TC MARMARA ÜNİVERSİTESİ TEKNOLOJİ FAKÜLTESİ

2024 – 2025 EĞİTİM ÖĞRETİM YILI

GÜZ DÖNEMİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

VERİ TABANI YÖNETİM SİSTEMLERİ DERSİ

PROJE-FİNAL DÖKÜMANI

1. **Gerçekleştirdiğiniz veri tabanı projesi için grup arkadaşlarınızın isimlerini yazınız ve projenize ait veri tabanı/diğer yazılım bileşenleri hakkında bilgi veriniz.**

Eren ŞEREMET ve Abdulsamet ÇAKIR

Projede frontend tarafı için HTML, CSS, JavaScript, Bootstrap4 kullanıldı.

HTML ile arayüzün iskelet yapısı oluşturulduktan sonra yazdığımız CSS kodları ve Bootstap4’teki class yapıları kullanılarak bu yapıya şekil verildi. JavaScript ile web sitesine dinamik bir yapı kazandırıldı.

Backend taraf için PHP programlama dili ve veri tabanı olarak MySQL veri tabanı kullanıldı.

**HTML:** Web sayfalarının yapısını belirlemek için kullanılan bir işaretleme dilidir. Web sayfanızdaki metin, görseller, bağlantılar, formlar gibi içerikleri yapılandırır. HTML, web sayfasının temel iskeletini oluşturur.

**CSS**: HTML ile yazılmış içeriğin stilini (renkler, yazı tipleri, hizalamalar vb.) düzenlemek için kullanılır. HTML sayfasının görsel görünümünü ve düzenini tanımlar. Bootstrap gibi CSS çerçeveleri, hızlı ve etkili tasarım yapmanızı sağlar.

**Javascript:** Web sayfalarına etkileşim ve dinamiklik eklemek için kullanılan bir programlama dilidir. Örneğin, kullanıcı bir bağış formunu gönderdiğinde sayfanın yeniden yüklenmeden güncellenmesi, bir form doğrulama işlemi gibi etkileşimler JavaScript ile yapılır.

**Bootstrap4:** Popüler bir açık kaynaklı CSS ve JS çerçevesidir. Web sayfalarını hızlıca tasarlamak için hazır şablonlar ve bileşenler sunar.

**PHP(Hypertext Preprocessor):** Dinamik web sayfaları oluşturmak için kullanılan sunucu taraflı bir programlama dilidir. PHP, HTML içine gömülü olarak çalışabilir ve veri işleme, form verilerini alma, kullanıcı oturumlarını yönetme gibi işlemleri yapar. Web sunucusunda çalışır ve veritabanı işlemleri için kullanılır.

**MySQL:** Açık kaynaklı bir ilişkisel veritabanı yönetim sistemidir. Web projelerinde genellikle kullanıcıverileri ve diğer uygulama verilerini saklamak için kullanılır. PHP ile MySQL arasında veri alışverişi yapmak yaygın bir uygulamadır. SQL sorguları kullanarak verileri ekleyebilir, silebilir, güncelleyebilir ve sorgulayabiliriz.

1. Githublink
2. **Gerçekleştirdiğiniz projenin amacını detaylı bir şekilde açıklayınız.**

Bağış takip sistemi bireylerin bağışlarını yönetmelerini, izlemelerini, kurumların etkinlikler ve kampanyalar oluşturarak bağışçılara ulabilmesini sağlayan bir platformdur. Bu platform bağış sürecini şeffaf ve düzenli hale getirmektedir. Bağışçılar bağışlanan paraları doğru bir şekilde tekip edebilir. Bağışların etkin bir şekilde kullanılmasını sağlar. Uygulama amaçlarını başlıklar altında toplarsak:

1. **Bağış Sürecini Kolaylaştırmak ve Yönetmek:** 
   1. Kullanıcılar, sisteme giriş yaparak bağışlarını hızlıca tamamlar ve geçmişteki bağışlarını inceleyebilir.
   2. Bağış tutarları, kullanıcı bilgileri, bağışın yapıldığı tarih ve diğer detaylar kaydedilir.
2. **Bağışların İzlenmesi ve Raporlanması:**
   1. Hangi etkinliklere bağış yapıldığı ve bağış tutarları analiz edilebilir.
   2. Bağışçıların coğrafi konumları, meslek grupları gibi veriler toplanarak, bağışçı profilleri oluşturulabilir.
3. **Şeffaflık ve Güven Sağlamak**
   1. Bağışçıların yaptıkları bağışların hangi etkinliklere harcandığını gösteren raporlar oluşturulabilir.
   2. Bağışçılara, yaptıkları bağışlar için otomatik teşekkür e-postaları gönderilebilir.
4. **Veritabanı Yönetimi ve Kolay Erişim**
   1. Bağışçıların tüm bilgileri güvenli bir veritabanında depolanır ve gerektiğinde kolayca erişilir.
   2. Veritabanı sayesinde geçmişteki tüm bağışlar, zaman dilimlerine göre sıralanabilir ve analiz edilebilir.
5. **Bağış Kampanyaları ve Etkinlikleri Yönetmek**
   1. Farklı kampanyalar için belirlenen hedeflere ulaşmak amacıyla bağış toplama işlemleri kolayca organize edilir.
   2. Kampanya süresi boyunca bağış miktarları ve bağışçılar raporlanabilir.
6. **Zamanında ve Hedef Odaklı Bağış Yönetimi**
   1. Bağış kampanyaları için hedef tutarları belirlenir ve ilerleme grafiklerle görselleştirilir.
   2. Bağışçılar, bağışladıkları tutarı ve kampanya hedefine katkılarını kolayca izleyebilir.
7. **Tasarladığınız veri tabanı mimarisinde hangi tablo ve ilişkileri kullanıldığınızı**

**açıklayınız.**

**recipient**

**donor**

**recipient\_id (int) <PK>**

institution\_name(varchar(100))

email (varchar(200))

phone (varchar(20))

address (varchar(150))

activity (varchar(100))

purpose (text)

founding\_date (date)

photo (varchar(100))

