# AKILLI YOKLAMA SİSTEMİ- AYS

### Github Adresi

https://github.com/CiscoPython10/Grup10Proje

### Grubu oluşturan öğretmenlerin listesi

Ahmet ERYILMAZ, Çağlar AKÇAY, Ekrem TÜRKAY, Gökhan TOR, Kemal Erdem ÖZÇELİK, Mehmet KAVAKLI, Serkan ÇAM, Volkan ÇINAR

# Proje ismi Analiz Raporu

### Kısa Özet

### Problem Tanımı

Oluşturulan yazılım ile yüz tanıma sistemi kullanılarak giriş çıkışların kayıt altına alınması ve özellikle okullarda yoklama için harcanan zamanın geri kazanılması hedeflenmektedir.

### Analiz Süreci

#### İhtiyaç Analizi

Okullarda öğretmenler derse girdiklerinde yoklama almak ve bunu kayıt altına almak zorundadırlar. Ayrıca gün sonunda yoklama fişini idareye ulaştırmak zorundalar. Günlük 10 dersin işlendiği bir lise düşünüldüğünde ortalama 10 dakika süreden günlük 100 dakika verimsiz kullanılmış olmaktadır. BU kaybedilen zamanı geri kazanmak için bu proje geliştirilmiştir.

#### İçerik Analizi

Projemiz öğrencilerin sınıfa giriş ve çıkışlarını otomatik takip etmektedir. Projemiz kurulum gerektirmemesi açısından web tabanlı olarak düşünülmüştür. Kurumda kullanılacak bir server üzerinden sınıflardan ulaşılabilir ve kontrolü de tek bir noktadan yapılabilir.

Ayrıca oluşabilecek arıza durumlarında alternatif olması açısından kartlı sistem entegre edilmiş ve arıza giderilene kadar yoklama işleminin öğrencilerde bulunan kartlar ile yapılması sağlanmıştır

#### Durum Ortam Analizi

Yazılımın geliştirilme sürecinde kullanılan kütüphaneler aşağıda listelenmiştir.

* Arduino
* RFID Kartlar
* Flask, Flask\_SqlAlchemy Kütüphanesi
* Sqlite3 Kütüphanesi
* Wtforms Kütüphanesi
* Serial Kütüphanesi
* DateTime Kütüphanesi
* OpenCV Kütüphanesi
* OS Kütüphanesi
* Numpy Kütüphanesi
* Pil Kütüphanesi
* Functools Kütüphanesi

#### Kullanıcı Analizi

Bu yazılım her ne kadar devlet veya özel eğitim kurumların yoklama için harcadıkları zamanın geri kazanımı için hazırlanmış olsa da günlük devam durumu, çalışma saati, mesai hesaplama vb hizmetleri kullanan işletmeler, fabrikalar ve kurumlar tarafından da kullanılabilir.

# Proje ismi Tasarım Raporu

### Kısa Özet

Proje geliştirme süreci, kullanılacak yapıların tespiti ve amacın net belirlenmesi ile başlamıştır. İlk önce web arayüz için kullanılacak iş çatısı olarak Flask seçilmiştir. Veritabanı yapısı olarak SqLite seçilmiştir. Burada flask ve sqlite ikilisi prototip olarak hazırlanacak proje için yeterince esnek ve hafif bir çalışma ortamı sunmaları sebebi ile seçilmiştir.

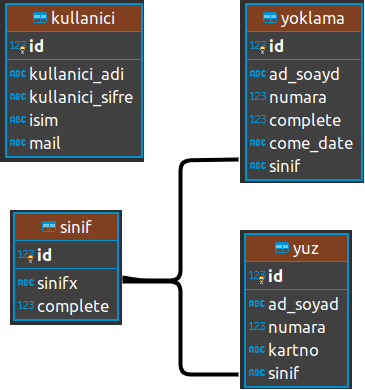
Hazırlanacak proje daha çok prototip amaçlı olup temel düzeyde işlevselliğe sahip olacak şekilde planlanmıştır. Sisteme yüz tanımaya ek olarak RFID kartlar ile yoklama sistemi de eklenmiştir.

Bu doğrultuda ilk önce veritabanı tasarımı gerçekleştirilmiştir. Veritabanı sadece amaca yönelik yeterli sayıda veri tutacak şekilde tasarlanmıştır.

Flask ile basit bir arayüz yapısı oluşturularak; yoklama sistemine yardımcı olacak sunucu tarafındaki yüz tanıma ve kart okuma sistemleri oluşturulmuştur. Burada her işlem(kayıt, eğitme ve tespit etme, yoklama) ile ilgili sayfalar ve kod arkası çalışmaları yapılmıştır.

Son olarak sistemde kartla yoklama işlemini gerçekleştirecek arduino tabanlı devreler tasarlanıp istenilen işlevsellikler eklenmiştir.

