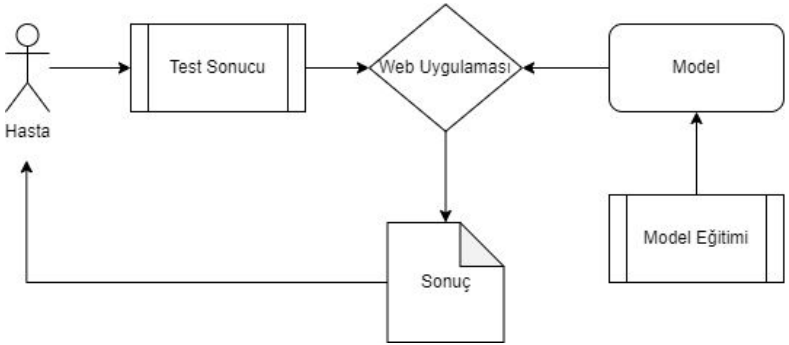
Diyabet Hastalığı Tespit Sistemi

Tasarım Sunumu

Hakam CHEDO	152120181096
Hüseyin KAYMAK	152120181100
Bengisu ŞAHİN	152120191064
Muhammet Eren SÖME	152120201049

Sistem Tasarımları



- The N. Inst. of Diabetes & Diges. & Kidney Dis'"den diyabet veri kümesi kullanıyoruz.
- sklearn python kütüphanesi, diyabet veri setini taramak için kullanılır.
- steamlit python kütüphanesi, diyabet tahmini web uygulama sistemi tasarlamak için kullanılır.

Kullanıcı ve Sistem Arayüzü Tasarımları

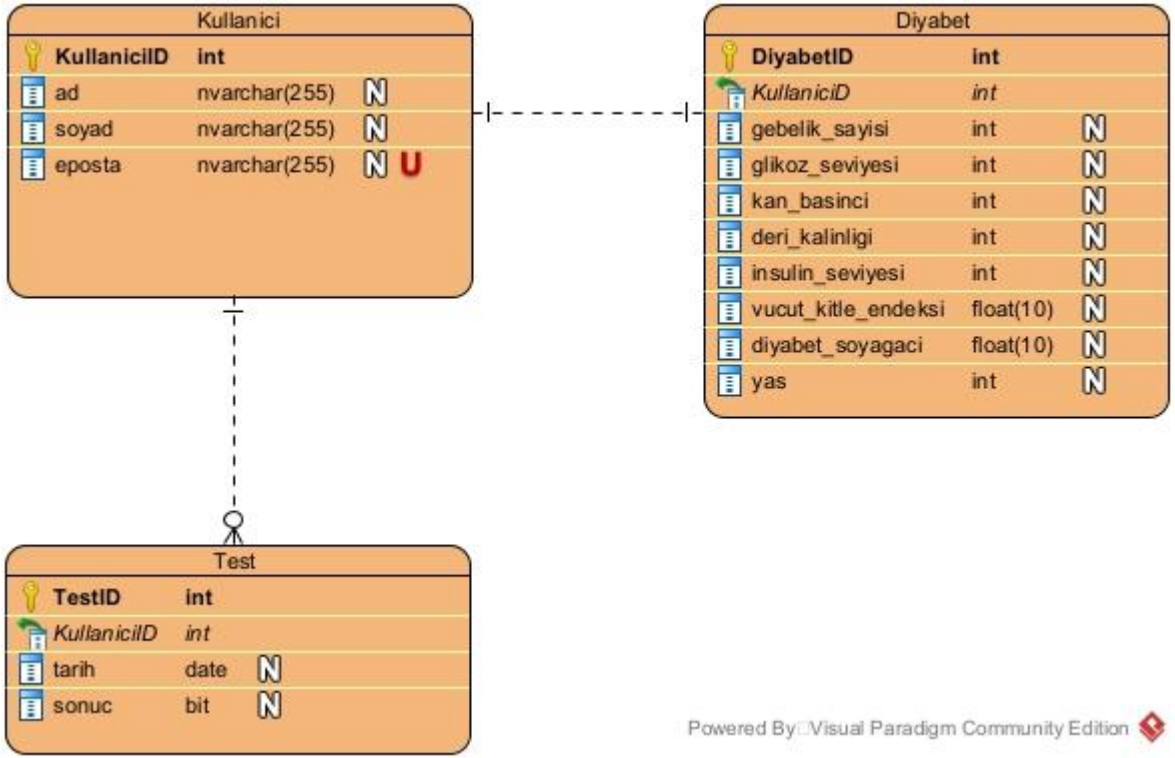
Kullanıcı dostu olması için tek sayfalık bir web sistemi oluşturduk. Kullanıcılar, kullanıcı verilerini girebilecek ve test sonuçlarını hemen görüntüleyebilecek.

