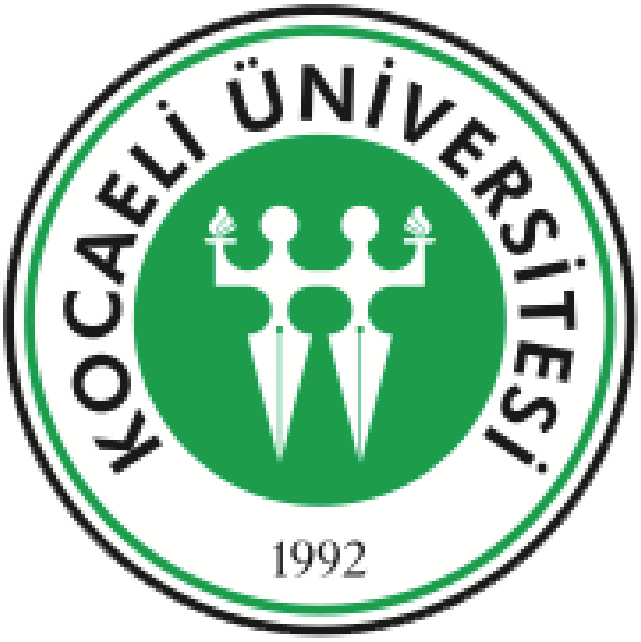


Fuar/Konferans Alanı Otomasyon Programı Projesi

Veritabanı Yönetim Sistemleri

Dr. Öğr. Üyesi Serdar SOLAK

Hazırlayanlar

Ahmet Barış Yardımcı - 221307078
Emrullah Donsak - 221307108
Emre Kardaş - 221307095

Proje Github Linki: https://github.com/brokolifha/veriTabaniProje

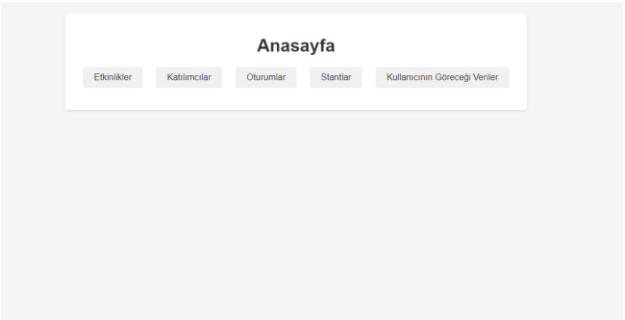
Fuar/Konferans Alanı Otomasyon Programı Projesi

Özet: Bu projede amaç oluşturulan web sitesinin yönetim paneli kısmına veri ekleme, silme, güncelleme işlemlerinin yapılabilmesidir. Sitenin kullanıcı paneli kısmında da bütün verileri değil de veritabanında oluşturulan viewler sayesinde belirli verileri görüntüleyebilmesi amaçlanmıştır. Proje PHP tabanlı bir projedir. Veritabanı işlemleri için MySQL kullanılmıştır.Projenin çalışabilmesi için bir Apache modülüne gerek duyulmaktadır. Bu nedenle "XAMPP" uygulamasının bilgisayarınızda kurulu olması gerekmektedir. Arayüz geliştirme ve etkileşim için JavaScript ve jQuery kullanılmıştır. Özellikle, tarih ve saat seçiminde JavaScript kütüphaneleri kullanılmıştır.

Kurulum: Projeye ait veritabanı dosyası, MySQL veritabanına aktarılarak proje çalışır hale getirilebilir. Proje dosyaları, XAMPP'ın kurulu olduğu dizindeki htdocs klasörüne kopyalanmalıdır. Daha sonra Apache sunucusu başlatılarak projeye "localhost:80/proje" adresinden erişilebilir hale getirilir.

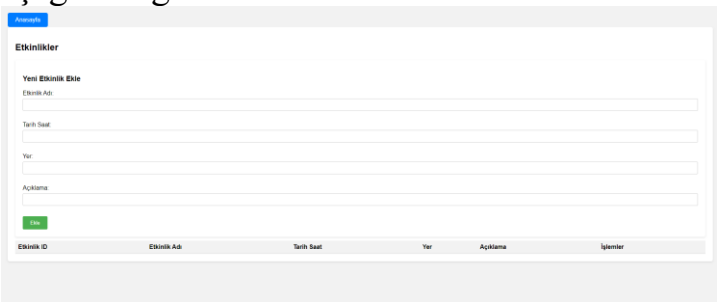
UYGULAMA

Proje açıldığı zaman karşımıza ilk olarak gelen ekran bir yönetici paneli olarak düşünülebilir.