## Veri Tasarımı



## Ara yüz Tasarımı

* Ana sayfaekran görüntüsü içeren bir resim

  Açıklama otomatik olarak oluşturuldu
* Kayıt

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

* Giriş(login)ekran görüntüsü içeren bir resim

  Açıklama otomatik olarak oluşturuldu
* Kullanıcı Bilgileri

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

* Sınıf Ekleme

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

* Yoklama

ekran görüntüsü, ışık, trafik içeren bir resim

Açıklama otomatik olarak oluşturuldu

* Kontrol Paneli

ekran görüntüsü içeren bir resim

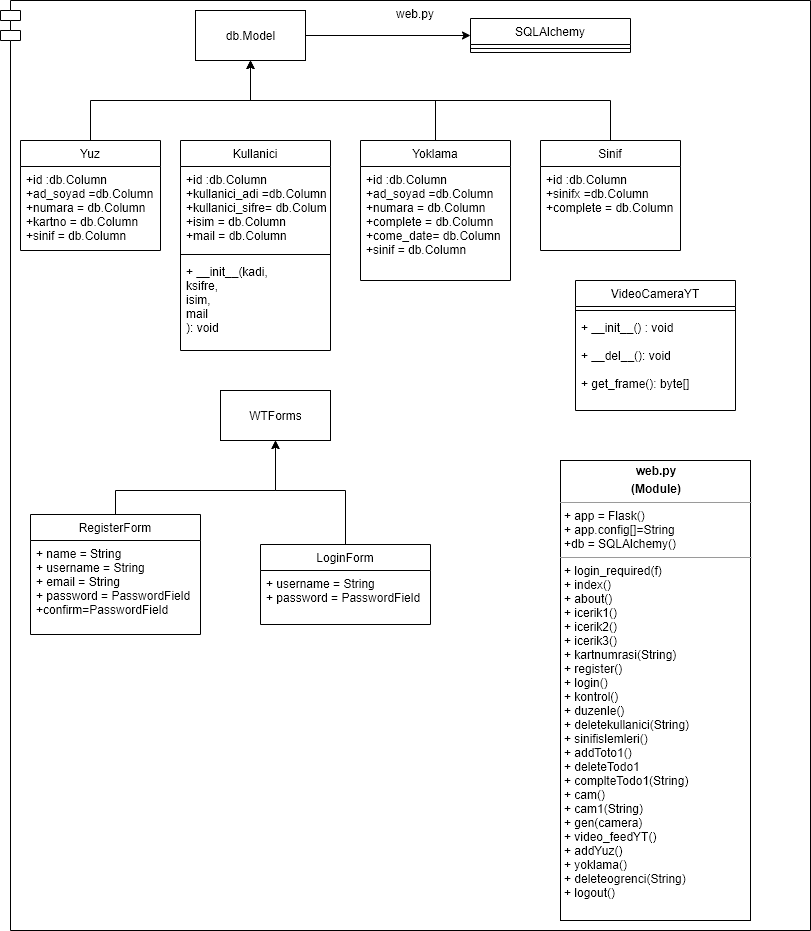
Açıklama otomatik olarak oluşturuldu

* Tanıma işlemleri

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

## Kod Tasarımı



ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

## Zaman Çizelgesi

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| İş Tanımı/Zaman | Temmuz | | | | | Ağustos | | | |
| 1. Hafta | 2. Hafta | 3. Hafta | 4. Hafta | 5. Hafta | 1. Hafta | 2. Hafta | 3. Hafta | 4. Hafta |
| Proje Planının Yapılması |  |  |  |  |  |  |  |  |  |
| Ekipte Görev Dağılımlarının Yapılması |  |  |  |  |  |  |  |  |  |
| Veritabanı Hazırlanması |  |  |  |  |  |  |  |  |  |
| Arayüz Hazırlanması |  |  |  |  |  |  |  |  |  |
| Yüz Tanıma Kod Arkasının Hazırlanması |  |  |  |  |  |  |  |  |  |
| Arduino Yapısının Kurulması |  |  |  |  |  |  |  |  |  |
| Hazırlanan Sistemin Çalışmasının Denenmesi |  |  |  |  |  |  |  |  |  |
| Proje Raporunun Yazılması |  |  |  |  |  |  |  |  |  |

# Akıllı Yoklama Sistemi Gerçekleştirme Raporu - MEHMET

## Karşılaşılan Sorunlar ve Uygulanan Çözümler

Projemizi hazırlamaya başlayacağımız dönemde ilk olarak tasarım hakkında genel bir plan hazırladık. Bu kısımla alakalı çok fazla bir sorun yaşamadık. Daha sonra ise tasarım bittikten sonra kodlama kısmında webcam ile alınan görüntülerde aynı frame de açma sorunu yaşadık. Ancak bu sorunu yapmış olduğumuz araştırmalarla hatalarımızı görerek uygun kısımlara yeni kodlara ekleyerek sorunları çözmeyi başardık.

Temele indiğimizde sorun olarak belki de kodlamalarda yaşadığımız ufak tefek küçük hataları söyleyebiliriz.  Bunlar opencv özelliklere tam hakim olmadığımızdan dolayı gözden kaçan basit hatalardır. Grup olarak geliştirdiğimiz bu projede tüm arkadaşlarımızın dikkatli kontrolleri sonucu karşılaştığımız o basit hataları kolayca halletmiş olduk.

## Proje Bileşenleri ve Görevleri

# AKILLI YOKLAMA SİSTEMİ - AYS

# KULLANICI ARAYÜZÜ

# Yoklama ve Sınıf Listesi Modülleri

Öğrencilerin okulda olduklarına ait bilgiler ile sınıf listelerinin alındığı modül

# Sınıf Modülü

Okulda var olan sınıfların eklenip aktif veya pasif olmasını sağlayan modül

# Yüz Tanımlama Modülü

Sınıflarda bulunan öğrencilerin bilgileri (ad, no, ıdkart, sınıf) ve web cam aracılığı ile çekilen resimlerin veritanabına kayıt edilmesi

# Kullanıcı Kayıt Paneli

Sisteme dahil olacak kullanıcıların kaydedilmesi

# Kontrol Paneli

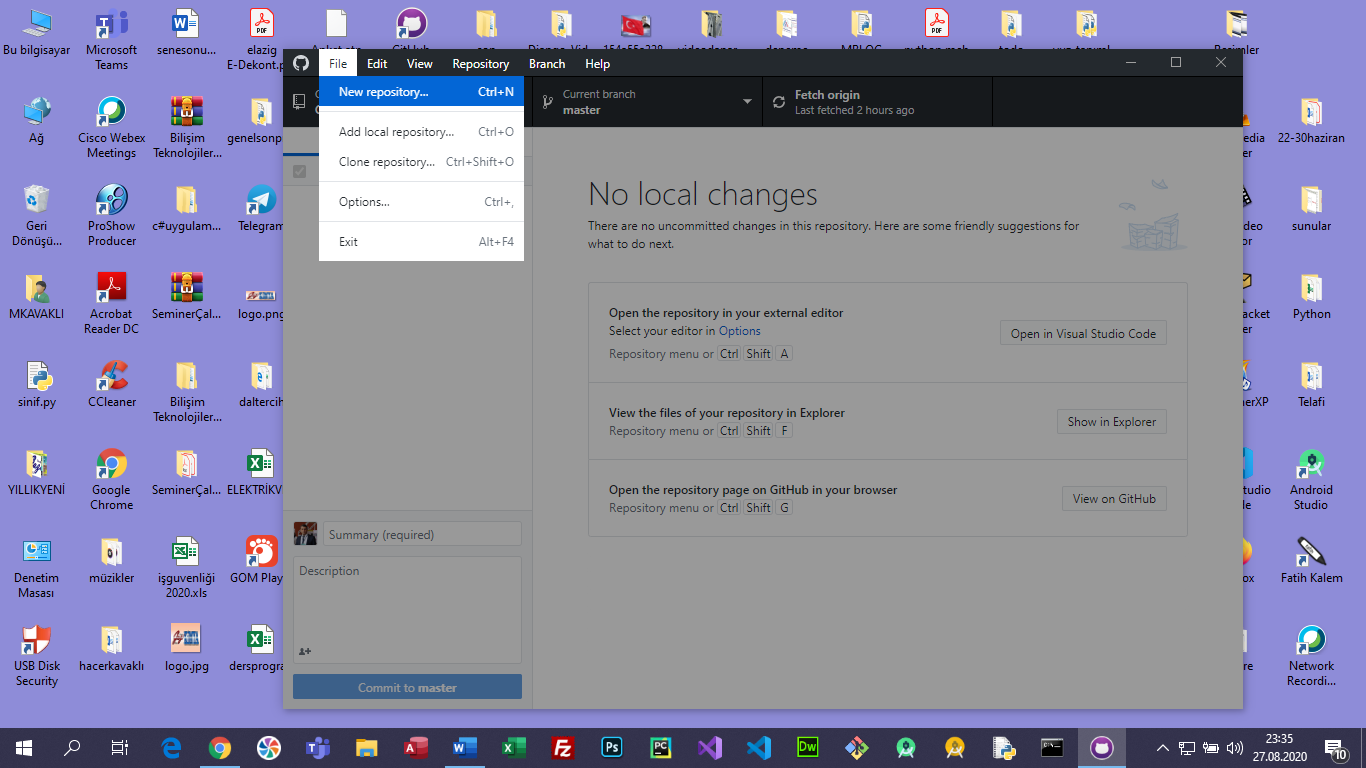
Sisteme Dahil Olan Kullanıcıların Yapacakları İş ve İşlemler Modülü

