDI")
entry4['state']='normal'
entry4.delete(0,tk.END)
entry4['state']='disabled'
b16['state']='disabled'
break

#####
# RGB LED

#Kirmizi

def kirmiziAc():
    #pi.set_PWM_dutycycle(27, 0) # Red kanalı 16
    #pi.set_PWM_dutycycle(22, 0)
    pi.set_PWM_dutycycle(17, 255)
    ....
while True:

```

```

print("Red RGB is on")

kirmiziAc()

break

"""

#Yesil

def yesilAc():

    pi.set_PWM_dutycycle(27, 255)

    #pi.set_PWM_dutycycle(22, 0) # Green Kanalı 20

    #pi.set_PWM_dutycycle(17, 0)

"""

while True:

    print("Green RGB is on")

    yesilAc()

    break

"""

#Mavi

def maviAc():

    #pi.set_PWM_dutycycle(27, 0)

    pi.set_PWM_dutycycle(22, 255)

    #pi.set_PWM_dutycycle(17, 0) # Blue Kanalı 21

"""

while True:

    print("Blue RGB is on")

    maviAc()

    break

```

:::::

#Kirmizikapat

```
def kirmiziKapat():
    #pi.set_PWM_dutycycle(27, 0)
    #pi.set_PWM_dutycycle(22, 0)
    pi.set_PWM_dutycycle(17, 0)
```

#Yesilkapat

```
def yesilKapat():
    pi.set_PWM_dutycycle(27, 0)
    #pi.set_PWM_dutycycle(22, 0)
    #pi.set_PWM_dutycycle(17, 0)
```

#Mavi

```
def maviKapat():
    #pi.set_PWM_dutycycle(27, 0)
    pi.set_PWM_dutycycle(22, 0)
    #pi.set_PWM_dutycycle(17, 0)
```

#####

def read_temp_raw():

```

f = open(device_file, 'r')

lines = f.readlines()

f.close()

return lines


def read_temp():

    lines = read_temp_raw()

    while lines[0].strip()[-3:] != 'YES':

        #time.def yem_at():sleep(0.2)

        lines = read_temp_raw()

        equals_pos = lines[1].find('t=')

        if equals_pos != -1:

            temp_string = lines[1][equals_pos+2:]

            temp_c = float(temp_string) / 1000.0

            #temp_f = temp_c * 9.0 / 5.0 + 32.0

            return temp_c

#####
#



def sistemi_kapat():

    cevap=messagebox.askyesno("Çıkış","Programı Kapatmak İstedığınızden\nEminmisiniz")

    if cevap==1:

        exit()

    else:

```

```
pass
```

```
#####
#
```

```
def suSeviyesi():
```

```
    entry['state']='normal'
```

```
    entry.delete(0,tk.END)
```

```
    entry['state']='disabled'
```

```
    b9['state']='normal'
```

```
while True:
```

```
    global pulse_duration
```

```
    global pulse_start
```

```
    global pulse_end
```

```
    GPIO.output(TRIG, False)
```

```
    #print "Waiting For Sensor To Settle"
```

```
    time.sleep(0.5)
```

```
    GPIO.output(TRIG, True)
```

```
    time.sleep(0.00001)
```

```
    GPIO.output(TRIG, False)
```

```
while GPIO.input(ECHO)==0:
```

```
    pulse_start = time.time()
```

```

while GPIO.input(ECHO)==1:
    pulse_end = time.time()

pulse_duration = pulse_end - pulse_start

distance = pulse_duration * 17150

distance = round(distance, 2)

print ("Su Seviyesi: ",distance,"cm")

#return distance

entry['state']='normal'

entry.insert(0,distance)

entry.insert(tk.END," cm")

entry['state']='disabled'

break

#-----



#####
#####

def sicaklikDurumu():

    entry2['state']='normal'

    entry2.delete(0,tk.END)

```

```
entry2['state']='disabled'

b9['state']='normal'

while True:

    #print(type(read_temp()))

    print(str(read_temp())+" Derece")

    x = read_temp()

    time.sleep(1)

    entry2['state']='normal'

    entry2.insert(0,x)

    entry2.insert(tk.END," Derece")

    entry2['state']='disabled'

    break

#-----
```

```
#####
```

```
def isikKapat():

    while True:

        #GPIO.setup(14,GPIO.OUT) #Relay IN4 for Ligh

        print("-----")
```

```
print("IŞIK KAPANDI")
GPIO.output(14,GPIO.LOW)
time.sleep(1)

print("-----")
break

def isikAc():
    while True:
        #GPIO.setup(14,GPIO.OUT) #Relay IN4 for Light

        print("-----")
        print("IŞIK AÇILDI")

        GPIO.output(14,GPIO.HIGH)
        time.sleep(1)

        print("-----")
        break

#####
#####
```

```
def faniAc():

    while True:

        GPIO.setup(25,GPIO.OUT) #Relay IN4 for Ligh

        print("-----")
        print("FAN AÇILDI")

        GPIO.output(25,GPIO.LOW)
        time.sleep(1)

        print("-----")
        break

#####
#####

def faniKapat():

    while True:

        GPIO.setup(25,GPIO.OUT) #Relay IN4 for Ligh

        print("-----")
        print("FAN KAPATILDI")
```

```

GPIO.output(25,GPIO.HIGH)
time.sleep(1)

print("-----")
break

#####
#####

def pompa1Ac():

    temizle_cevap=messagebox.askyesno("Su Boşalt","Su Boşaltmak\nİstediğinizden
Eminmisiniz ?")

    if temizle_cevap==1:

        while True:

            #GPIO.setup(7,GPIO.OUT) #Relay IN4 for Ligh

            print("-----")
            print("POMPA 1 AÇILDI")

            GPIO.output(7,GPIO.LOW)
            time.sleep(1)

```

```
print("-----")
messagebox.showinfo("Su Boşalt","SU BOŞALTIYOR.")
break
else:
    pass
```

```
#####
#####
```

```
def pompa1Kapat():

    temizle_cevap=messagebox.askyesno("Su Boşalt","Pompa 1'yi Kapatmak\nİstediğinizden
    Eminmisiniz ?")

    if temizle_cevap==1:
```

```
        while True:
```

```
            #GPIO.setup(7,GPIO.OUT) #Relay IN4 for Ligh
```

```
            print("-----")
            print("POMPA 1 KAPATILDI")
```

```
            GPIO.output(7,GPIO.HIGH)
            time.sleep(1)
```

```
print("-----")
messagebox.showinfo("Su Boşalt","POMPA KAPATILDI.")
break
else:
    pass
```

```
#####
#####
```

```
def pompa2Ac():
```

```
    temizle_cevap=messagebox.askyesno("Su Doldur","Su Doldurmak\nİstediğinizden  
Eminmisiniz ?")
```

```
    if temizle_cevap==1:
```

```
        while True:
```

```
            GPIO.setup(15,GPIO.OUT) #Relay IN4 for Ligh
```

```
            print("-----")
```

```
            print("POMPA 2 AÇILDI")
```

```
            GPIO.output(15,GPIO.LOW)
```

```
            time.sleep(1)
```

```
print("-----")
messagebox.showinfo("Su Doldur","SU DOLDURULUYOR.")
break

else:
    pass

#####
#####

def pompa2Kapat():

    temizle_cevap=messagebox.askyesno("Su Doldur","Pompa 2'yi
    Kapatmak\nİstediğinizden Eminmisiniz ?")

    if temizle_cevap==1:
        while True:
            GPIO.setup(15,GPIO.OUT) #Relay IN4 for Light

            print("-----")
            print("POMPA 2 KAPATILDI")

            GPIO.output(15,GPIO.HIGH)
```

```
time.sleep(1)

print("-----")
messagebox.showinfo("Su Boşalt","POMPA 2 KAPATILDI.")

break

else:
    pass
```

```
#####
#####
```

```
#####
#####
```

```
def suSizintiDurumu():

    entry3['state']='normal'

    entry3.delete(0,tk.END)

    entry3['state']='disabled'

    b9['state']='normal'

    while True:

        state = GPIO.input(26)

        if state == GPIO.HIGH:

            print("SIZINTI VAR")

            entry3['state']='normal'

            entry3.insert(0,"Sızıntı Var")

            entry3['state']='disabled'

            messagebox.showinfo("Sızıntı","SIZINTI VAR")
```

```
break  
else:  
    print("SIZINTI YOK")  
    entry3['state']='normal'  
    entry3.insert(0,"Sızıntı Yok")  
    entry3['state']='disabled'  
    messagebox.showinfo("Sizinti","SIZINTI YOK")  
    break
```