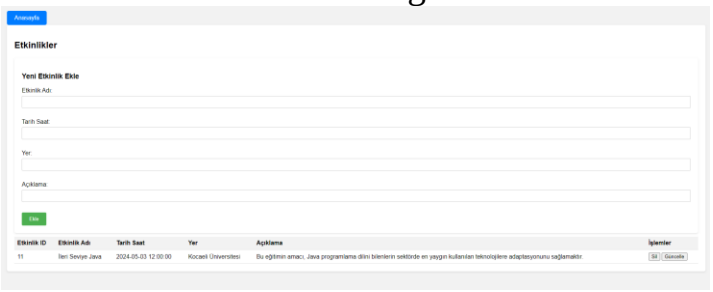


Bu ekranda hem yönetici hem de kullanıcı paneli, gerekli düzenlemeler ve sorgular ile oluşturulabilir.

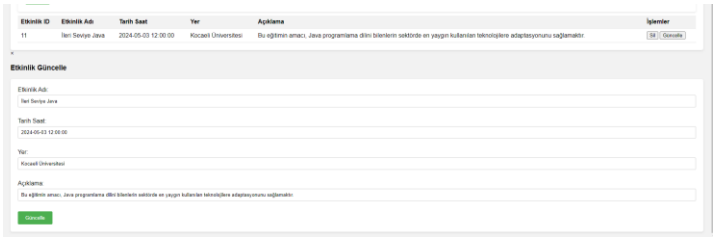
Bu panelde etkinlikler sayfasına tıklandığında karşımıza çıkacak ekran aşağıdaki gibidir.



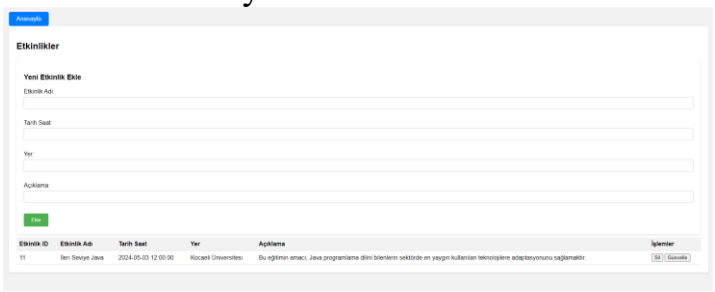
Aşağıdaki fotoğrafta da görüldüğü gibi bir etkinlik eklenmek istenirse, belirli inputlardan alınan veriler değişkenlere aktarılır ve veritabanına kaydedilir. Eklenen verilerin yanındaki butonlar sayesinde isteğe bağlı olarak veriler silinebilir ve güncellenebilir.



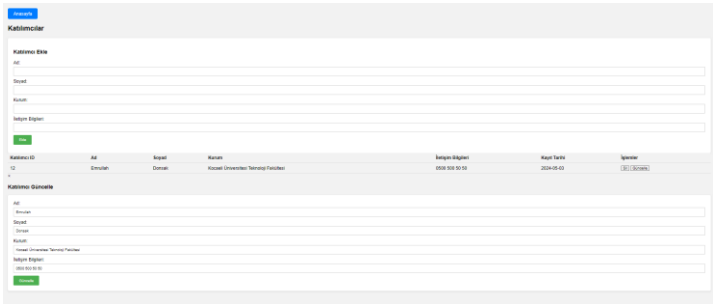
Aşağıdaki resimde görüldüğü gibi eklenen bir etkinlikte güncelleme yapılmak istenirse güncelle butonuna tıklanır. Güncelleme butonuna tıklandığında JavaScript tarafından oluşturulan bir Modal açılır. Bu modal, güncelleme butonuna tıklandığında yeni bir form oluşturur ve tüm veriler otomatik olarak formdaki textboxlara aktarılır, böylece üzerlerinde değişiklikler yapılabilir. Değişiklik yapıldıktan sonra güncelle butonuna tıklanılır. Daha sonra Modal'ı kapatmak için sol üstteki (x) işaretine tıklanarak Modal kapatılır.



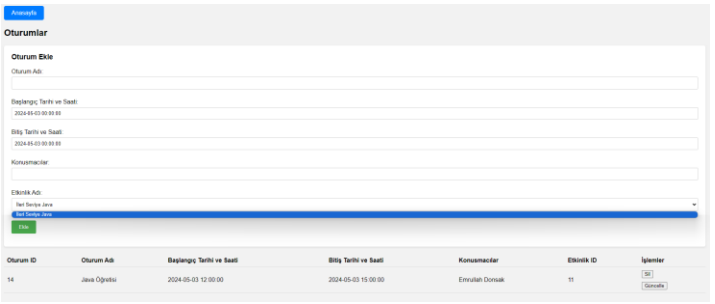
Tekrardan anasayfaya dönmek için sol üstte bulunan “Anasayfa” butonuna tıklanılır.