## Github Yükleme Süreci

GitHub’ın masaüstü uygulaması Windows ve Mac için mevcut. En rahat GitHub kullanımı konsol üzerindendir.

**GitHub for Windows**

[GitHub](https://github.com/) anasayfasına gidip ve “Sign Up” diyerek hesap oluşturulur. İşletim sisteminize göre “[GitHub for Mac](https://mac.github.com/" \t "_blank)” ve “[GitHub for Windows](https://windows.github.com/" \t "_blank)” indirme düğmelerine basarak .zip dosyasını indirebilirsiniz. Mac ve Windows için klasik yükleme adımlarını yaparak yüklemeyi tamamlayabilirsiniz.

Öncelikle işe bir depo (repository) oluşturarak başlanır. Oluşturulan depolarla farklı projeler yönetilir. Bir projenin genellikle bir deposu olur. Altında da her bir uygulamanın ayrı klasörleri olur. Bu yapının nasıl olacağı tamamen kullanıcıya bağlıdır.

**GitHub’da Depo Oluşturmak**

GitHub’da birçok şeyi hem masaüstü uygulamasından hem de web sitesi üzerinden yapılabilir.

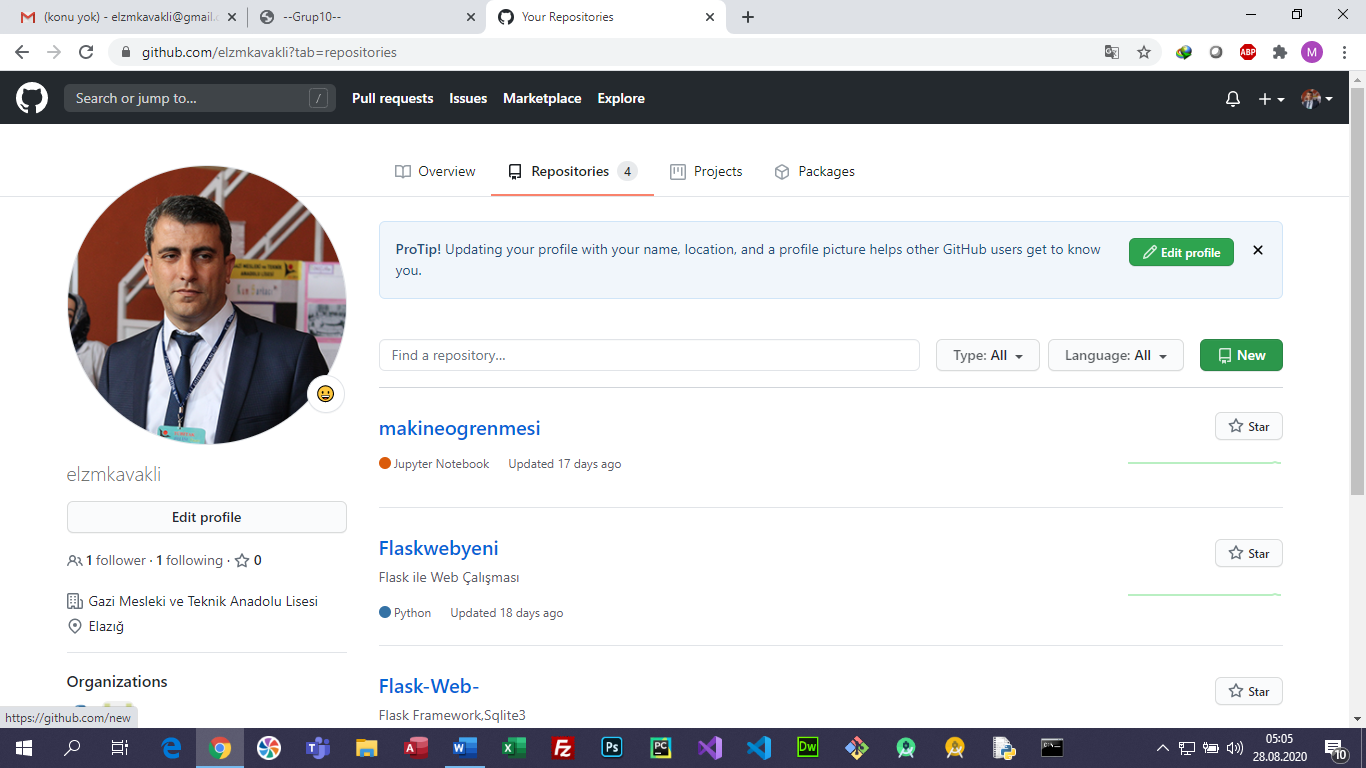
**GitHub’da Roller**

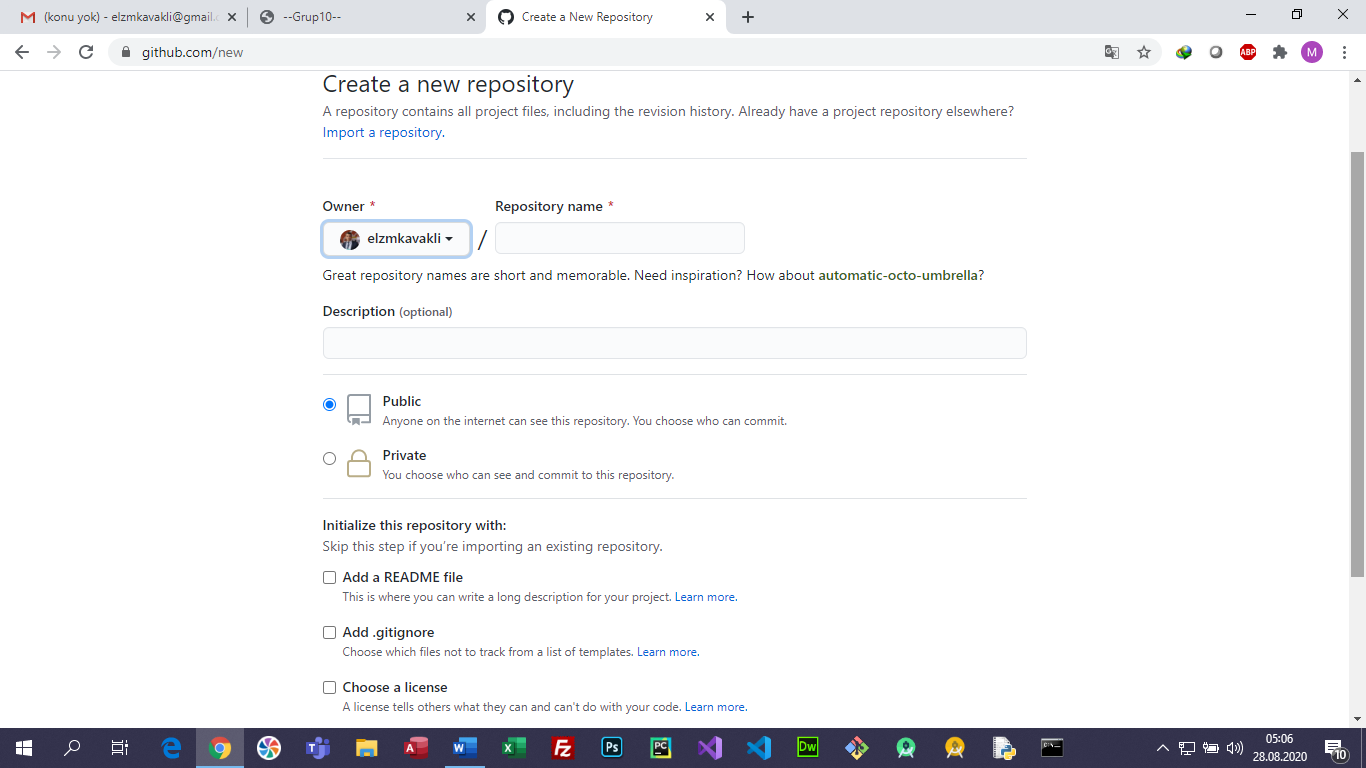
