**Php Üyelik Sistemi**

PDO kullanarak PHP ile üyelik sistemi oluşturmak oldukça kolaydır. Bu makalede, PHP PDO kullanarak üyelik sistemi oluşturmanın adımlarını ayrıntılı olarak ele alacağız.

* Veritabanı ve Kayıt Formunun Oluşturulması
* Php Üyelik Sistemi için Veritabanı Bağlantısının Sağlanması
* Kullanıcı Kayıt İşlemi
* Kullanıcı Giriş Formunun Oluşturulması ve Giriş İşlemi
* Oturum Yönetimi
* Kullanıcı Çıkış İşlemi

**Adım 1 : Veritabanı ve Kayıt Formunun Oluşturulması**

Öncelikle,php üyelik sistemi için bir veritabanımızı ve tablonuzu oluşturuyoruz.

**CREATE** **DATABASE** ornek;

**USE** ornek;

**CREATE** **TABLE** **`uyeler`** (

**`uye\_id`** int(11) **NOT NULL** AUTO\_INCREMENT,

**`uye\_kadi`** varchar(50) **NOT NULL**,

**`uye\_sifre`** varchar(20) **NOT NULL**,

**`uye\_eposta`** varchar(50) **NOT NULL**,

**PRIMARY KEY** (**`uye\_id`**))

SQL **create database**komutu ile örnek veritabanımızı oluşturuyoruz. **Use**komutu ile örnek isimli veritabanını kullanacağımızı belirttikten sonra örnek veritabanı içerisine **create table** komutu ile üyeler isimli tablomuzu oluşturuyoruz.

Veritabanı ve tablomuzu oluşturduktan sonra kullanıcının sisteme kayıt olabilmesi için bir HTML kayıt formu oluşturuyoruz. Kayıt formumuzun daha düzenli ve modern görünmesi için [bootstrap](https://getbootstrap.com/docs/5.3/getting-started/download/)kütüphanesini kullanıldı.

Kayıt formumuzu **kayit.php**isimli dosyamızda oluşturalım.

<!doctype html>

<**html** lang="tr">

<**head**>

<**meta** charset="UTF-8">

<**meta** name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<**meta** http-equiv="X-UA-Compatible" content="ie=edge">

<**title**>Kayıt Formu</**title**>

<**link** rel="stylesheet" href="boot/css/bootstrap.min.css">

</**head**>

<**body**>

<**div** class="container-fluid">

<**div** class="row m-5">

<**div** class="col-3">

<**form** action="" method="post">

<**legend**>Kayıt Formu</**legend**>

<**div** class="mb-3">

<**label** for="" class="form-label">Eposta:</**label**>

<**input** type="email" class="form-control" id="" name="uye\_eposta" required>

</**div**>

<**div** class="mb-3">

<**label** for="" class="form-label">Kullanıcı Adı:</**label**>

<**input** type="text" class="form-control" id="" name="uye\_kadi" required>

</**div**>

<**div** class="mb-3">

<**label** for="" class="form-label">Şifre:</**label**>

<**input** type="password" class="form-control" id="" name="uye\_sifre" required>

</**div**>

<**div** class="mb-3">

<**label** for="" class="form-label">Şifre Tekrar:</**label**>

<**input** type="password" class="form-control" id="" name="uye\_sifre\_tekrar" required>

</**div**>

<?php

if (isset($msg)) { ?>

<**div** class="alert <?php echo $status; ?>" role="alert">

<?php echo $msg; ?>

</**div**>

<?php } ?>

<**button** class="btn btn-primary" type="submit">Kayıt Ol</**button**>

</**form**>

</**div**>

</**div**>

</**div**>

</**body**>

</**html**>

Yukarıdaki html kodlar ile formu direkt oluşturabilirsiniz. Bootstrap dosyalarının doğru konumda olduğuna dikkat etmeniz yeterli.