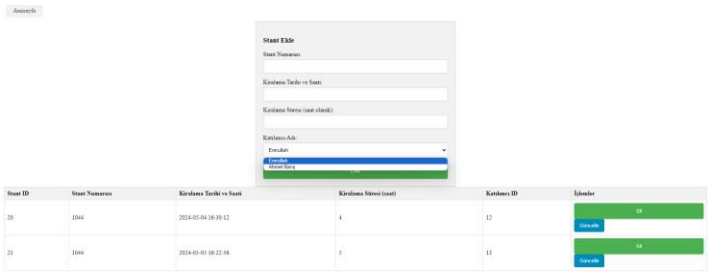
Anasayfa üzerinden "Katılımcılar" butonuna tıkladığımızda aşağıdaki resimde görüldüğü gibi benzer görünüme sahip bir sayfa daha açılır. Burada da aynı özellikler geçerlidir: ekleme, silme ve güncelleme işlemleri yapılabilir. Güncelleme butonuna tıklandığında bir güncelleme modalı açılır ve gerekli işlemler gerçekleştirilebilir.



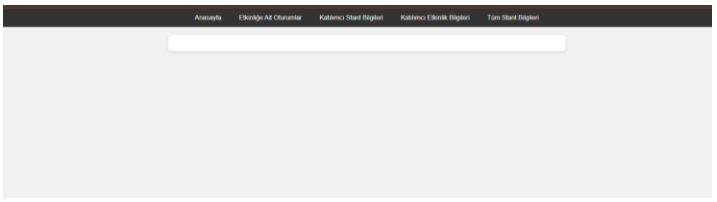
Tekrar anasayfa ekranına döndüğümüzde "Oturumlar" butonuna tıkladığımızda oturumlar.php sayfasına yönlendiririliriz. Burada diğer sayfalarda olduğu gibi aynı işlemleri yapabiliriz ancak buradaki fark, veritabanındaki Etkinlikler tablosundaki etkinlik adı verilerini bir select inputuyla ekranda gösterebilmemizdir. Bunun amacı, oturumlar ve etkinliklerin bağlantılı iki tablo olduğu için etkinlikleri ID numaralarına göre listeleyip oturumların hangi konularla ilişkili olduğunu seçip eklemeler yapabilmektir.

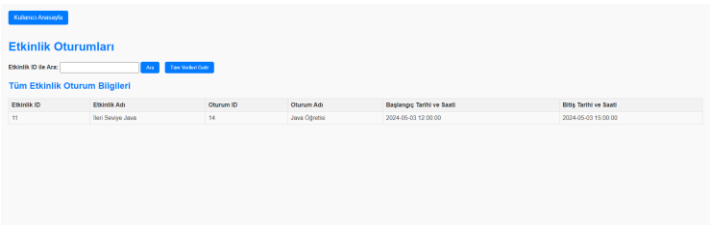


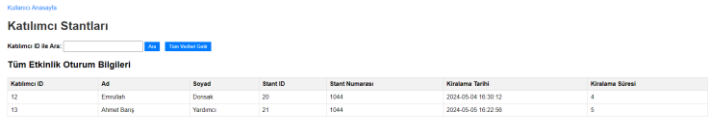
Anasayfaya geri dönüp "Stantlar" butonuna tıkladığımızda benzer bir ekranla karşılaşacağız. Burada da stantları hangi katılımcıların kullanacağı ve ne kadar süreyle kullanmak istediklerinin verileri tutulmaktadır. Katılımcı adı select inputunda yine katılımcılar tablosundan katılımcı ID verisine göre katılımcıların bilgileri çekilir ve buna göre eklemeler yapılabilir.

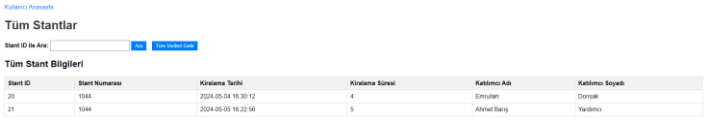


Anasayfaya geri döndükten sonra, sağ üst köşede bulunan "Kullanıcının Görebileceği Veriler" butonuna tıkladığımızda, veritabanında oluşturduğumuz viewleri görüntüleyebileceğimiz bir ekrana yönlendiririliriz. Bu ekran kullanıcı paneli olarak düşünülebilir. Burda herhangi bir güncelleme silme işlemi yapılamaz.









Projedeki Bazı Önemli Kod Parçalarına Yakından Bakış

PHP'de Veritabanı Bağlantısı Sağlayan Sınıf Oluşturma :

Aşağıdaki kod parçası, bir veritabanına bağlanmak için bir PHP sınıfı sağlar. Veritabanı bağlantısı için gerekli bilgileri içerir: sunucu adı, kullanıcı adı, parola ve veritabanı adı. Kod, bu bilgileri kullanarak PDO (PHP Data Objects) kullanarak bir veritabanı bağlantısı oluşturur. Bağlantı sırasında oluşabilecek hataları ele alır ve ekrana yazdırır. Bu sınıf, başka PHP dosyalarında kullanılmak üzere yeniden kullanılabilir bir veritabanı bağlantısı sağlar.