GitHub’da genellikle, her sistemde olduğu gibi, Admin (Yönetici) ve Contributer (Katılımcılar) vardır. Bir deponun bir yöneticisi olması, özellikle büyük bir proje değilse, yerinde olacaktır. Katılımcı sayısı ise yüzlerce, binlerce olabilir.

Deponun yöneticisi; depoyu oluşturan, kuralları belirleyen, tüm değişiklikleri gözlemleyen ve son karar verendir. Katılımcılar ise; tüm kod üzerine, aksi yönetici tarafından belirlenmedikçe, katkıda bulunabilirler.

Eğer depo oluşturacaksanız, en basit yol web tarafında oluşturmaktır.

Github.com’a gidilir ve hesap ile giriş yapılır. Tek yapmanız gereken sağ üstte bulunan artı (+) düğmesine basmak ve ardından “New repository” bağlantısına tıklamak.



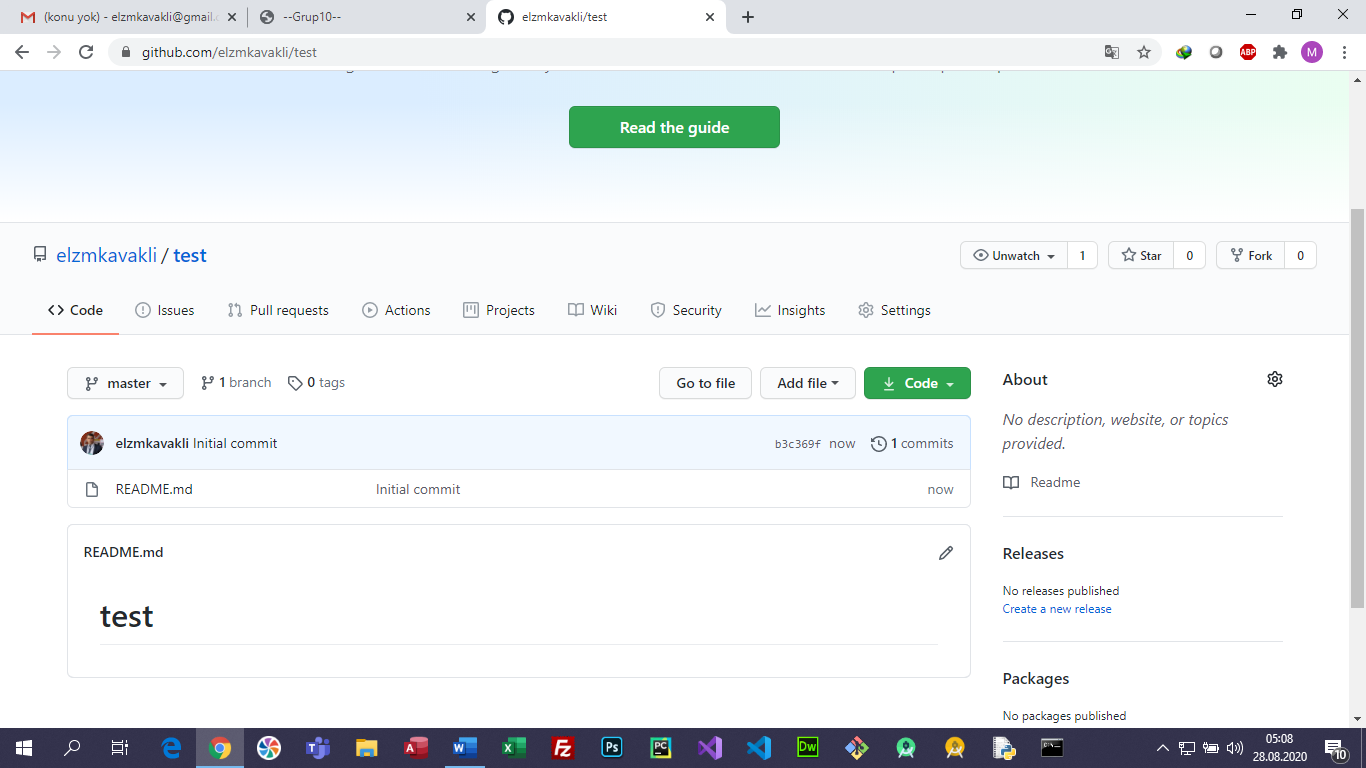
Ardından açılan ekranda, deponuzun adını giriniz ve mümkünse bir de açıklamasını altına not düşünüz. Eğer kodlarınızı açık kaynak tutacaksanız “Public” seçeneğini, gizli kalmasını istiyorsanız da “Private” seçeneğini işaretleyiniz.