**donor\_id (int) <PK>**

first\_name (varchar(50))

last\_name (varchar(50))

birthday (date)

job\_title (varchar(50))

institution (varchar(50))

email (varchar(150))

phone (varchar(20))

**donation**

**donor-event**

donor\_id <FK>

event\_id <FK>

registration (date)

donation\_id <PK>

donor\_id <FK>

event\_id <FK>

amount (decimal)

note (text)

donation\_date (date)

**event**

**event\_id (int) <PK>**

recipient\_id <FK>

event\_name (varchar(200))

image (varchar(100))

location (varchar(200))

event\_date (date)

description (text)

created\_time (datetime)

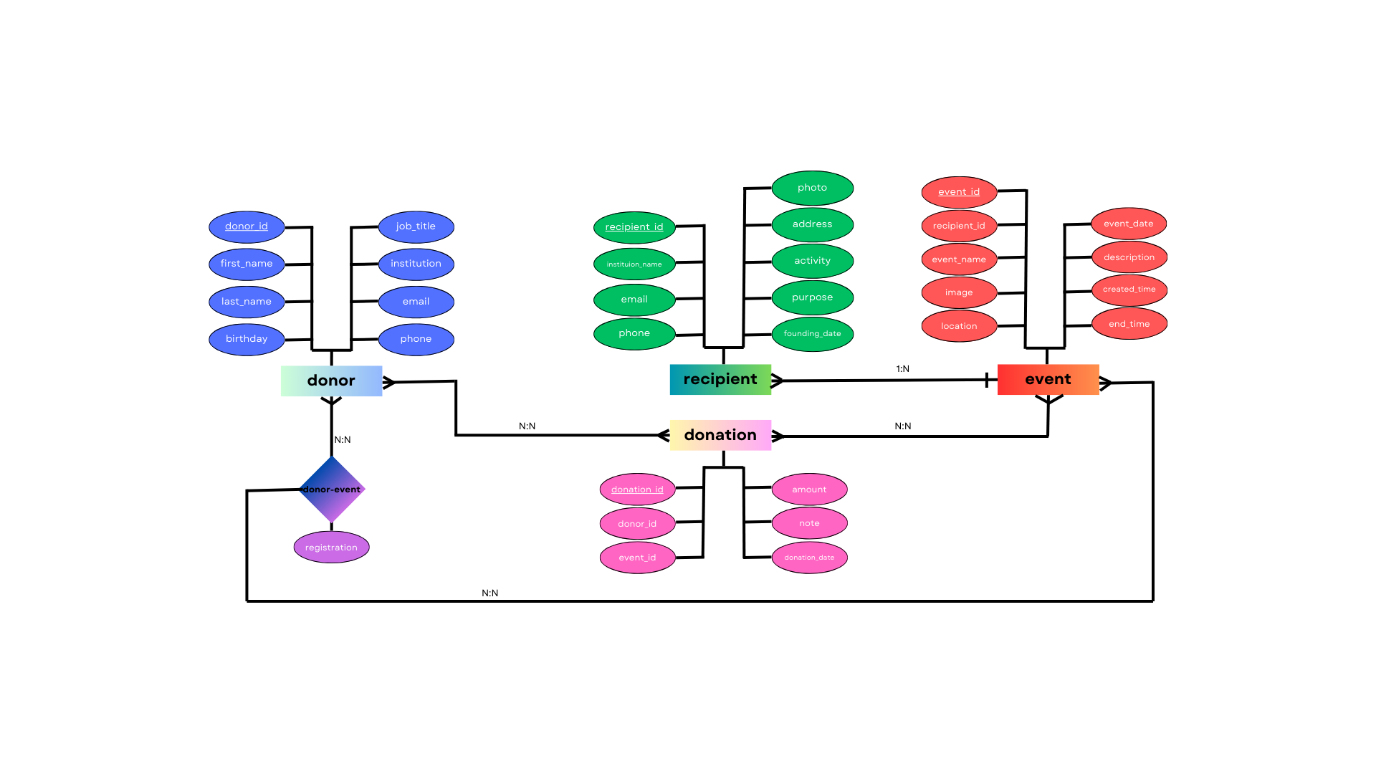
end\_time (datetime)

* donationdb
  + Proje için oluşturulan veri tabanı adı
* **donor table**
  + donor\_id: Bağışçı için benzersiz id numarası
  + first\_name: Bağışçı ismi
  + last\_name: Bağışçı soyismi
  + birthday: Bağışçı doğum tarihi, zorunlu değil
  + job\_title: Bağışçı mesleği, zorunlu değil
  + institution: Bağışçının çalıştığı kurum, zorunlu değil
  + email: Bağışçının benzersiz email adresi
  + phone: Bağışçı telefon numarası

Bağışçı herhangi bir yerde çalışmıyor olabilir veya bunu belirtmek istemiyor olabilir. Aynı şekilde birthday ve job\_title alanları için de geçerli. Bu nedenle bu alanlar **DEFAULT NULL** olarak ayarlandı.

