**PROJE RAPORU:** WEB UYGULAMASI GELİŞTİRME

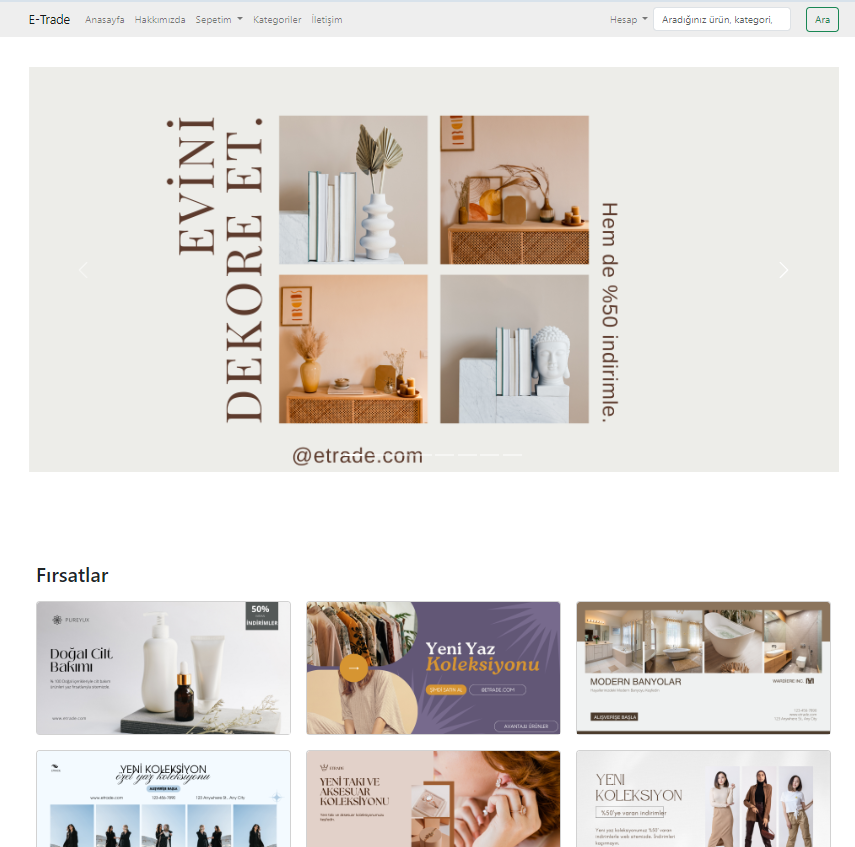
**PROJE ADI:** E-TRADE APP

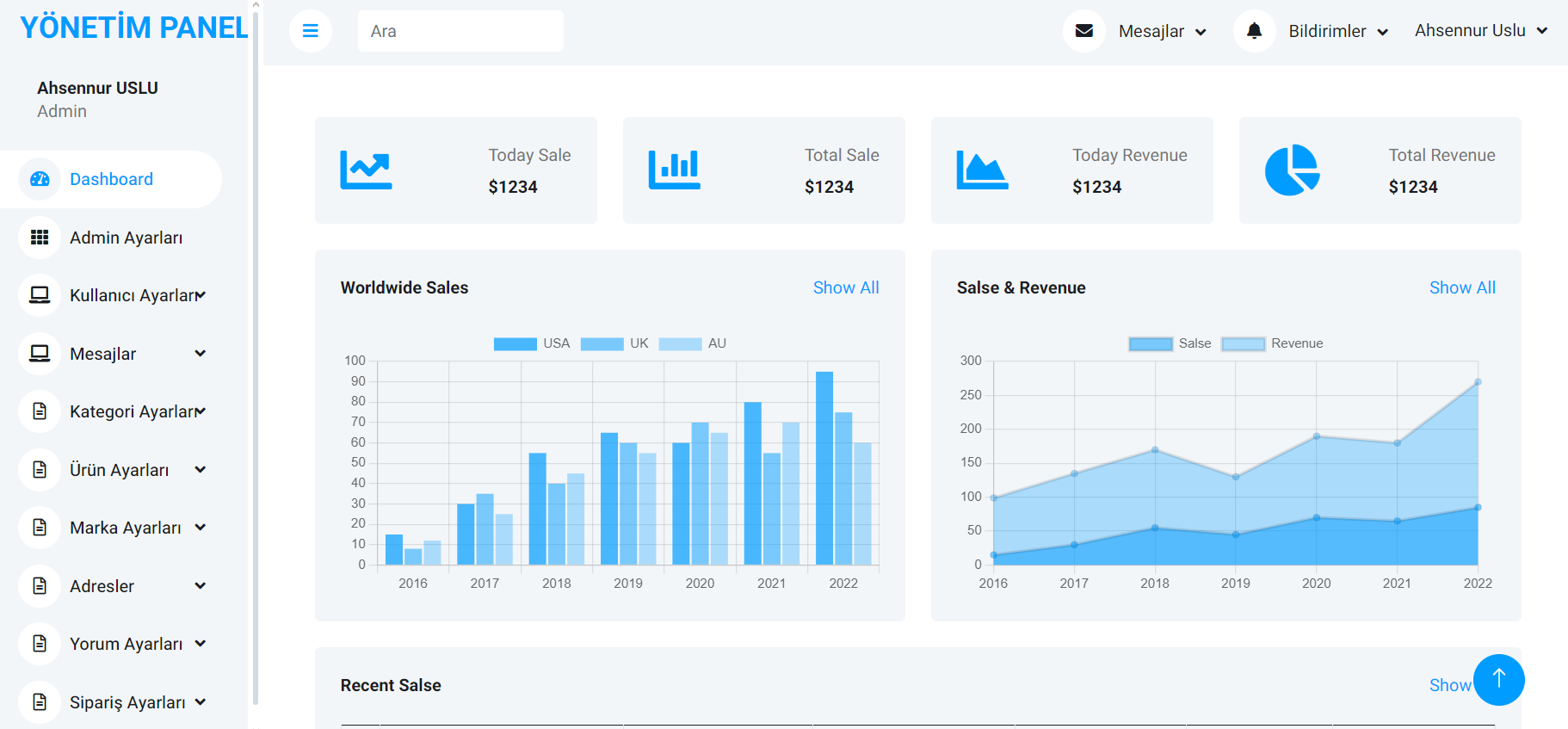
**GELİŞTİRİCİ:** AHSENNUR USLU

**TARİH:** 07/06/2024

**1.GİRİŞ**

Bu rapor E-Trade App web uygulamasının geliştirilme sürecini, kullanılan teknolojileri, uygulanan özellikleri ve karşılaşılan zorlukları kapsamaktadır. Bu web uygulaması kullanıcıların e-ticaret hizmetlerine kolayca erişebilecekleri ve yönetebilecekleri bir platform sağlamaktadır. Uygulama genel olarak iki farklı arayüzden oluşmaktadır. Bunlar Yönetici kullanıcının siteyi daha kolay idare etmesini sağlayan Admin Panel arayüzü, diğer arayüz de siteyi kullanan kullanıcıların etkileşimde bulunduğu sitenin kendi arayüzüdür.





**2. PROJE TANIMI**

E-Trade App PHP, MYSQL, JSON kullanılarak geliştirilmiş bir web uygulamasıdır. Uygulama kullanıcının kayıt olabileceği, kayıt işleminin ardından uygulamaya giriş yapabileceği ardından profil sayfasından kişisel bilgilerini, fotoğrafını ve