**Adım 2 : Php Üyelik Sistemi için Veritabanı Bağlantısının Sağlanması**

**baglanti.php** isimli dosyamızı oluşturup veritabanı bağlantımız için gerekli kodları yazıyoruz.

<?php

error\_reporting(0); // hataları bastırıyoruz hataları görmek için satırı silmeniz yeterli

session\_start(); // oturum başlatıyoruz

## Bağlantı Değişkenleri ##

**$hostname** = "localhost";

**$username** = "root";

**$pass** = "";

**$database** = "ornek";

## Mysql Bağlantı ##

**try** {

**$db** = new PDO("mysql:host=" . **$hostname** . "; dbname=" . **$database** . "; charset=utf8", "$username", "$pass");

} **catch** (PDOException **$error**) {

**print** **$error**->getMessage();

**exit**();

}

?>

Yukarıdaki kod parçacığında veri tabanı bağlantımızda kullanacağımız değerleri değişkenlere atıyoruz. Daha sonrasında try catch kontrol yapısı ile hata ayıklama modunu etkinleştiriyoruz.

**Adım 3 : Kullanıcı Kayıt İşlemi**

Php ile bir üyelik sistemi oluştururken öncelikle üyelerin kayıt olabileceği bir alan oluşturmamız gerekir. Bazı sistemlerde üyelerin yetkili kullanıcı tarafından tanımlandığını görebilirsiniz ancak mantık aynıdır.

Kullanıcı kayıt işlemi için yine **kayit.php** dosyamızda işlem yapmaya devam ediyoruz. Bu aşamada formumuzu tanımlarken kullandığımız form elemanlarının name değerleri ile formdan gelen verileri **$\_POST** methodu ile alacağız.

<?php

**include** "baglanti.php";

**if** (**$\_SERVER[**"REQUEST\_METHOD"] == "POST") {

**$uye\_kadi** = trim(**$\_POST[**"uye\_kadi"]);

**$uye\_sifre** = trim(**$\_POST[**"uye\_sifre"]);

**$uye\_sifre\_tekrar** = trim(**$\_POST[**"uye\_sifre\_tekrar"]);

**$uye\_eposta** = trim(**$\_POST[**"uye\_eposta"]);

**if** (**empty**(**$uye\_kadi**) || **empty**(**$uye\_sifre**) || **empty**(**$uye\_eposta**)) {

**$msg** = 'Yıldızlı alanlar boş bırakılamaz.';

**$status** = 'alert-danger';

} **else** {

**$ayni\_uye\_varmi** = **$db**->prepare("SELECT \* FROM uyeler WHERE uye\_kadi = ?");

**$ayni\_uye\_varmi**->execute(**array**(**$uye\_kadi**));

**if** (**$ayni\_uye\_varmi**->rowCount()) {

**$msg** = 'Bu kullanıcı adı zaten kullanılıyor. Farklı bir kullanıcı adı deneyin.';

**$status** = 'alert-danger';

} **else** {

**if** (**$uye\_sifre** == **$uye\_sifre\_tekrar**) {

**$uye\_ekle** = **$db**->prepare("INSERT INTO uyeler (uye\_kadi, uye\_sifre, uye\_eposta) VALUES (?,?,?)");

**$uye\_ekle**->execute(**array**(**$uye\_kadi,** **$uye\_sifre,** **$uye\_eposta**));

**if** (**$uye\_ekle**) {

**$msg** = 'Kayıt işlemi tamamlandı.';

**$status** = 'alert-success';

} **else** {

**$msg** = 'Üye kaydı başarısız. Bir sorun oluştu.';

**$status** = 'alert-danger';

}

} **else** {

**$msg** = 'Şifreler uyuşmuyor kontrol ediniz!';

**$status** = 'alert-danger';

}

}

}

}

?>

Yukarıdaki kodumuz kademe kademe incelediğimizde ilk önce **baglanti.php** isimli dosyamızı **kayit.php** isimli dosyamıza dahil ediyoruz.Daha sonra **POST**isteği olup olmadığını kontrol ediyoruz.

Veritabanı bağlantımızı sağladıktan sonra form elementlerinden gelen verileri elementlerin name değerlerini kullanarak **POST**metodu ile alıyoruz ve değişkene atıyoruz. Kullanıcıdan aldığımız verileri daha detaylı kontrol geçirmemiz güvenlik açısından önemlidir. Fakat konumuz üyelik sisteminin mantığını anlamak olduğu için verilerde detaylı filtreleme yapmıyoruz.