db.class.php

```php
<?php

class Db{
    private $hots = "localhost";
    private $user = "root";
    private $password = "[redacted]";
    private $dbName = "veritabani";


    protected function connect(){
        try{
            $dsn = "mysql:host=". $this->hots.";dbname=". $this->dbName;
            $pdo = new PDO($dsn,$this->user,$this->password);
            $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
            $pdo->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE,PDO::FETCH_OBJ);

            return $pdo;

        }catch(PDOException $e){
            echo "Bağlantı Hatası". $e->getMessage();

        }
    }

}


?>
```

Otomasyon Sistemi İçin Veritabanı CRUD İşlemlerini Yapan PHP Sınıfı:

Aşağıdaki kod parçası, bir otomasyon sistemine ait olan farklı tablolar üzerinde CRUD (Create, Read, Update, Delete) işlemlerini gerçekleştirmek için bir sınıf içerir. Kod, önceden tanımlanan Db sınıfını genişletir ve bu sınıf aracılığıyla veritabanına bağlanır. Sınıf içindeki yöntemler, farklı tablolardan veri çekmek için kullanılır. Örneğin, listEtkinlikler() yöntemi etkinlikler tablosundan verileri çeker, listKatilimcilar() yöntemi katılımcılar tablosundan verileri çeker ve benzer şekilde diğer yöntemler de ilgili tablolardan verileri çeker.

tablesCrud.class.php

```php
<?php
//Otomasyonda tablolarda ki crud işlemler için kullanılacak class

class tableCrud extends Db{

    //Ekrana yazdırma fonksiyonları:----------

    public function listEtkinlikler(){
        $sql = "SELECT * FROM etkinlikler";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }

    public function listKatilimcilar(){
        $sql = "SELECT * FROM katilimcilar";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }

    public function listKatilimcilarOturumlar(){
        $sql = "SELECT * FROM katilimcilar_oturumlar";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }


    public function listOturumlar(){
        $sql = "SELECT * FROM oturumlar";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }

    public function listGeçmisOturumlar(){
        $sql = "SELECT * FROM silinenoturumlar";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }
    public function liststantlar(){
        $sql = "SELECT * FROM stantlar";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }

//----------
```

Belirli ID'ye Göre Veri Almak İçin PHP Sınıfı

Aşağıdaki kod parçası, farklı tablolardan belirli bir ID'ye göre veri almak için kullanılır. Her bir yöntem, belirli bir tablodan belirli bir ID'ye sahip olan bir öğeyi seçer ve döndürür. Örneğin, listEtkinliklerById() yöntemi, etkinlikler tablosundan belirli bir etkinlik ID'sine sahip etkinliği seçer ve döndürür. Benzer şekilde, diğer yöntemler de ilgili tablolardan belirli bir ID'ye sahip öğeleri seçer ve döndürür.

```php
//Id ye göre ekrana yazdırma kodları



public function listEtkinliklerById(int $id){
    $sql = "SELECT * FROM etkinlikler WHERE EtkinlikID = :id";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute(['id' => $id]);
        return $stmt->fetch();
}

public function listKatilimcilarById(int $id){
    $sql = "SELECT * FROM katilimcilar WHERE KatilimciID = :id";
    $stmt = $this->connect()->prepare($sql);
    $stmt->execute(['id' => $id]);
    return $stmt->fetch();

}


public function listOturumlarById(int $id){
    $sql = "SELECT * FROM oturumlar WHERE OturumID = :id";
    $stmt = $this->connect()->prepare($sql);
    $stmt->execute(['id' => $id]);
    return $stmt->fetch();

}

public function listSilinenOturumlarById(int $id){
    $sql = "SELECT * FROM silinenoturumlar WHERE SilinenOturumID = :id";
    $stmt = $this->connect()->prepare($sql);
    $stmt->execute(['id' => $id]);
    return $stmt->fetch();
}



public function listStantlarById(int $id){
    $sql = "SELECT * FROM stantlar WHERE StantID = :id";
    $stmt = $this->connect()->prepare($sql);
    $stmt->execute(['id' => $id]);
    return $stmt->fetch();

}

//-------------------------------------------------------------------------------
```