adreslerini ekleyip düzenleme ve silme işlemlerini gerçekleştirdiği , verilerini yönetebileceği, belirli kategorilere göre listelenen ürünlerden dilediğini sepete ekleyip toplam tutar, miktar ve ürünleri listeleyebileceği, karşılaştığı sorunlarda iletişim sayfasında bulunan iletişim formunu doldurarak yönetici ile iletişime geçebileceği ve çeşitli işlemler gerçekleştirebileceği bir platform sunmaktadır. Admin arayüzünde de yönetici olan kullanıcının kayıtlı olan kullanıcıların listesine ulaşabildiği Kullanıcı Listesi sayfası, kullanıcı mesajlarını görebildiği Mesaj Kutusu sayfası, Kategori ekleme bu kategorilerin listelendiği ve çeşitli ayarlamalarla silme işlemlerinin yapılabildiği sayfalar, Kategoriye göre ürün eklenen, listelenen ve değişikliklerin yapılıp silme işlemlerinin yapıldığı sayfalar, Kullanıcı adreslerinin kullanıcı ID’sine göre tutulduğu adres listeleme sayfaları bulunmaktadır.

**3. KULLANILAN TEKNOLOJİLER**

PHP: Sunucu tarafında çalışan betik dili.

MySQL: Veritabanı yönetim sistemi.