* **recipient table**
  + recipient\_id: Kurum için benzersiz id numarası
  + institution\_name: Kurum adı, zorunlu
  + email: Kurum için benzersiz email adresi
  + phone: Kurum telefon numarası
  + address: Kurum adresi
  + activity: Kurumun kısaca kuruluş amacı etkinliği
  + purpose: Kurumun kuruluş amacı, kurumun kurulma nedeni
  + founding\_date: Kurumun kuruluş tarihi
  + photo: Kurum sembolü, görseli
* **event table**
  + event\_id: Etkinlikler için benzersiz id numarası
  + recipient\_id: Etkinliği hangi kurumun oluşturduğunu gösteren, recipient table’dan kurum bilgilerini almak için attribute
    - **event ile recipient arasında Many-to-One ilişkisi bulunmaktadır. Yani bir kurum birden fazla etkinlik düzenleyebilir fakat her etkinlik yalnızca bir kurum tarafından düzenlenebilir.**
  + event\_name: Etkinlik adı
  + image: Etkinliği temsil eden bir görsel
  + location: Etkinliğin nerede yapılacağı
  + description: Etkinliğin amacını, nedenlerini içeren bir açıklama metni
  + created\_time: Etkinlik oluşturulma zamanı
  + end\_time: Etkinlik sona erme zamanı
* **donor-event table**
  + donor\_id: Etkinliğe bağış yapan bağışçı id numarası. Bu id numarası ile donor table’dan bağışçı bilgilerine erişilebilir.
  + event\_id: Bağış yapılan etkinlik id numarası. Bu id numarası ile event table’dan etkinlik bilgilerine erişilebilir.
  + registration: Bağışçının etkinliğe kayıt olduğu zaman
    - **Bu tablo donor ile event arasında Many-to-Many ilişkiyi gösteren tablodur. Yani bir bağışçı birden fazla etkinliğe katılabilir ve bir etkinlikte birden fazla bağışçı bulunabilir.**
* **donation table**
  + donation\_id: Bağışlar için benzersiz id numarası
  + donor\_id: Bağış yapan bağışçı ad numarası. Bu id numarası ile donor table’dan bağışçı bilgilerine erişilebilir.
  + event\_id: Bağış yapılan etkinlik id numarası. Bu id numarası ile event table’dan etkinlik bilgilerine erişilebilir.
  + amount: Bağışçının o etkinlikte yaptığı bağış miktarı
  + note: Bağışçının etkinlik için bıraktığı not, mesaj
  + donation\_date: Bağışçının bağış yaptığı tarih
    - **Bu tablo donor ile event arasında Many-to-Many ilişkiyi gösteren tablodur. Yani bir bağışçı birden fazla etkinlikte bağış yapabileceği gibi bir etkinliğe birden fazla bağışçı bağış yapabilir.**

1. Veri tabanı ER (Entity Relationship) diagramının bilgisayar ortamında çizilmiş halini paylaşınız. (Ara raporda eksik kısımlar bu raporda giderilmelidir ve ER çizme programlarından faydalanıbilir. Elle çizim, çizip fotoğrafını çekme vb. kabul edilmeyecektir.)



1. Herhangi iki tablonuz için DDL (create) kodları yazılmalıdır.

* donor table

CREATE TABLE donor (

donor\_id INT PRIMARY KEY AUTO\_INCREMENT,

first\_name VARCHAR(100) NOT NULL,

last\_name VARCHAR(100) NOT NULL,

birthday DATE DEFAULT NULL,

job\_title VARCHAR(100) DEFAULT NULL,

institution VARCHAR(100) DEFAULT NULL,

email VARCHAR(200) NOT NULL UNIQUE,

phone VARCHAR(20) NOT NULL UNIQUE

)

* event table

CREATE TABLE event (

event\_id INT PRIMARY KEY AUTO\_INCREMENT,

recipient\_id INT,

event\_name VARCHAR(100) NOT NULL,

image VARCHAR(100) DEFAULT NULL,

location VARCHAR(100) NOT NULL,

description TEXT NOT NUL,

created\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

end\_time DATETIME

FOREIGN KEY (recipient\_id) REFERENCES recipient(recipient\_id)

)

1. 5 adet DML (update, insert, delete) içeren kodları yazılmalıdır.

* $stmt = $db->prepare("INSERT INTO events (recipient\_id, event\_name, image, location, description, end\_time) VALUES (?, ?, ?, ?, ?, ?)");

$stmt->bind\_param('isssss', $recipient\_id, $event\_name, $image\_name, $location, $description, $end\_time);

* $firstname = $\_POST["firstname"];

$lastname = $\_POST["lastname"];

$birthday = $\_POST["birthday"];

$jobtitle = $\_POST["jobtitle"];

$institution = $\_POST["institution"];

$email = $\_POST["email"];

$phone = $\_POST["phone"];

$psw = $\_POST["password"];

$sql = "INSERT INTO donor (first\_name, last\_name, birthday, job\_title, institution, email, phone, password)

VALUES ('$firstname', '$lastname', '$birthday', '$jobtitle', '$institution', '$email', '$phone', '$psw')";

* $user\_id = $\_SESSION[“recipient\_id”];

$institution = $\_POST[“institution”];

$sql = “UPDATE recipient SET institution = ‘$institution’

WHERE recipient\_id = ‘$user\_id’”;

* $user\_id = $\_SESSION[“recipient\_id”];

$event\_id = $\_GET[“event\_id”];

$sql = “DELETE FROM event

WHERE recipient\_id = ‘$user\_id’ AND event\_id = ‘$vent\_id’”;

* $user\_id = $\_SESSION[“donor\_id”];

$firstname = $\_POST[“firstname”];

$lastname = $\_POST[“lastname”];