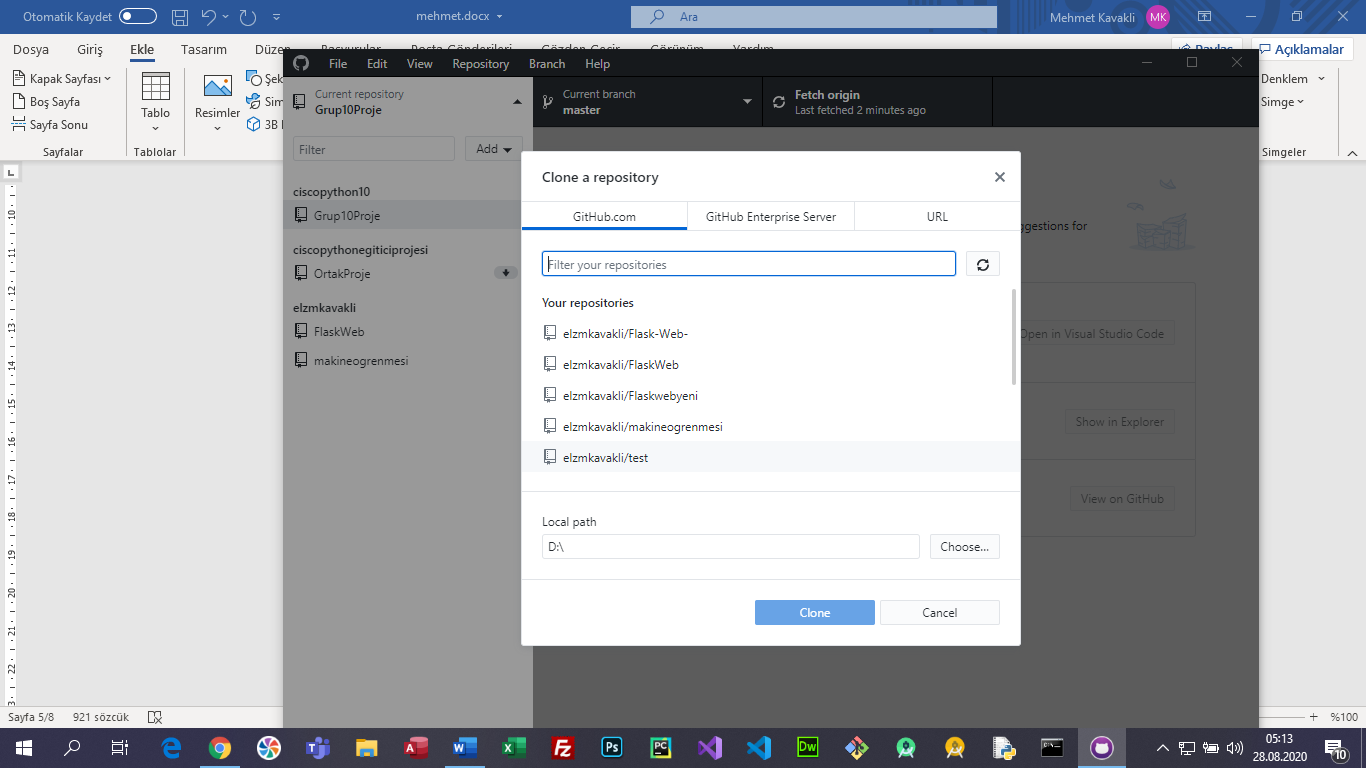
Açık kaynak olarak sınırsız sayıda depo yaratabilir ve istediğiniz kadar uygulama yayımlayabilirsiniz. Gizli olaraksa sadece 1 tane hakkınız vardır.

Burada en önemli noktalardan biri, GitHub her deponun ve uygulamanın README dosyasının olmasını çok istiyor. Bu yüzden “initialize this repository with a readme” seçeneğini kesinlikle işaretleyin, sonra içeriğini dolduracağız.