```
time.sleep(1)  
print("-----")
```

```
#####
```

```
def uyari():  
    led_kirmizi.value =False  
    buzzer.duty_cycle=0
```

```
#####
```

```
def ent_temizle():
```

```
temizle_cevap=messagebox.askyesno("Temizleme","Bütün Verileri Temizlemek  
İstedığınızden\nEminmisiniz ?")  
  
if temizle_cevap==1:  
  
    entry['state']='normal'  
  
    entry.delete(0,tk.END)  
  
    entry['state']='disabled'  
  
  
    entry2['state']='normal'  
  
    entry2.delete(0,tk.END)  
  
    entry2['state']='disabled'  
  
  
    entry3['state']='normal'  
  
    entry3.delete(0,tk.END)  
  
    entry3['state']='disabled'  
  
    b9['state']='disabled'  
  
  
    """""  
  
    entry4['state']='normal'  
  
    entry4.delete(0,tk.END)  
  
    entry4['state']='disabled'  
  
    b16['state']='disabled'  
  
    """"  
  
  
  
#cv2.destroyAllWindows()  
  
else:
```

```
pass

#####
def veri_kaydet():

    y1=str(entry.get())
    y2=str(entry2.get())
    y3=str(entry3.get())

    an = datetime.datetime.now()
    st_an=datetime.datetime.strftime(an,"Tarih: %d.%m.%Y Saat %H.%M.%S")

    dosya_2 = open("akvaryum_bilgileri.txt", "a", encoding="utf-8")

    dosya_2.write("*****")
    dosya_2.write("\n")
    dosya_2.write(st_an)
    dosya_2.write("\n")
    dosya_2.write("Su Seviyesi: "+y1)
    dosya_2.write("\n")
    dosya_2.write("Sıcaklık Durumu: "+y2)
    dosya_2.write("\n")
    dosya_2.write("Sızıntı Durumu: "+y3)
    dosya_2.write("\n")
    dosya_2.write("*****")
    dosya_2.write("\n")
```

```
dosya_2.close()  
messagebox.showinfo("Kayıt","BİLGİLER KAYIT EDİLDİ")
```

```
#####
#####
```

```
pencere.title("Akvaryum Kontrol Sistemi -Anasayfa")  
pencere.geometry("700x550+0+0")
```

```
baslik=tk.Label(text="AKVARYUM KONTROL SİSTEMİ",font="Verdana 22  
bold",bg="white")  
baslik.place(x=80,y=10)
```

```
an = datetime.datetime.now()  
saat_yazi=tk.Label(text="Saat : "+(an.strftime("%H:%M:%S")),fg="white",font="Verdana  
15 bold",bg="grey")  
saat_yazi.place(x=10,y=60)
```

```
girdi_yazi=tk.Label(text="Lütfen Dakika Giriniz!",fg="white",font="Verdana 15  
bold",bg="red")  
girdi_yazi.place(x=200,y=60)
```

```
#kam_yazi2=tk.Label(text="ÇIKIŞ için Klavyeden 'E' Tuşuna Basınız",font="Verdana 10 bold",bg="white")
```

```
#kam_yazi2.place(x=165,y=130)
```

```
b2=tk.Button(text="Su Seviyesi",bg="blue",fg="#FFFFFF",font="bold",command = suSeviyesi)
```

```
b2.place(x=10,y=190)
```

```
b3=tk.Button(text="IŞIĞI KAPAT",  
,bg="red",fg="#FFFFFF",font="bold",command=isikKapat)
```

```
b3.place(x=10,y=300)
```

```
b5=tk.Button(text="IŞIĞI AÇ", bg="green",font="bold",fg="#FFFFFF",command=isikAc)
```

```
b5.place(x=10,y=255)
```

```
b7=tk.Button(text="Programı Kapat", command =sistemi_kapat)
```

```
b7.place(x=550,y=500)
```

```
b8=tk.Button(text="Temizle",command =ent_temizle)
```

```
b8.place(x=330,y=500)
```

```
b9=tk.Button(text="Sisteme Kaydet", state='disabled',command =veri_kaydet)
```