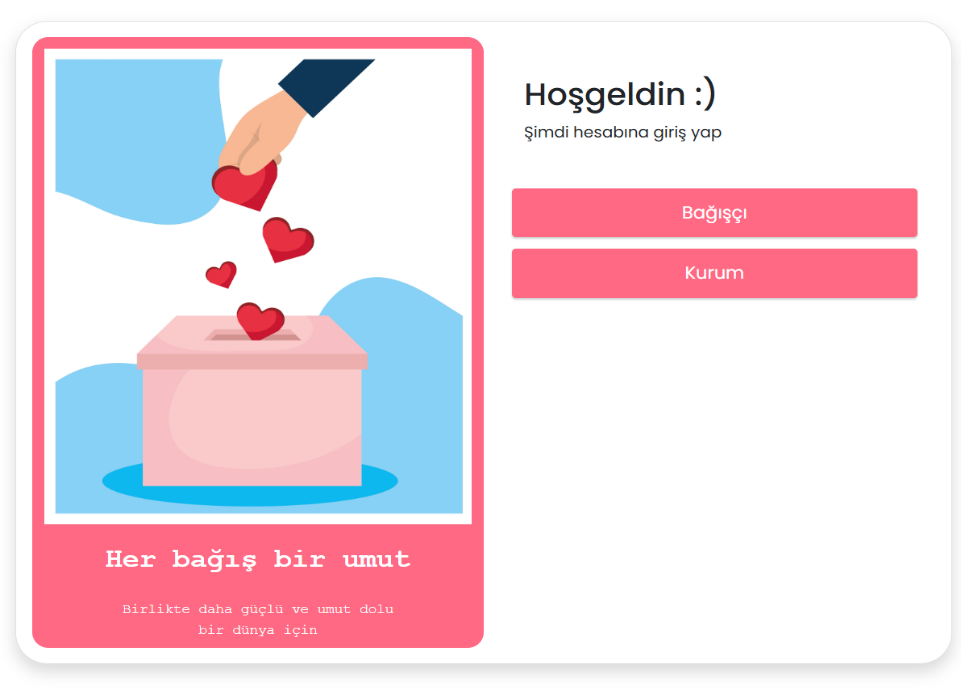
$sql = “UPDATE donor SET first\_name = ‘$firstname’ AND last\_name = ‘$lastname’ WHERE donor\_id = ‘$user\_id’”;

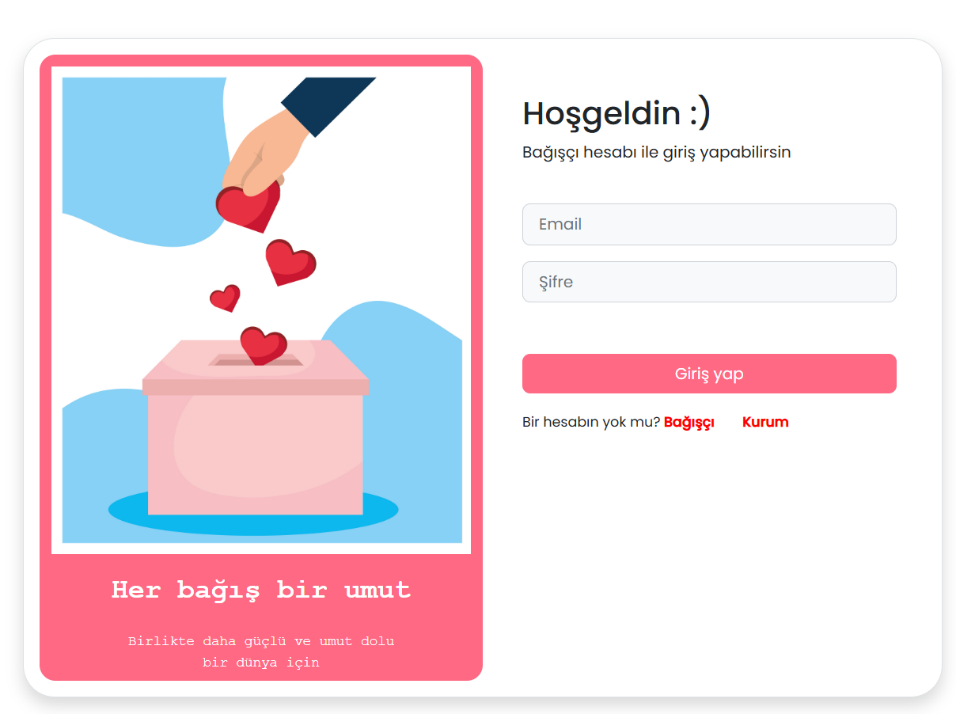
1. Projenize ait kendi belirlediğiniz 10 adet SQL sorgusu yazınız, sorguların amacını ve sonuç çıktısını da lütfen ekleyiniz. (Açıklama: Sorgular Select deyimleri ve gruplama fonksiyonlarını HAVING deyimini (min, max, avg, count gibi) ve join deyimlerini (en az iki tablo ile birleştirme sorgusu) içerecek şekilde basitten karmaşığa doğru gitmelidir. Proje sunum anında veri tabanınıza ait sorular SQL ortamında gösterilecek ve açıklanacaktır. Raporunuzda ise sorgular, sorguların cevap ve sonuçlarının ekran görüntüsü olarak paylaşılması beklenmektedir.

* $email = $\_POST["email"];

$password = $\_POST["password"];