- Kullanıcı kullanıcı adını soyadını ve diyabet hastalığı bilgilerini veri giriş textboxlara girebilir
- Kullanıcı test result butonuna basıp sonucu görebilir.

Veri Tabanı Tasarımı

- Uygulamada tek tip kullanıcı vardır ve bu da hastaları temsil etmektedir.
- Her kullanıcıdan diyabet testi için gerekli olan hastalık bilgileri istenmektedir.
- Kullanıcıların testleri ayrı bir tabloda sonucuyla birlikte tutulur.
- Bir kullanıcı birden fazla defa test yaptırabilir.
- Kullanıcı sistemden silindiği zaman yaptırdığı testlere ve diyabet bilgilerine erişilemez.

ER Diyagramı



Stored Procedures (Saklı Yordamlar)

BelirtilenSonucTestGetir

```
CREATE PROCEDURE BelirtilenSonucTestGetir @test_sonucu bit
AS
begin
SELECT  T.TestID as 'Test ID', K.Ad + ' ' + K.Soyad as 'Ad-Soyad', D.Yas as 'Yaş', T.Tarih as 'Test Tarihi', T.sonuc as 'Test Sonucu'
FROM Test T, Kullanici K, Diyabet D
WHERE T.Sonuc = @test_sonucu AND T.KullaniciID = K.KullaniciID AND D.KullaniciID = K.KullaniciID
ORDER BY 'Ad-Soyad', T.Tarih
end

EXEC BelirtilenSonucTestGetir @test_sonucu = 1;
```

Çıktı:

	Test ID	Ad-Soyad	Yaş	Test Tarihi	Test Sonucu
1	1	Eren Söme	20	2012-08-13	1
2	12	Eren Söme	20	2019-01-17	1
3	13	Eren Söme	20	2020-02-15	1

Triggers (Tetikleyiciler)

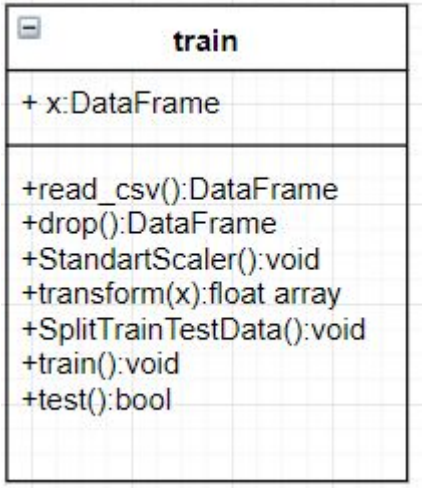
Test_Tarih tetikleyicimiz, test satırına yeni kayıt eklenirken tarih sütunu boş bırakıldıysa tarih sütununu güncel zamana eşitler.