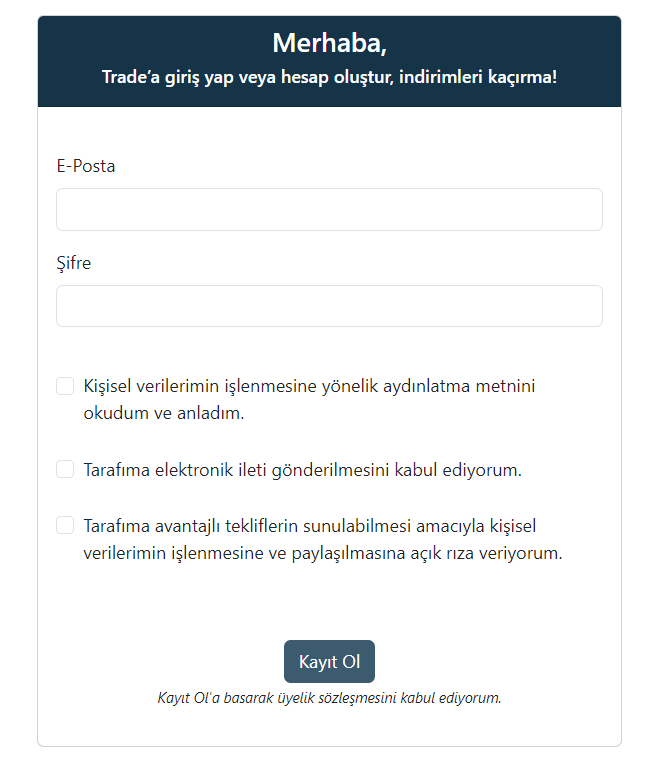
JSON: Veri değişim formatı.

**4. ÖZELLİKLER**

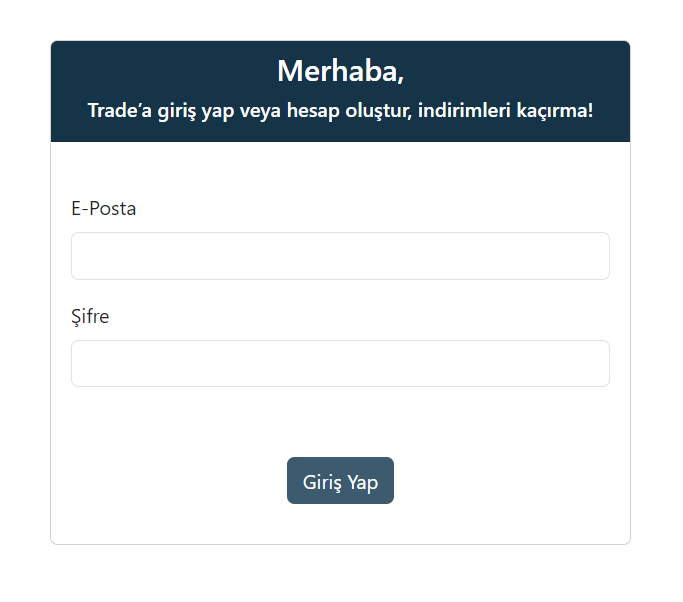
**4.1 KULLANICI YÖNETİMİ**

4.1.1. KULLANICI KAYIT VE GİRİŞ SİSTEMİ

Kullanıcılar sisteme kaydolmak için Navbarda bulunan “Hesap” dropdown menüsü içerisindeki “Kayıt Ol” butonuna tıklayarak yönlendirildikleri sayfada E-Posta ve Şifre bilgileri ile sisteme kayıt olabilmektedirler.

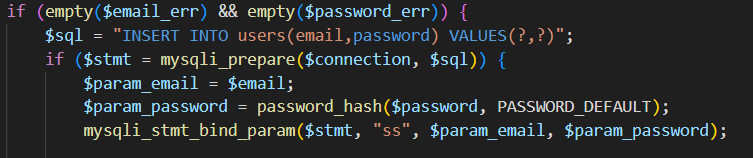


Kayıt olma işleminin ardından kullanıcılar Giriş yapma sayfasına yönlendirilerek sisteme giriş yapmaları sağlanmaktadır.