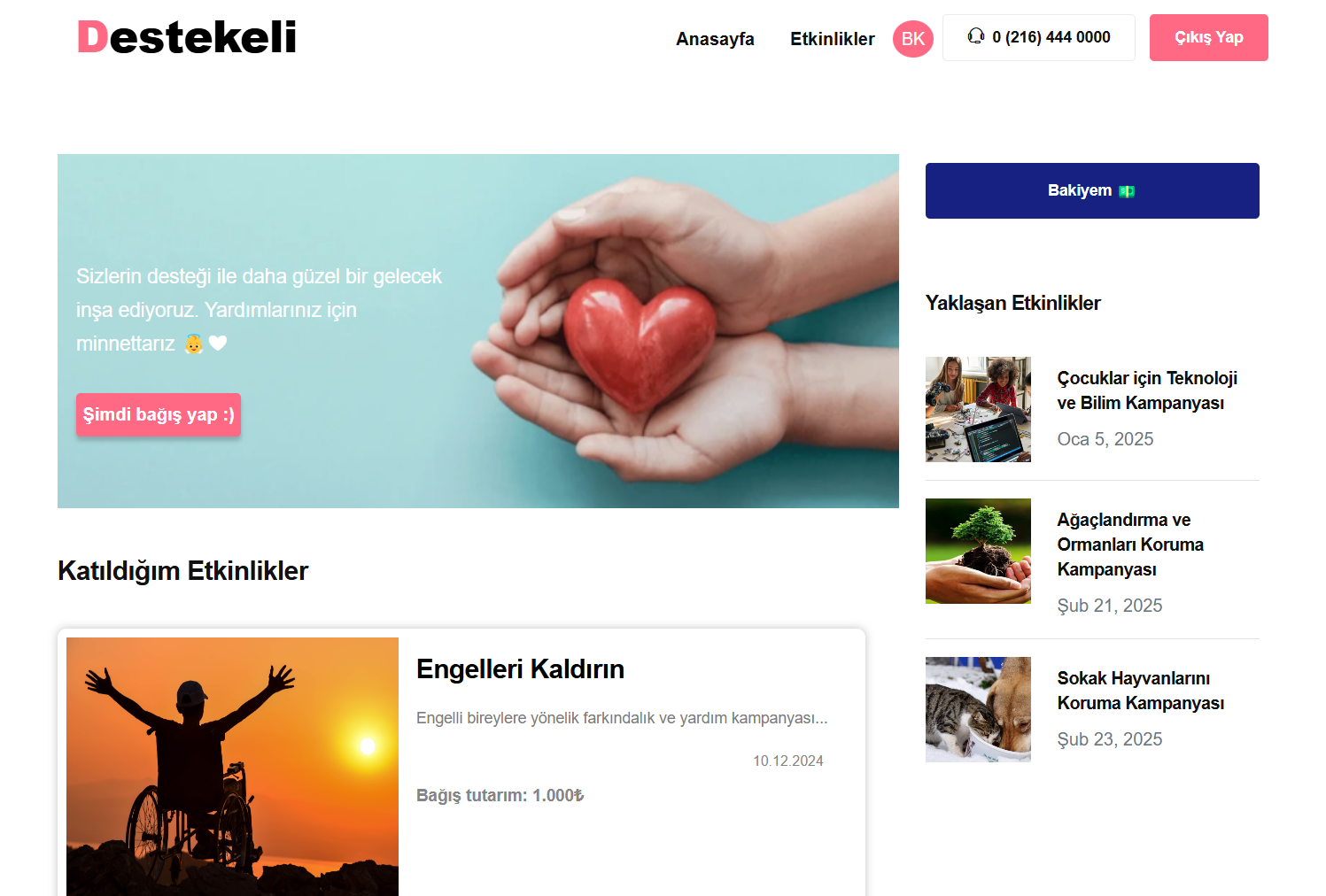
$sql = "SELECT \* FROM donor WHERE email = '$email' AND password = '$password'";

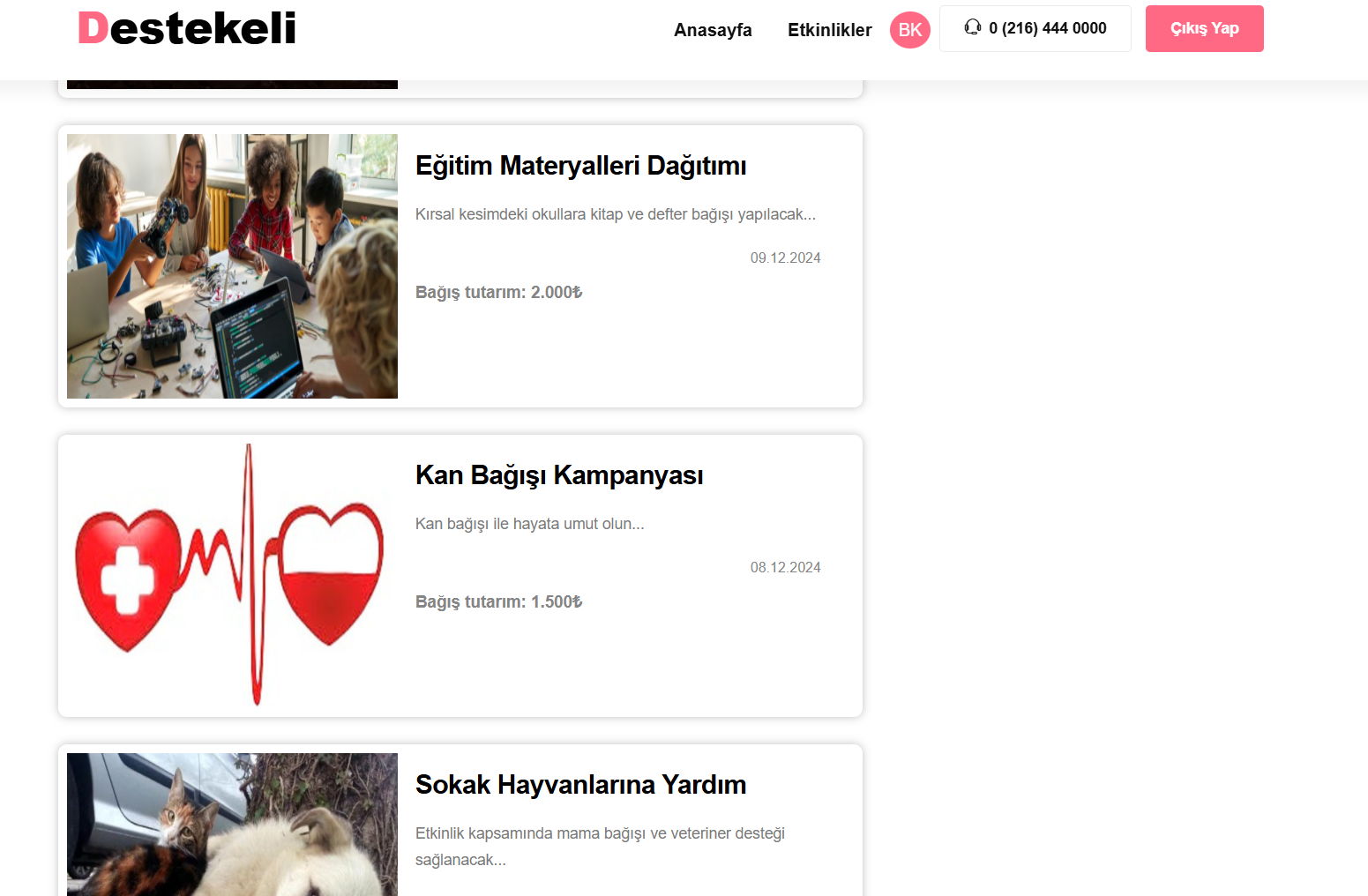




* $donor\_id = $\_SESSION["user\_id"];