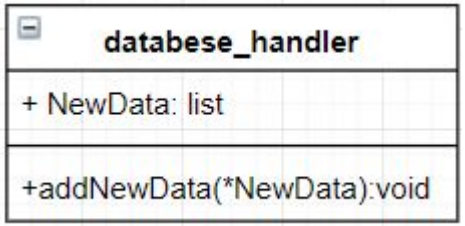
```
CREATE TRIGGER Test_Tarih
ON dbo.Test
AFTER INSERT
AS
begin
    UPDATE dbo.Test
    SET Tarih = GETDATE()
    WHERE Tarih IS NULL;
end
```

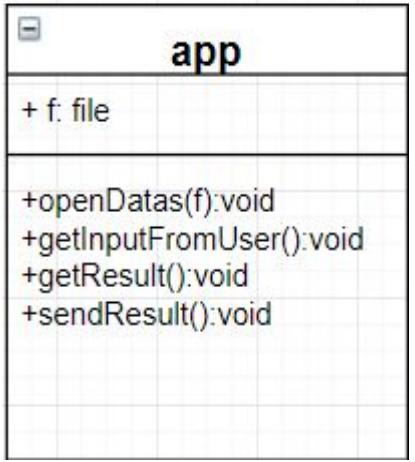
Gereksinime Bağlı Tasarım Kalıpları Seçimi

Geliştirilecek uygulama için herhangi bir tasarım kalıbı kullanılmasına ihtiyaç duyulmamaktadır.

UML Kullanarak Tasarım Diyagramları Oluşturma







Test Tasarımı

Yazılım testi, belirli gereksinimleri karşılayıp karşılamadığını belirlemek ve herhangi bir kusuru belirlemek için bir yazılım uygulamasını veya sistemini değerlendirme sürecidir. Etkili yazılım testi, yazılım uygulamalarının kalitesini ve güvenilirliğini sağlamak için gereklidir. Zamandan ve kaynaklardan tasarruf sağlayabilen geliştirme sürecinin erken aşamalarında kusurları tespit edip düzeltmeye yardımcı olur.

Aşağıdakiler de dahil olmak üzere farklı yazılım testi türleri vardır:

Unit Testing (Birim testi)

Integration Testing (Entegrasyon testi)

System Testing (Sistem testi)

Acceptance Testing (Kabul testi)

Gereksinim Analizlerinden Test Hedeflerinin Belirlenmesi

- Modelin yeni veri eklenmesinden sonra yeniden eğitime tabi tutulur.
- Doğruluk değerinin %78 ve üzerinde olması hedeflenmektedir.
- Doğruluk oranının düşmesi durumunda veri setini kontrol etme ve sinir ağı yapısında değişiklik yapma gibi işlemler gerçekleştirilecektir.

Birim Testleri Tasarımı

Birim testi, bir sistemde mantıksal olarak izole edilebilecek en küçük kod parçası olan bir birimi test etmenin bir yoludur. Projede birim testin uygulanacağı kısım makine öğrenme modelinin verilen değerlerden hastalık sonucunun tespitidir.

```
import unittest
class TestResult(unittest.TestCase):
    def test(self):
        self.assertTrue(is_sick(6,148,72,35,0,33.6,0.627,50))


def is_sick(Pregnancies, Glucose, BloodPressure,  SkinThickness,  Insulin,  BMI,  DiabetesPedigreeFunction, Age):
  input_data = [Pregnancies, Glucose, BloodPressure,  SkinThickness,  Insulin,  BMI,  DiabetesPedigreeFunction, Age]
  input_data_as_numpy_array = np.asarray(input_data)

  # reshape the array as we are predicting for one instance
  input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
  print(input_data_reshaped)
  # standardize the input data
  std_data = scaler.transform(input_data_reshaped)
  print(std_data)
  prediction = trained_model.predict(std_data)
  print(prediction)
  if (prediction[0] == 0):
    return False
  else:
    return True

unittest.main()
```

```
----------------------------------------------------------------------
Ran 1 test in 0.002s

OK
[[  6.    148.     72.     35.      0.     33.6     0.627  50.   ]]
[[ 0.64602007  0.85464203  0.14982594  0.90469953 -0.68957614  0.20596027
   0.47274601  1.43277032]]
[1]
```