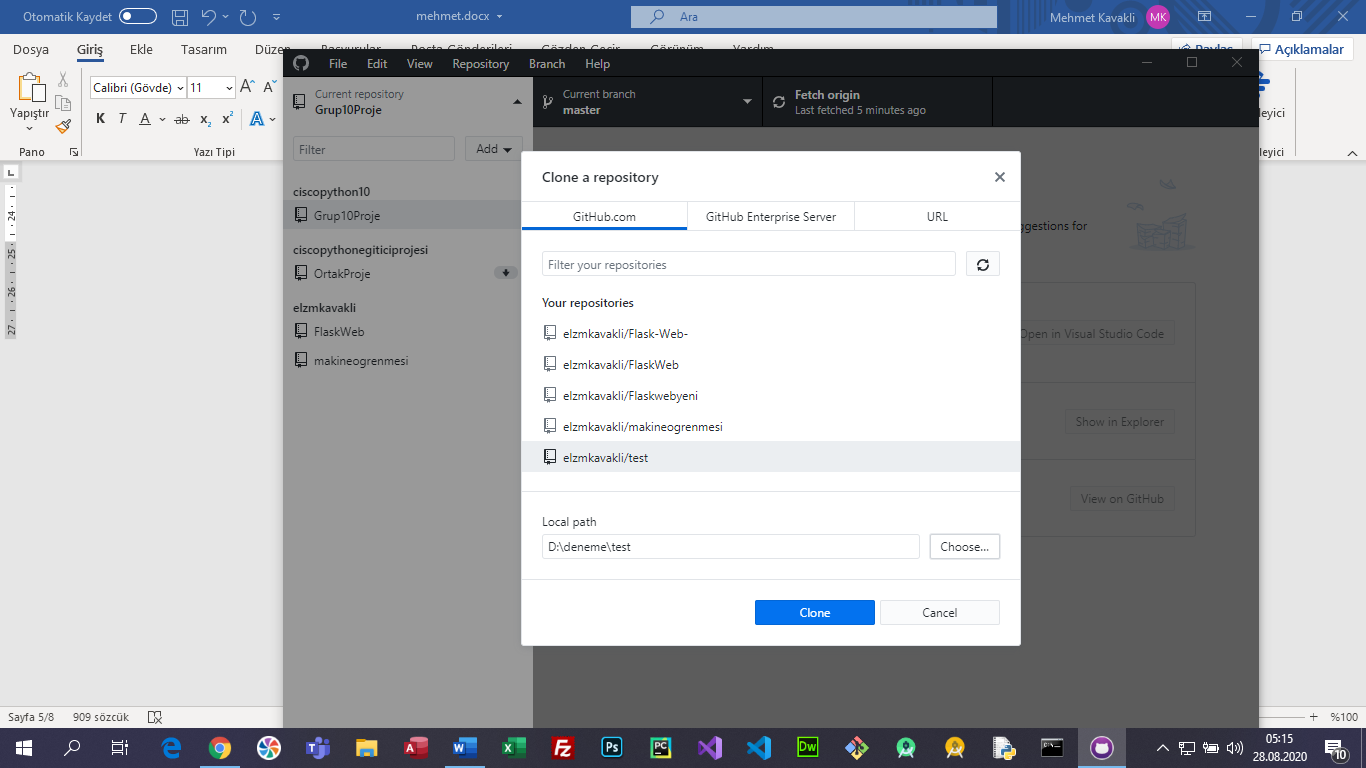
Onların hemen altında yer alan seçeneklerde ise, neyle ilgili kodlama yaptığınızı (genellikle kodunuzu yüklediğinizde GitHub zaten otomatik olarak algılayacaktır) ve varsa yayımlamak istediğiniz lisans modelini seçebilirsiniz.

Sonra karşınıza boş bir depo gelecektir. Ben test adlı bir depo oluşturdum örnek olarak.

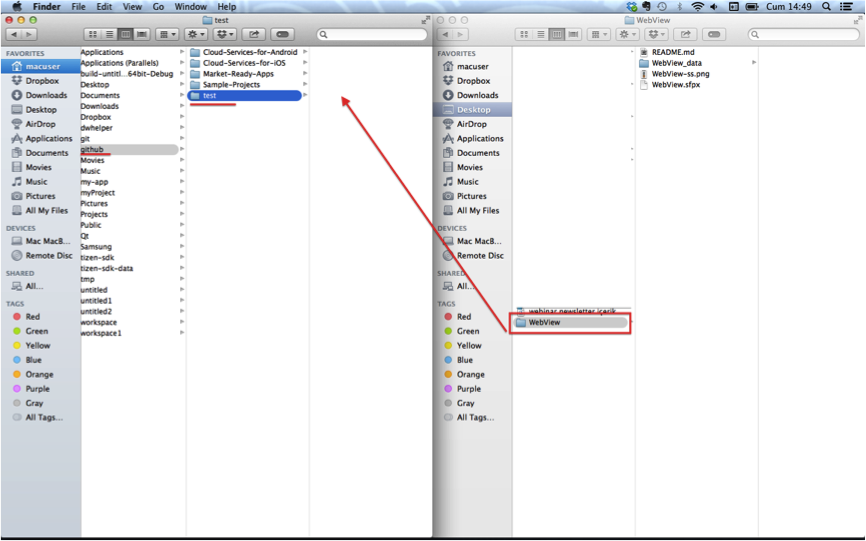
  
GitHub for Windows’dan:

Uygulamanın sol üst köşesinde yer Current repository sekmesindeki (v) düğmesine bastığımızda bize “Add” sekmesiyle yeni bir depo oluşturmayı ya da “Clone” sekmesiyle mevcut bir depoyu klonlamamızı teklif edecek. Eğer bir depoya katkıda bulunacaksak ya da yönetici olarak ilk kez masaüstü uygulamasından yöneteceksek, o depoyu bir kez klonlamamız gerekmektedir. “Clone” sekmesine bastığımızda, daha önce oluşturduğumuz “test” adlı depomuzu göreceğiz. Onu seçerek altta yer alan “Clone Repository” düğmesine tıklanır.