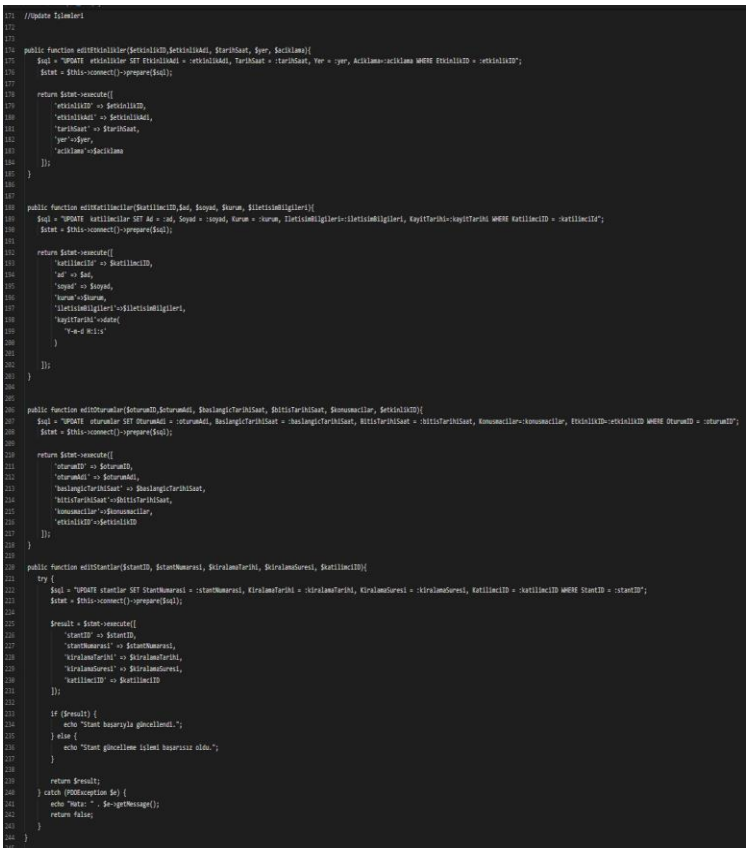
PHP ile Veritabanına Yeni Veri Ekleme İşlemleri

Aşağıdaki kod parçası, veritabanına yeni veri eklemek için kullanılan PHP işlevlerini içerir. Her bir yöntem, belirli bir tabloya yeni bir kayıt eklemek için kullanılır.Örneğin, createEtkinlik() yöntemi, etkinlikler tablosuna yeni bir etkinlik eklemek için kullanılır. Benzer şekilde, diğer yöntemler de ilgili tablolara yeni kayıtlar eklemek için kullanılır.

```php
//veri Ekleme işlemleri

public function createetkinlik($etkinlikAdi, $tarihSaat, $yer, $aciklama){
    $sql = "INSERT INTO etkinlikler (EtkinlikAdi, TarihSaat, Yer, Aciklama) VALUES (:etkinlikAdi,:tarihSaat,:yer, :aciklama)";
    $stmt = $this->connect()->prepare($sql);

    return $stmt->execute([

        'etkinlikAdi' => $etkinlikAdi,
        'tarihSaat' => $tarihSaat,
        'yer'=>$yer,
        'aciklama'=>$aciklama
    ]);
}


public function createkatilimci($ad, $soyad, $kurum, $iletisimBilgileri){
    $sql = "INSERT INTO katilimcilar (Ad, Soyad, Kurum, IletisimBilgileri,KayitTarihi) VALUES (:ad,:soyad,:kurum, :iletisim,:kayitTarihi)";
    $stmt = $this->connect()->prepare($sql);

    return $stmt->execute([

        'ad' => $ad,
        'soyad' => $soyad,
        'kurum'=>$kurum,
        'iletisim'=>$iletisimBilgileri,
        'kayitTarihi'=>date(
            'Y-m-d H:i:s'
        )
    ]);
}

public function createOturumlar($oturumAdi, $baslangicTarihiSaat, $bitisTarihiSaat, $konusmacilar,$etkinlikID){
    $sql = "INSERT INTO oturumlar (OturumAdi, BaslangicTarihiSaat, BitisTarihiSaat, konusmacilar,EtkinlikID) VALUES (:oturumAd,:baslangicTarihSaat,:bitisSaat, :konusmacilar,:etkinlikId)";
    $stmt = $this->connect()->prepare($sql);

    return $stmt->execute([

        'oturumAd' => $oturumAdi,
        'baslangicTarihSaat' => $baslangicTarihiSaat,
        'bitisSaat'=>$bitisTarihiSaat,
        'konusmacilar'=>$konusmacilar,
        'etkinlikId'=>$etkinlikID
    ]);
}

public function createStant($stantNumarasi, $kiralamaTarihi, $kiralamaSuresi, $katilimciID){
    $sql = "INSERT INTO stantlar (StantNumarasi, KiralamaTarihi, KiralamaSuresi, KatilimciID) VALUES (:stantNumarasi, :kiralamaTarihi, :kiralamaSuresi, :katilimciID)";
    $stmt = $this->connect()->prepare($sql);

    return $stmt->execute([
        'stantNumarasi' => $stantNumarasi,
        'kiralamaTarihi' => $kiralamaTarihi,
        'kiralamaSuresi' => $kiralamaSuresi,
        'katilimciID' => $katilimciID
    ]);
}


//Insert işlemleri bitti
```