Gelen verileri değişkene atadıktan sonra değişkene bir değer atanıp atanmadığını kontrol ediyoruz.

Daha sonra aynı kullanıcı adı ile farklı bir kullanıcının sistemde var olup olmadığını kontrol ediyoruz. Eğer kullanıcı adı daha önce kullanılmamış ise gelen veriler ile birlikte yeni kullanıcıyı veritabanımıza ekliyoruz.

**Adım 4 : Kullanıcı Giriş Formunun Oluşturulması ve Giriş İşlemi**

Üyelik sistemi oluştururken ki en önemli adım üyenin giriş yapabilmesidir. Üye sisteme giriş sağlayarak belirli sayfalara erişim kazanmalıdır.

Giriş formumuzu yine bootstrap kütüphanesi yardımı ile giris.php isimli dosyamızda oluşturuyoruz.



<!doctype html>

<**html** lang="tr">

<**head**>

<**meta** charset="UTF-8">

<**meta** name="viewport"

content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<**meta** http-equiv="X-UA-Compatible" content="ie=edge">

<**title**>Giriş</**title**>

<**link** rel="stylesheet" href="boot/css/bootstrap.min.css">

</**head**>

<**body**>

<**div** class="container-fluid">

<**div** class="row m-5">

<**div** class="col-3">

<**form** action="" method="post">

<**legend**>Giriş</**legend**>

<**div** class="mb-3">

<**label** for="" class="form-label">Kullanıcı Adı:</**label**>

<**input** type="text" class="form-control" id="" name="uye\_kadi" required>

</**div**>

<**div** class="mb-3">

<**label** for="" class="form-label">Şifre:</**label**>

<**input** type="password" class="form-control" id="" name="uye\_sifre" required>

</**div**>

<?php

if (isset($msg)) { ?>

<**div** class="alert <?php echo $status; ?>" role="alert">

<?php echo $msg; ?>

</**div**>

<?php } ?>

<**button** class="btn btn-success" type="submit">Giriş Yap</**button**>

</**form**>

</**div**>

</**div**>

</**div**>

</**body**>

</**html**>

Formumuzu tanımladıktan sonra kullanıcıdan gelen kullanıcı adı ve şifrelerimizi kontrol edip daha sonrasında oturum nesnesini tanımlayacağımız kodları yazıyoruz.

<?php

**include** 'baglanti.php';

**if** (**$\_SESSION[**'login']){

header('Location:index.php');

}

**if** (**$\_SERVER[**"REQUEST\_METHOD"] == "POST") {

**$kullanici\_adi** = trim(**$\_POST[**'uye\_kadi']);

**$sifre** = trim(**$\_POST[**'uye\_sifre']);

**if** (**empty**(**$kullanici\_adi**) || **empty**(**$sifre**)) {

**$msg** = 'Kullanıcı adı veya şifre boş bırakılamaz!';

**$status** = 'alert-danger';

} **else** {

**$kullanici\_kontrol** = **$db**->prepare("SELECT uye\_id,uye\_kadi, uye\_sifre FROM uyeler WHERE uye\_kadi = ? AND uye\_sifre = ?");

**$kullanici\_kontrol**->execute(**array**(**$kullanici\_adi,** **$sifre**));

**if** (**$kullanici\_kontrol**->rowCount() > 0) {

**$row** = **$kullanici\_kontrol**->fetch(PDO::FETCH\_OBJ);

**$\_SESSION[**'login'] = **true**;

**$\_SESSION[**'kullanici\_id'] = **$row**->uye\_id;

header("Location:index.php");

} **else** {

**$msg** = 'Sisteme kayıtlı kullanıcı bulunmadı!';

**$status** = 'alert-danger';

}

}

}

?>

Yukarıdaki kodları incelediğimizde öncelikle **bağlanti.php** isimli dosyamızı **giris.php**dosyamıza dahil ediyoruz.