$sql = "SELECT \* FROM donor WHERE donor\_id = $donor\_id";





* $institution = $\_POST["institution"];

$foundingdate = $\_POST["foundingdate"];

$purpose = $\_POST["purpose"];

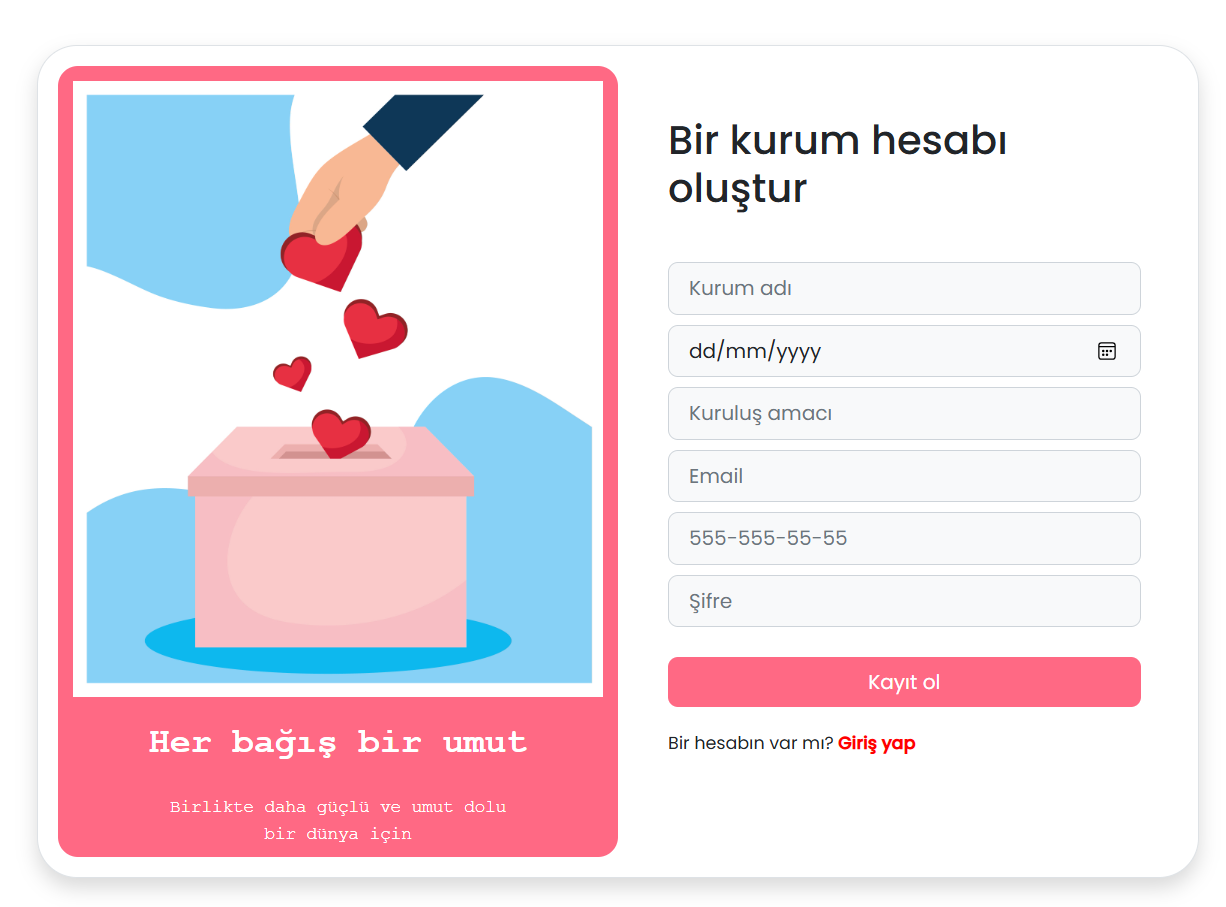
$email = $\_POST["email"];

$phone = $\_POST["phone"];

$psw = $\_POST["password"];

$sql = "INSERT INTO recipient (institution\_name, founding\_date, purpose, email, phone, password)

VALUES ('$institution', '$foundingdate', '$purpose', '$email', '$phone', '$psw')";



* $event\_name = htmlspecialchars($\_POST['event\_name']);

$location = htmlspecialchars($\_POST['location']);

$description = htmlspecialchars($\_POST['description']);

$end\_time = $\_POST['end\_time'];

$recipient\_id = $\_SESSION["user\_id"];