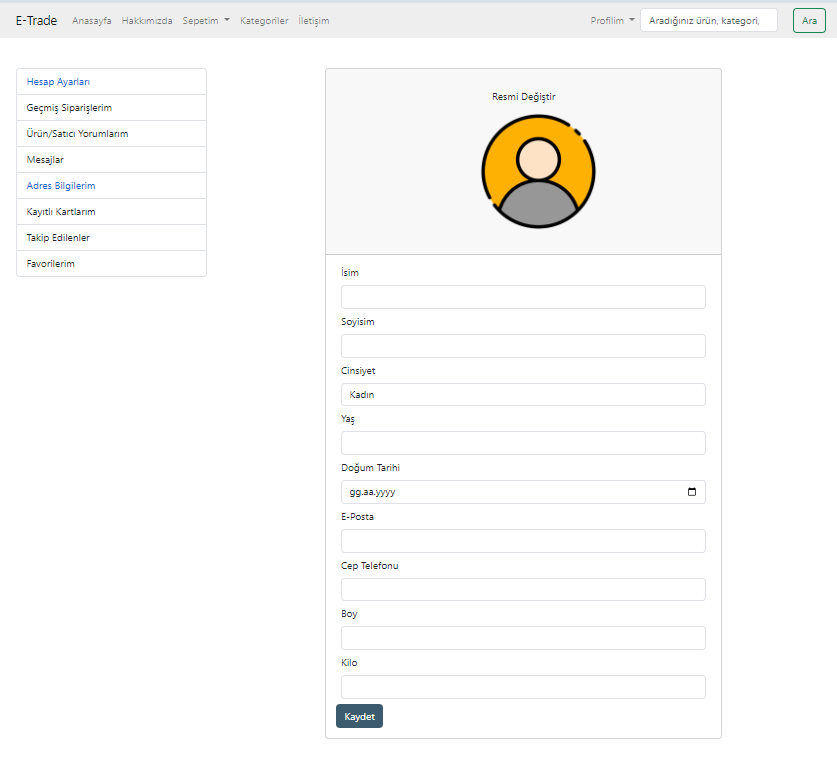
4.1.2. ŞİFRELERİN GÜVENLİ BİR ŞEKİLDE SAKLANMASI (HASHING)

Kullanıcı kaydı yapılırken kullanıcının girdiği şifreler password\_hash() fonksiyonu ile veri tabanına hashlenerek kaydedilmektedir.