Oturum nesnelerini oluşturmadan daha önce oturum nesnesi oluşturulmuş mu onu kontrol ediyoruz eğer oluşturulmuş ise direk **index.php**‘ye yönlendiriyoruz. Oluşturulmadı ise giriş işlemlerine devam ediyoruz.

Veritabanı bağlantımızı dahil ettikten sonra kullanıcıdan gelen kullanıcı adı ve şifre verilerini değişkenlere atayıp [else if karar yapısı](https://mustafaazak.com/php-if-ve-switch-karar-yapilarinin-kullanimi/) ile bir değere sahip olup olmadığını kontrol ediyoruz.

Değişkenlerimizin bir değere sahip olduklarından emin olduktan sonra bu değerleri veritabanımızdaki veriler ile karşılaştırıyoruz. Eğer veritabanı ile bir eşleşme sağlanırsa oturum nesnelerimizin tanımlamasını gerçekleştiriyoruz ve sayfamızı **index.php** isimli dosyamıza yönlendiriyoruz.

**Adım 5 : Oturum Yönetimi**

Php ile üyelik sistemi oluştururken en çok detayın olacağı aşama oturum yönetimidir. Bu aşamada üyenin sisteme giriş yaptıktan sonra hangi sayfalara erişebileceğinden, ne tür işlemler yapabileceğine kadar her türlü işlemi gerçekleştirebilirsiniz.

Giriş işleminden sonra oturum yönetimi için **index.php** isimli dosyamızı oluşturuyoruz.

Bu sayfada oturum açıldıktan sonra göstermek istediğimiz bilgileri gösterebiliriz.

<?php

**include** "baglanti.php";

**if** (!**$\_SESSION[**'login']) {

header('Location:giris.php');

}

**$kullanici\_getir** = **$db**->prepare("SELECT \* FROM uyeler WHERE uye\_id = ?");

**$kullanici\_getir**->execute(**array**(**$\_SESSION[**'uye\_id']));

**if** (**$kullanici\_getir**->rowCount() > 0) {

**$row** = **$kullanici\_getir**->fetch(PDO::FETCH\_OBJ);

}

**echo** 'Kullanıcı adınız : '. **$row**->uye\_kadi . '<br/>';

**echo** 'Eposta adresiniz : '. **$row**->uye\_eposta;

?>

<br/><a href="cikis.php">Çıkış Yap</a>

index.php dosyamızda yine ilk olarak daha önce oturum nesnesi oluşturulup oluşturulmadığını kontrol ediyoruz. Eğer oturum nesnesi oluşturulmadı ise giris.php dosyamıza yönlendiriyoruz.

Oturum açıldı ise uye\_id isimli oturum nesnemize atadığımız üye id değeri veritabanından sorgulayıp oturum açan kullanıcının tüm bilgilerine ulaşabiliriz. Burada sadece kullanıcı verileri olarak aklınıza gelmesin bu çok basit bir örnek olduğu için bakış açınızı daraltabilir ancak kullanıcıya giriş yaptırdıktan sonra aklınıza gelen her türlü işlemi yaptırabilirsiniz.

**Adım 6 : Kullanıcı Çıkış İşlemi**

Çıkış işlemi oturum nesnelerini tek tek yok ederek yada toplu bir şekilde yok ederek sağlanabilir. Bunun için öncelikle cikis.php isimli dosyamızı oluşturalım.

<?php

session\_start();

session\_destroy();

header('Location:giris.php');

?>

Çıkış işleminde tüm oturum nesneleri **session\_destroy**fonksiyonu ile yok ediyoruz daha sonra giris.php sayfamıza yönlendiriyoruz.

Oluşturduğumuz oturum nesnelerini tek tek yok etmek için aşağıdaki yolu izlediğimizde yine aynı sonucu elde edebiliriz.

<?php

session\_start();

unset(**$\_SESSION[**'login']);

unset(**$\_SESSION[**'uye\_id']);

header('Location:giris.php');

?>

Php ile üyelik sistemi mantığı kullanıcının oturum açmasından ve oturum kapatmasına kadar olan tüm süreç bu şekilde ilerlemektedir.