// Resmi yükle

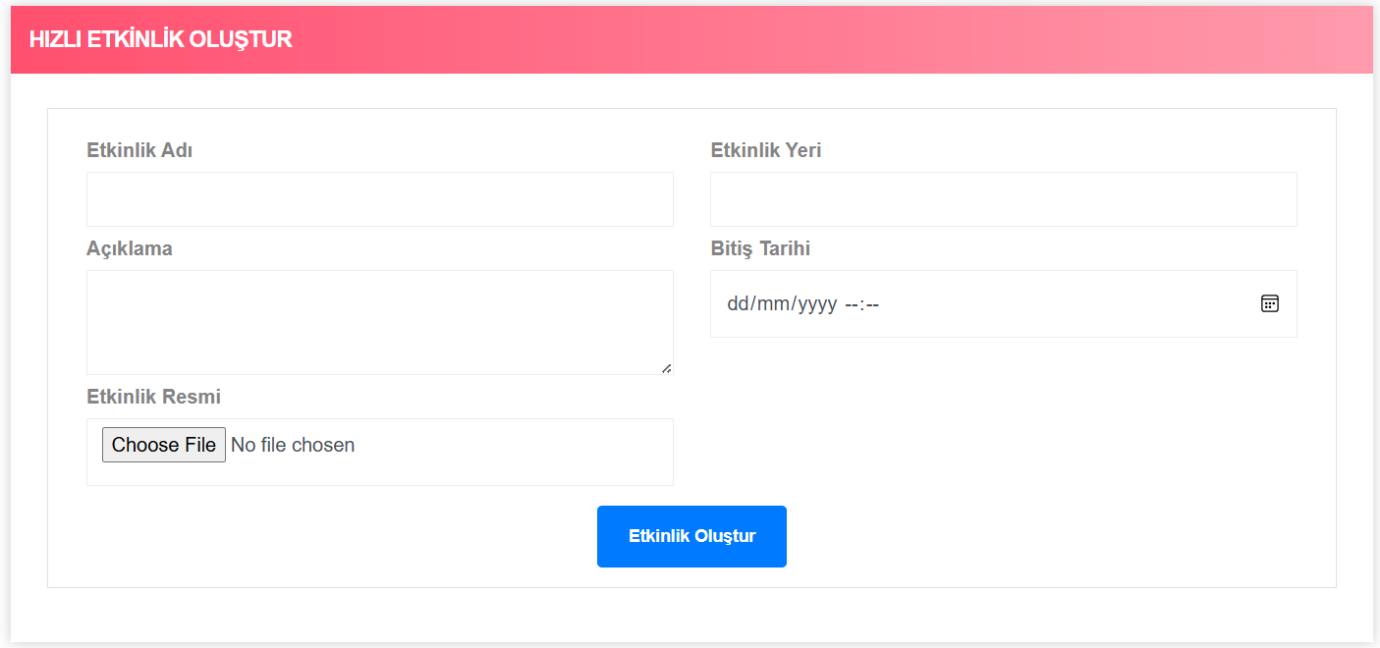
$image\_name = rand(100000000, 999999999) . '.' . pathinfo($\_FILES['image']['name'], PATHINFO\_EXTENSION);

$image\_tmp\_name = $\_FILES['image']['tmp\_name'];

$image\_path = "../uploads/" . basename($image\_name);

$stmt = $db->prepare("INSERT INTO events (recipient\_id, event\_name, image, location, description, end\_time) VALUES (?, ?, ?, ?, ?, ?)");

$stmt->bind\_param('isssss', $recipient\_id, $event\_name, $image\_name, $location, $description, $end\_time);



* $donor\_id = $\_SESSION["user\_id"];

$sql = "SELECT de.\*, e.\*, r.\*, do.\*

FROM donor\_event de

INNER JOIN events e

ON de.event\_id = e.event\_id

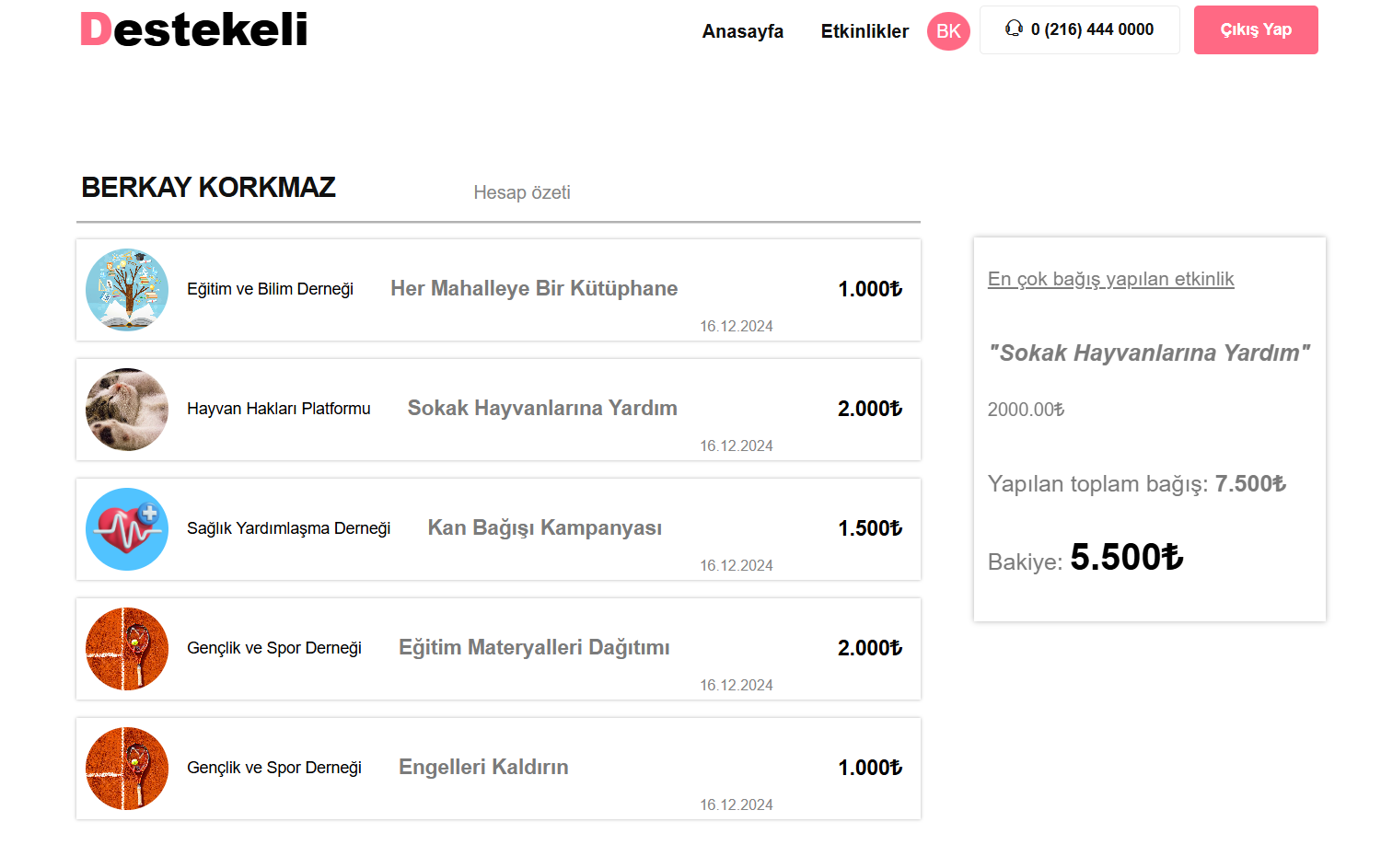
INNER JOIN recipient r

ON r.recipient\_id = e.recipient\_id

INNER JOIN donation do

ON do.event\_id = de.event\_id

WHERE do.donor\_id = '$donor\_id' AND de.donor\_id = '$donor\_id'";