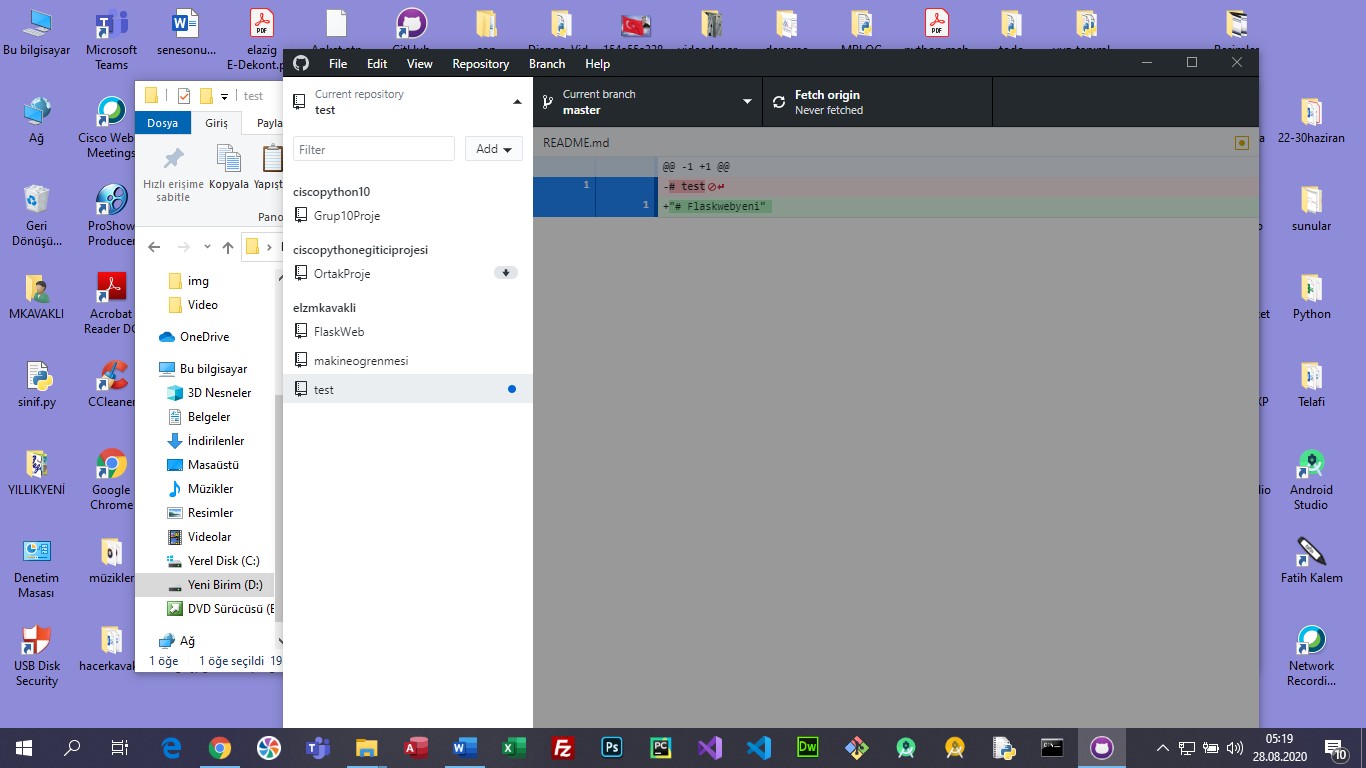
Windows Explorer içerisinde nereye klonlamamızı istediğimizi soracak ve yer belirtildikten sonra “Clone” düğmesine basarak ilgili depoyu bilgisayarınıza ve tabii ki aynı anda masaüstü GitHub uygulamanıza klonlamış olursunuz.



Kontrol etmeniz gereken en önemli şey; Finder’ı veya Windows Explorer’ı açıp ve oluşturduğunuz GitHub klasörünün içine bakınız. “test” adlı bir klasör varsa, işlem tamamdır. İçerisinde readme dosyası olması normaldir, web tarafında oluştururken bu seçeneği işaretlemiştik. Bu şekilde işlem tamamlanmış olur.

[](https://oytuneren.net/wp-content/uploads/2014/09/copygithubfile1.png)Bundan sonra işler aşırı kolay devam edecektir. Elinizde kaynak kodları hazır bir uygulamanız varsa, bilgisayarınızda nerede olursa olsun, tek yapmanız gereken ilgili klasörü buraya kopyalamak. Örneğin, masaüstündeki Webview uygulamamı sürükle bırakla GitHub klasörünün altında “test” klasörüne kopyalıyorum:

Herhangi bir dosyayı ya da klasörü ilgili depoya kopyaladığınız anda, GitHub uygulamasında bu değişiklikler belirecektir:



İlk kez bir dosya attığımız için hepsi “New” şeklinde gösteriliyor. Bu aşamadan sonra tek yapmanız gereken yaptığınız değişikliğin “Summary” özetini yazmanız ve “Commit & Sync” etmenizdir.

Eğer yaptığınız değişikliğin aynı zamanda senkronize etmesini, ilgili depoda, istemiyorsanız, sadece “Commit” seçeneğiyle gönderiniz.

Üstteki sekmelerden “History” adı üstünde ilgili dosyalarda yapılmış değişiklikleri, kimlerin hangi saatlerde ve hangi satırlara yapmış olduğunu gösterir ve deponun yöneticisi tarafından da herhangi bir hata da geriye alınabilir.

Uygulamanın sağ üst köşesinde yer alan sync düğmesi, adı üstünde deponuzdaki son gelişmeleri senkronize eder.

Yönetici olarak bundan sonra yapmanız gereken tek şey, varsa uygulamanızı an be an güncel tutmak ve sürekli GitHub’a senkronize etmek. İsterseniz, artık kaynak kodlarınızı kopyaladığınız GitHub klasörünün altında da çalışabilirsiniz. GitHub uygulaması tüm değişiklikleri bilgisayarınızdaki bu klasörün içinden okur ve uygulamasını da size sunar. Projenizi ilgili klasör altında çalıştırdıktan sonra kaydeder (Save) ve sonrasında GitHub uygulamasını açarak “Commit” edebilirsiniz. Böylece depoda güncellenir ve herkes bu güncellemeyi anında görebilirler.

# Proje ismi Test Raporu

## Karşılaşılan Sorunlar ve Uygulanan Çözümler

**Web alt yapısı olarak Flask ya da Django arasında seçim yapmak**:08/08/2020 Tarihli 627584627 numaralı “Webex” toplantısı ile aramızdaki koordinasyonun nasıl olacağını konuştuğumuz toplantıda web altyapısı olarak kullanacağımız mimariye karar veririken avantaj-dezavantaj durumunu değerlendirmek için” Django Vs Flask| Django mu yoksa Flask mı” (<https://www.mertmekatronik.com/django-vs-flask-django-mu-yoksa-flask-mi#:~:text=Django%20t%C3%BCm%20%C3%B6%C4%9Feleri%20i%C3%A7eren%20bir,nazaran%20daha%20mikro%20bir%20frameworkt%C3%BCr.&text=E%C4%9Fer%20web%20projeniz%20SQLite%2C%20PostreSQL,daha%20mant%C4%B1kl%C4%B1%20bir%20se%C3%A7im%20olacakt%C4%B1r>) makalesinden yararlandık. Flask django kadar karmaşık olmadığından. Bizde büyük web siteleri oturum yönetimleri değil de genelde öğrenciler ile küçük projeler yapacağımız için flask daha ideal bir çözüm olur karına vardık.