PHP ile Veritabanındaki Verileri Güncelleme İşlemleri

Aşağıdaki kod parçası, veritabanındaki mevcut verilerin güncellenmesi için kullanılan PHP işlevlerini içerir. Her bir yöntem, belirli bir tablodaki belirli bir kaydı güncellemek için kullanılır.Örneğin, editEtkinlikler() yöntemi, etkinlikler tablosundaki belirli bir etkinliği güncellemek için kullanılır. Benzer şekilde, diğer yöntemler de ilgili tablolardaki belirli kayıtları güncellemek için kullanılır.



```
//Update İşlemleri


public function editEtkinlikler($etkinlikID,$etkinlikAdi, $tarihSaat, $yer, $aciklama){
    $sql = "UPDATE  etkinlikler SET EtkinlikAdi = :etkinlikAdi, TarihSaat = :tarihSaat, Yer = :yer, Aciklama=:aciklama WHERE EtkinlikID = :etkinlikID";
     $stmt = $this->connect()->prepare($sql);

    return $stmt->execute([
        'etkinlikID' => $etkinlikID,
        'etkinlikAdi' => $etkinlikAdi,
        'tarihSaat' => $tarihSaat,
        'yer'=>$yer,
        'aciklama'=>$aciklama
    ]);
  }


public function editKatilimcilar($katilimciID,$ad, $soyad, $kurum, $iletisimBilgileri){
    $sql = "UPDATE  katilimcilar SET Ad = :ad, Soyad = :soyad, Kurum = :kurum, IletisimBilgileri=:iletisimBilgileri, KayitTarihi=:kayitTarihi WHERE KatilimciID = :katilimciId";
     $stmt = $this->connect()->prepare($sql);

    return $stmt->execute([
        'katilimciId' => $katilimciID,
        'ad' => $ad,
        'soyad' => $soyad,
        'kurum'=>$kurum,
        'iletisimBilgileri'=>$iletisimBilgileri,
        'kayitTarihi'=>date(
            'Y-m-d H:i:s'
        )

    ]);
  }


public function editOturumlar($oturumID,$oturumAdi, $baslangicTarihiSaat, $bitisTarihiSaat, $konusmacilar, $etkinlikID){
    $sql = "UPDATE  oturumlar SET OturumAdi = :oturumAdi, BaslangicTarihiSaat = :baslangicTarihiSaat, BitisTarihiSaat = :bitisTarihiSaat, Konusmacilar=:konusmacilar, EtkinlikID=:etkinlikID WHERE OturumID = :oturumID";
     $stmt = $this->connect()->prepare($sql);

    return $stmt->execute([
        'oturumID' => $oturumID,
        'oturumAdi' => $oturumAdi,
        'baslangicTarihiSaat' => $baslangicTarihiSaat,
        'bitisTarihiSaat'=>$bitisTarihiSaat,
        'konusmacilar'=>$konusmacilar,
        'etkinlikID'=>$etkinlikID
    ]);
  }
public function editStantlar($stantID, $stantNumarasi, $kiralamaTarihi, $kiralamaSuresi, $katilimciID){
    try {
        $sql = "UPDATE stantlar SET StantNumarasi = :stantNumarasi, KiralamaTarihi = :kiralamaTarihi, KiralamaSuresi = :kiralamaSuresi, KatilimciID = :katilimciID WHERE StantID = :stantID";
        $stmt = $this->connect()->prepare($sql);

        $result = $stmt->execute([
            'stantID' => $stantID,
            'stantNumarasi' => $stantNumarasi,
            'kiralamaTarihi' => $kiralamaTarihi,
            'kiralamaSuresi' => $kiralamaSuresi,
            'katilimciID' => $katilimciID
        ]);

        if ($result) {
            echo "Stant başarıyla güncellendi.";
        } else {
            echo "Stant güncelleme işlemi başarısız oldu.";
        }

        return $result;
    } catch (PDOException $e) {
        echo "Hata: " . $e->getMessage();
        return false;
    }
}
```