```
b9.place(x=415,y=500)
```

```
b10=tk.Button(text="Sıcaklık Durumu",  
bg="purple",font="bold",fg="#FFFFFF",command=sicaklikDurumu)
```

```
b10.place(x=10,y=150)
```

```
b11=tk.Button(text="FANI AÇ",  
bg="green",font="bold",fg="#FFFFFF",command=faniAc)
```

```
b11.place(x=150,y=255)
```

```
b12=tk.Button(text="FANI KAPAT",  
bg="red",font="bold",fg="#FFFFFF",command=faniKapat)
```

```
b12.place(x=150,y=300)
```

```
b11=tk.Button(text="SU BOŞALT",  
bg="green",font="bold",fg="#FFFFFF",command=pompa1Ac)
```

```
b11.place(x=300,y=255)
```

```
b12=tk.Button(text="1.POMPAYI KAPAT",  
bg="red",font="bold",fg="#FFFFFF",command=pompa1Kapat)
```

```
b12.place(x=300,y=300)
```

```
b13=tk.Button(text="SU DOLDUR",  
bg="green",font="bold",fg="#FFFFFF",command=pompa2Ac)
```

```
b13.place(x=490,y=255)
```

```
b14=tk.Button(text="2.POMPAYI KAPAT",  
bg="red",font="bold",fg="#FFFFFF",command=pompa2Kapat)
```

```
b14.place(x=490,y=300)
```

```
b15=tk.Button(text="Su Sızıntı Durumu",bg="blue",fg="#FFFFFF",font="bold",command = suSizintiDurumu)
```

```
b15.place(x=130,y=190)
```

```
b16=tk.Button(text="OTOMATİK MOD",  
state='disabled',bg="black",fg="#FFFFFF",font="bold",command = otoMod)
```

```
b16.place(x=10,y=110)
```

```
b17=tk.Button(text="RED",bg="red",fg="#FFFFFF",font="bold", command = kirmiziAc)
```

```
b17.place(x=300,y=150)
```

```
b18=tk.Button(text="GREEN",bg="green",fg="#FFFFFF",font="bold", command = yesilAc)
```

```
b18.place(x=360,y=150)
```

```
b19=tk.Button(text="BLUE",bg="blue",fg="#FFFFFF",font="bold", command = maviAc)
```

```
b19.place(x=440,y=150)
```

```
b20=tk.Button(text="RED",bg="black",fg="#FF0000",font="bold", command = kirmiziKapat)
```

```
b20.place(x=300,y=190)
```

```
b21=tk.Button(text="GREEN",bg="black",fg="#008000",font="bold", command = yesilKapat)
```

```
b21.place(x=360,y=190)
```

```
b22=tk.Button(text="BLUE",bg="black",fg="#0000FF",font="bold", command =  
maviKapat)  
  
b22.place(x=440,y=190)  
  
  
b23=tk.Button(text="GİR",bg="white",fg="#0000FF",font="bold", command = gir)  
  
b23.place(x=300,y=110)  
  
  
b24=tk.Button(text="YEM AT",bg="orange",fg="white",font="bold", command = yem_at)  
  
b24.place(x=520,y=190)  
  
  
b25 = tk.Button(text = "Kameraya Bağlan",bg="#D42866",fg="white",command =  
openweb)  
  
b25.place(x=520,y=110)  
  
  
b26 = tk.Button(text = "Ekran görüntüsü al",bg="#8128D4",fg="white",command =  
ekran_goruntusu)  
  
b26.place(x=520,y=150)  
  
  
su_seviyesi = tk.Label(text=("SU SEVİYESİ"),bg="white" ,font="Verdana 15 bold")  
su_seviyesi.place(x=10, y=370)  
  
  
sicaklik_durumu = tk.Label(text='SICAKLIK DURUMU'), bg="white",font="Verdana 15  
bold")  
  
sicaklik_durumu.place(x=10, y=420)
```

```
sizinti_durumu = tk.Label(text='SIZINTI DURUMU', bg="white",font="Verdana 15 bold")
sizinti_durumu.place(x=10, y=460)

pencere.mainloop()
```