**Flask yapısının projeye entegre edilmesi için ekibin bilgilendirilmesi:”**Flask Mini Serisi “ isimli video eğitim serisini izleyerek ekibin konu hakkında bilgi edinmesini sağladık. <https://www.youtube.com/playlist?list=PLIHume2cwmHfDUQ2t6f-Q7VCBvNSEAMAS>

**Frame sayfa üzerinde tanımlamayı açmamak ve tanımlama işleminde çıkarken hafızada resim bulunmadığından dolayı alınan hata:** Kamerayı sayfaya ekleyerek problem çözüldü

**Hep ayrı Frame açma sorunu:** Hem Linux hem de Windows tarafında ayrı çözüm yöntemi uygulanması gerektiğini fark ettik. Bulduğumuz çözüm:

cv2.VideoCapture(0) #win

cv2.VideoCapture(-1) #linux

If you play local video:

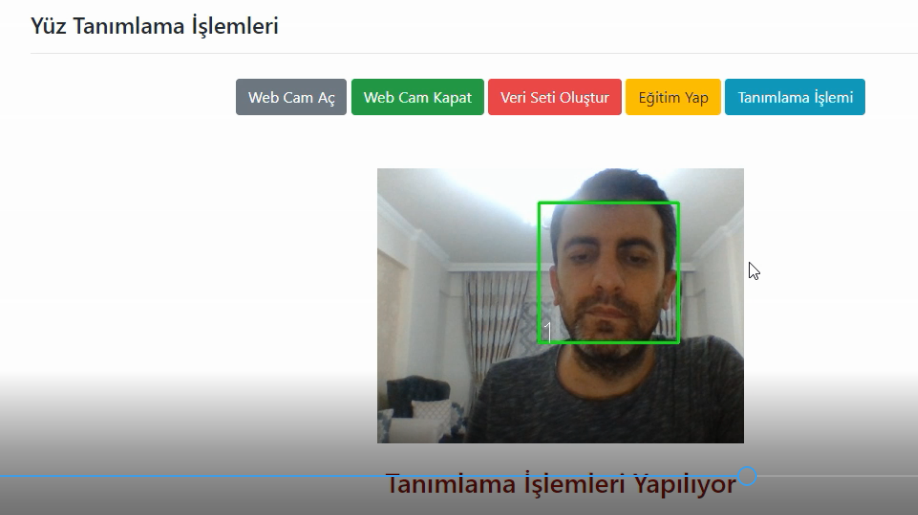
cv2.VideoCapture("videoFilePath")

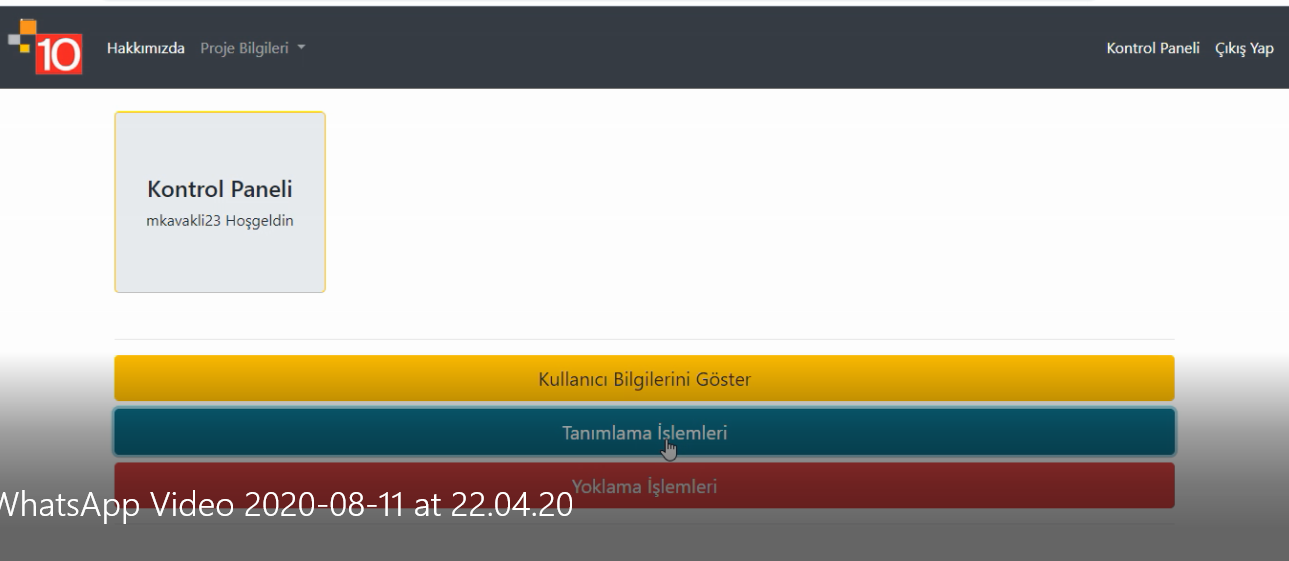
Şeklinde kodu revize etmek oldu.

**esp8266 Sorunu:** Arduino wemos d1r1 modülü kullanarak sorunu aştık.

**Test Süre cinde Kullanılan Modüller (Varsa)**

Projeyi gerçek ortamda test ettik ve problemsiz çalıştığını gördük.





tablo, iç mekan, oturma, ahşap içeren bir resim

Açıklama otomatik olarak oluşturuldu

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

iç mekan, oturma, tablo, ön içeren bir resim

Açıklama otomatik olarak oluşturuldu

## Değerlendirme Kriterleri

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **KRİTERLER** | Ekip Üyeleri | Eğitmen 1 | Eğitmen 2 | Eğitmen 3 | Ortalama |
| 1. Analiz Raporunun Tamamlanması |  |  |  |  |  |
| 1. Tasarım Raporunun Tamamlanması |  |  |  |  |  |
| 1. Gerçekleştirim Raporunun Tamamlanması |  |  |  |  |  |
| 1. Gantt Diagramı |  |  |  |  |  |
| 1. Arayüz tasarımı |  |  |  |  |  |
| 1. Veri Tasarımı-Sınıf Tasarımı |  |  |  |  |  |
| 1. Kullanıcı Yardım Dökümanı |  |  |  |  |  |
| 1. Programın Çalıştırılması |  |  |  |  |  |
| 1. Yazılım Test Çalışması |  |  |  |  |  |