PHP ile Veritabanındaki Verileri Silme İşlemleri

Aşağıdaki kod parçası, veritabanından belirli verilerin silinmesi için kullanılan PHP işlevlerini içerir. Her bir yöntem, belirli bir tablodaki belirli bir kaydı silmek için kullanılır. Örneğin, deleteEtkinlikler() yöntemi, etkinlikler tablosundaki belirli bir etkinliği silmek için kullanılır. Benzer şekilde, diğer yöntemler de ilgili tablolardaki belirli kayıtları silmek için kullanılır. Bazı yöntemlerde, örneğin deleteOturumlar() yönteminde, bir oturumu silerken ilgili etkinliği de silmek için bir işlem gerçekleştirilir. Bu şekilde ilişkili verilerin silinmesi sağlanır.

```
//Delete işlemleri


public function deleteEtkinlikler($etkinlikID){
    $sql = "DELETE FROM etkinlikler WHERE EtkinlikID=:etkinlikId";
    $stmt = $this->connect()->prepare($sql);
    return $stmt->execute([
        'etkinlikId'=> $etkinlikID
    ]);


}


public function deleteKatilimcilar($katilimciID){
    $sql = "DELETE FROM katilimcilar WHERE KatilimciID=:katilimciID";
    $stmt = $this->connect()->prepare($sql);
    return $stmt->execute([
        'katilimciID'=> $katilimciID
    ]);

}
public function deleteOturumlar($oturumID){
    $pdo = $this->connect();
    try {
        $pdo->beginTransaction();

        // Oturumu sil
        $sqlDeleteOturum = "DELETE FROM oturumlar WHERE OturumID = :oturumID";
        $stmtDeleteOturum = $pdo->prepare($sqlDeleteOturum);
        $stmtDeleteOturum->execute(['oturumID' => $oturumID]);

        // İlgili etkinliği sil
        $sqlDeleteEtkinlik = "DELETE FROM etkinlikler WHERE EtkinlikID = (
            SELECT EtkinlikID FROM oturumlar WHERE OturumID = :oturumID
        )";
        $stmtDeleteEtkinlik = $pdo->prepare($sqlDeleteEtkinlik);
        $stmtDeleteEtkinlik->execute(['oturumID' => $oturumID]);

        $pdo->commit();
        return true;
    } catch (PDOException $e) {
        $pdo->rollBack();
        // Hata mesajını ekrana yazdır
        echo "Silme Hatası: " . $e->getMessage();
        return false;
    }
}




public function deleteStantlar($stantID) {
    try {
        $sql = "DELETE FROM stantlar WHERE StantID = :stantID";
        $stmt = $this->connect()->prepare($sql);
        $stmt->bindParam(':stantID', $stantID, PDO::PARAM_INT);
        $result = $stmt->execute();

        if ($result) {
            echo "Stant başarıyla silindi.";
        } else {
            echo "Stant silme işlemi başarısız oldu.";
        }

        return $result;
    } catch (PDOException $e) {
        echo "Hata: " . $e->getMessage();
        return false;
    }
}
```

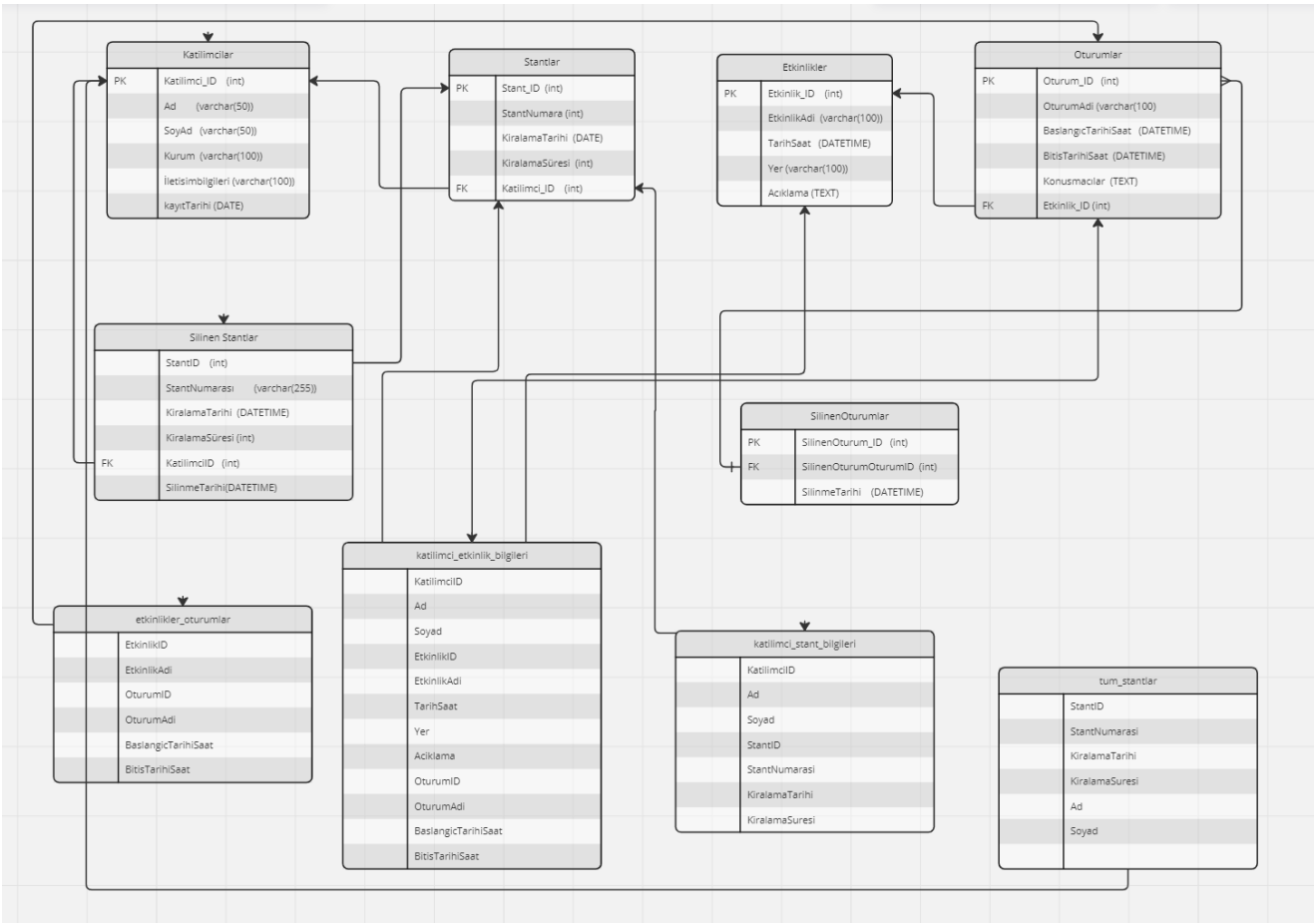
Kullanıcı Arayüzü İçin Veritabanı Görünümleri Erişimi Sağlayan PHP Sınıfı