Entegrasyon Testleri Tasarımı

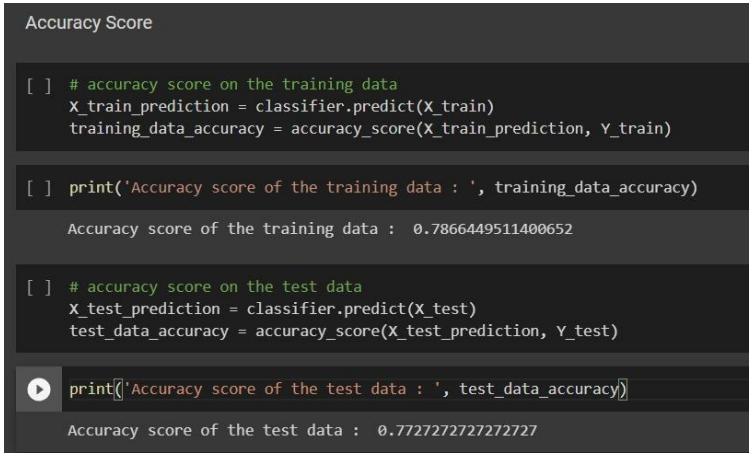
Yazılım geliştirmede entegrasyon testi, bir sistemin veya uygulamanın farklı bölümlerinin beklendiği gibi birlikte çalışıp çalışmadığını kontrol eden bir test türüdür. Bu testler genellikle bireysel bileşenler test edildikten sonra gerçekleştirilir ve farklı bileşenlerin beklenen şekilde entegre edilebilmesini ve birlikte çalışabilmesini sağlamayı amaçlar. Entegrasyon testleri, farklı modüllerin veya alt sistemlerin entegrasyonunun test edilmesini veya bir uygulamanın harici sistemler veya hizmetlerle entegrasyonunun test edilmesini içerebilir.

- Web uygulaması, bir python kütüphanesi olan streamlit ile geliştirilmiştir.
- Kullanıcıdan alınan veriler veri setine eklenmekte ve veritabanına kaydedilmektedir.
- Test sonucu uygulama üzerinden gösterilmekte ve ayrıca mail olarak tüm bilgiler gönderilmektedir.

Sistem Testleri Tasarımı

Sistem testi, belirtilen gereksinimleri karşıladığını ve amaçlandığı gibi çalıştığını doğrulamak için tüm sistemin veya uygulamanın test edilmesini içeren bir tür yazılım testidir. Sistem testi, sistemin performansının, güvenilirliğinin, güvenliğinin ve diğer işlevsel olmayan gereksinimlerinin test edilmesini içerebilir. Ayrıca, sistemin kullanıcı arayüzünün ve kullanıcı deneyiminin test edilmesinin yanı sıra sistemin diğer sistemler veya harici hizmetlerle entegrasyonunun test edilmesini de içerebilir.

- Sistem performansını etkileyecek gereksiz işlemlerden kaçınılmıştır.
- Uygulamanın hastalık tespitindeki doğruluk değeri %78 dir.
- Uygulamaya girilen veriler üçüncü kişilerle paylaşılmaz.



Kabul Testleri Tasarımı

Sisteme gerekli değerler girildiğinde, sistem doğru sonuç vermektedir. Ancak bu doğru sonuca ulaşmak için aşağıdaki kurallara uygun veri girişi yapılmalıdır.

- Ad, soyad, Eposta bilgileri boş olmamamalıdır.
- Gebelik sayısı, Glikoz miktarı, Kan basıncı, Cilt kalınlığı değeri, İnsülin miktarı, BMI, Diyabet soyağacı fonksiyonu, Yaş gibi test sonuçlarının boş olmaması ve negatif değer girilmemesi gerekir.
- Test sonuçlarına rakamdan başka karakter girilmemesi gerekir.

Dinlediğiniz için Çok Teşekkür ederiz