* // LEFT JOIN ile etkinliklere bağış toplamlarını dahil et

$stmt = $db->query('

SELECT e.\*,

COALESCE(SUM(d.amount), 0) AS total\_amount

FROM events e

LEFT JOIN donation d ON e.event\_id = d.event\_id

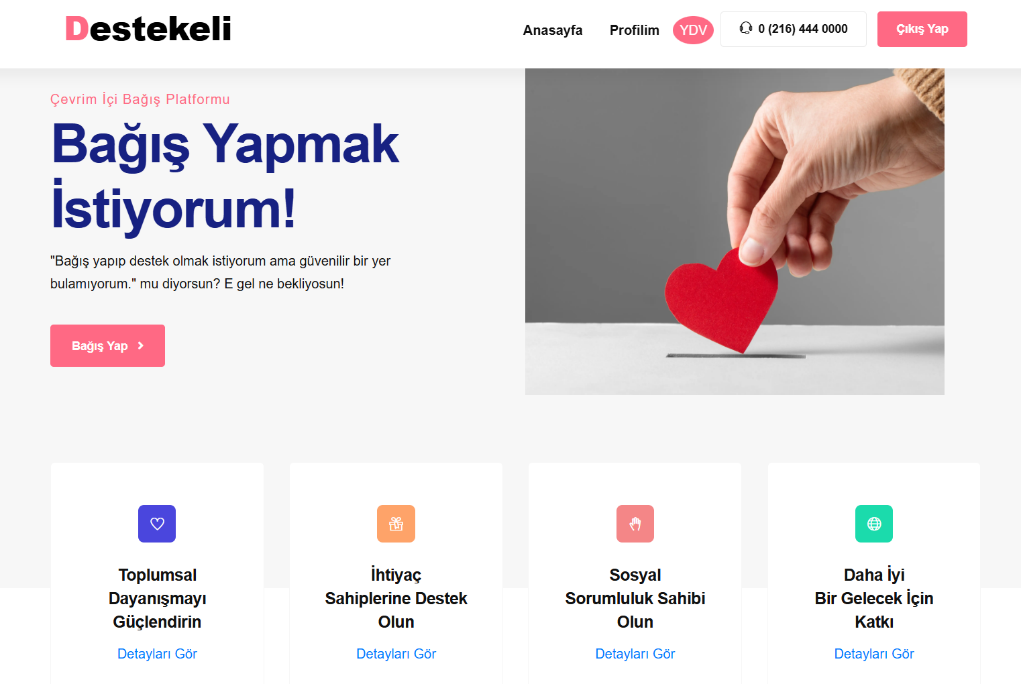
GROUP BY e.event\_id

ORDER BY total\_amount DESC

LIMIT 3;

');

$events = $stmt->fetch\_all(MYSQLI\_ASSOC);





* // Güncelleme Sorgusu

$updateQuery = "UPDATE recipient SET institution\_name = ?, email = ?, phone = ?, password = ?, address = ?, activity = ?, purpose = ?, founding\_date = ? WHERE recipient\_id = ?";

$updateStmt = $db->prepare($updateQuery);

$updateStmt->bind\_param("ssssssssi", $institution\_name, $email, $phone, $password, $address, $activity, $purpose, $founding\_date, $id);

$updateStmt->execute();

// Güncellenmiş veriyi yeniden çek

$stmt->execute();

$result = $stmt->get\_result();

$recipient = $result->fetch\_assoc();

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

* // Kayıt Silme

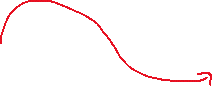
$query = "DELETE FROM recipient WHERE recipient\_id = ?";

$stmt = $db->prepare($query);

$stmt->bind\_param("i", $id);

metin, yazı tipi, sayı, numara, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu**9. Eğer gerçekleştirmiş iseniz, veri tabanı bağlama ve uygulama geliştirme aşamalarınızı kısaca açıklayarak, kullanıcı ara yüz ekranından bir örnek veriniz. Ve geliştirdiğiniz ara yüzü anlatınız. (10 p)**



**metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* $db değişkeni ile localde açtğımız sunucumuzu oluşturup veritabani yapımızı kullanıyoruz.
* Bu kod bloğunu bir php dosyasına ekleyip her sayfamızda çağırarak da kod tekrarının önüne geçmiş oluyoruz.

**10. View nedir açıklayınız ve bir adet view, bir adet saklı yordam (Stored Procedute) ifadesine ait SQL deyimlerinin sorgusunu ve cevabını yazınız. (10 p)**

**VİEW İÇİN KOD**

* CREATE VIEW TotalAmountPerUser AS SELECT UserID, SUM(Amount) AS TotalAmount FROM Donations GROUP BY UserID;
* SELECT \* FROM TotalAmountPerUserm WHERE UserID = 3;

**SP İÇİN KOD**

* CREATE PROCEDURE GetHighValueDonors(@MinDonationAmount DECIMAL(10, 2))

AS

BEGIN

SELECT d.FullName AS DonorName, d.Email AS DonorEmail, SUM(dt.Amount) AS TotalDonated

FROM Donors d

JOIN Donations dt ON d.DonorID = dt.DonorID

GROUP BY d.FullName, d.Email

HAVING SUM(dt.Amount) > @MinDonationAmount;

END;

* EXEC GetHighValueDonors @MinDonationAmount = 1000.00;