Aşağıdaki kod parçaları, kullanıcı arayüzünde görüntülemek için önceden tanımlanmış olan veritabanı görünümlerine erişim sağlar. Her bir yöntem, belirli bir görünümü almak veya belirli bir koşula göre arama yapmak için kullanılır. Örneğin, listEtkinlikOturumBilgisi() yöntemi, etkinlikler ve oturumlar hakkında bilgiler içeren bir görünümün tamamını getirir. Benzer şekilde, diğer yöntemler de ilgili görünümleri getirir veya belirli bir koşula göre arama yapar.

```
userView.class.php

<?php

//Bu classı oluşturmamızın amacı dabasede kullandığımız viewlerin kullanıcılara gösterimini belirtmektir.
//Bir geliştirici gibi değil daha sadece bir şekilde kullanıcı verileri görüp ona göre işlem yapabilmektedir.


//Burada sadece viewler yani kullanıcılar herhangi bir arama yapacağı zaman sadeleştirilmiş şekilde gösterilecektir.



class UserView extends Db{



    public function listEtkinlikOturumBilgisi(){
        $sql = "SELECT * FROM etkinlikler_oturumlar";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }
    public function listEtkinlikOturumBilgisiById(int $id){
        $sql = "SELECT * FROM etkinlikler_oturumlar WHERE EtkinlikID=:id";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute(['id'=>$id]);
        return $stmt->fetch();
    }


    public function searchEtkinlikOturumBilgisi($search){
        $sql = "SELECT * FROM etkinlikler_oturumlar WHERE
                    EtkinlikID LIKE :search";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute(['search' => "%$search%"]);
        return $stmt->fetchAll();
    }




//-----------------------------------------------------------------------------------------------------------


    public function listKatilimciEtkinlikBilgisi(){
        $sql = "SELECT * FROM katilimci_stant_bilgileri";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();
    }

public function listKatilimciEtkinlikBilgisiById(int $id){

    $sql = "SELECT * FROM katilimci_stant_bilgileri WHERE KatilimciID=:id";
    $stmt = $this->connect()->prepare($sql);
    $stmt->execute(['id'=>$id]);
    return $stmt->fetch();
}
public function searchKatilimciEtkinlikBilgisi($search){
    $sql = "SELECT * FROM katilimci_stant_bilgileri WHERE
                KatilimciID LIKE :search";
    $stmt = $this->connect()->prepare($sql);
    $stmt->execute(['search' => "%$search%"]);
    return $stmt->fetchAll();
}




//-----------------------------------------------------------------------------------------------------------
```

```
    public function listKatilimciOturumBilgisi(){
        $sql = "SELECT * FROM katilimci_etkinlik_bilgileri";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();
    }

    public function listKatilimciOturumBilgisiById(int $id){
        $sql = "SELECT * FROM katilimci_etkinlik_bilgileri WHERE KatilimciID=:id ";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute(['id'=>$id]);
        return $stmt->fetch();
    }

    public function searchKatilimciOturumBilgisiById($id){
        $sql = "SELECT * FROM katilimci_etkinlik_bilgileri WHERE KatilimciID=:id";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute(['id' => $id]);
        return $stmt->fetchAll();
    }



//-----------------------------------------------------------------------------------------------------------




    public function listStantKatilimciBilgisi(){
        $sql = "SELECT * FROM tum_stantlar";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute();
        return $stmt->fetchAll();

    }

    public function listStantKatilimciBilgisiById(int $id){
        $sql = "SELECT * FROM tum_stantlar WHERE StantID=:id";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute(['id'=>$id]);
        return $stmt->fetch();

    }

    public function searchStantKatilimciBilgisi($id){
        $sql = "SELECT * FROM tum_stantlar WHERE StantID=:id";
        $stmt = $this->connect()->prepare($sql);
        $stmt->execute(['id' => $id]);
        return $stmt->fetchAll();
    }



}


?>
```

VERİTABANI ER DİYAGRAMI



KAYNAKÇA:

https://www.w3schools.com/
https://www.npmjs.com/
https://getbootstrap.com/
https://laravel.com/
https://phpturkiye.net/
https://www.mysql.com/