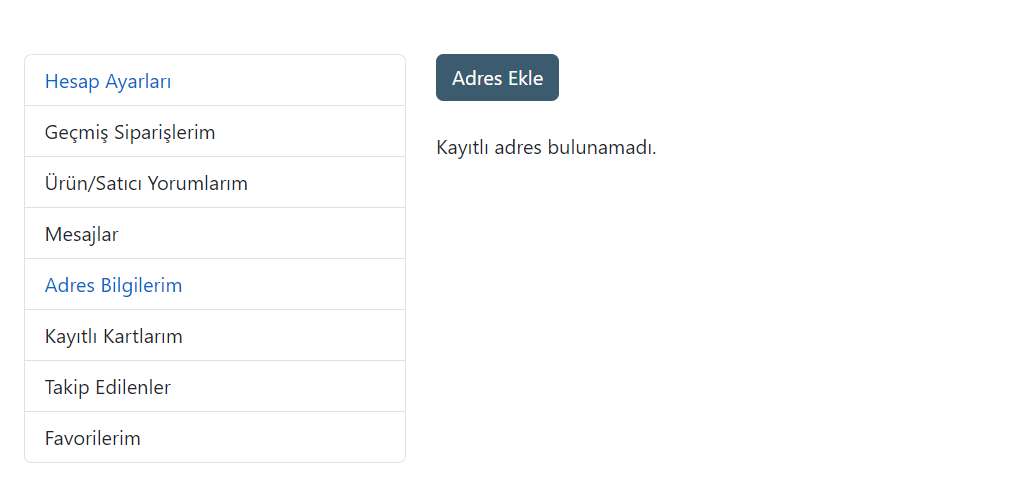


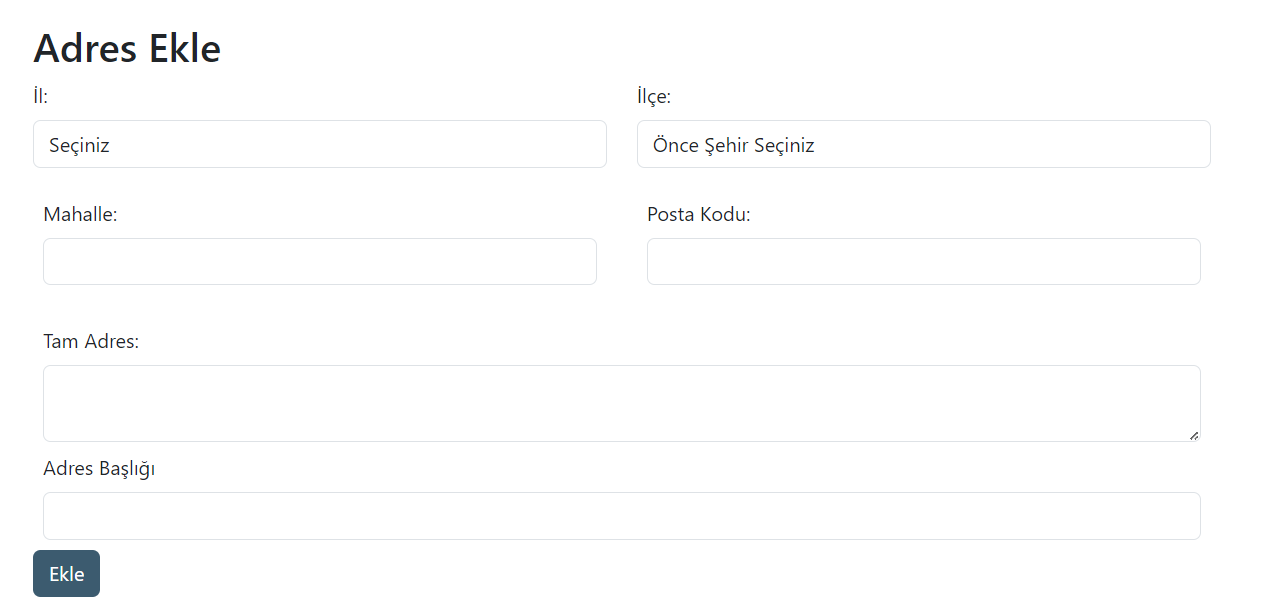
4.1.3. KULLANICI PROFİLİ VE HESAP YÖNETİMİ

Kullanıcı giriş işleminde başarılı olduğunda kullanıcı profil sayfasına yönlendirilir. Bu sayfada kullanıcının doldurabileceği bazı alanlar bulunmaktadır. Dilerse bu alandaki kişisel bilgilerini doldurup fotoğraf yükleyebilir. Buradaki bilgileri doldurup kaydettiğinde yine dilerse bu sayfadan güncelleme işlemlerini de yapabilir.

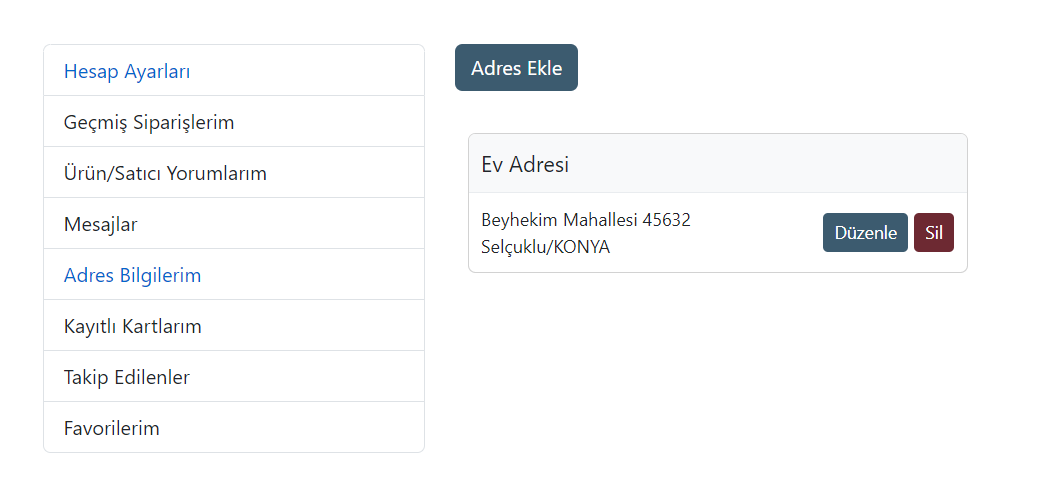


Profil bilgilerinin dışında Adres bilgileri de eklenebilir. Bunun için Sidebarda bulunan “Adres Bilgilerim” kısmına tıklayıp bu sayfadaki “Adres Ekle” butonuna tıklaması yeterlidir.





Burada öncelikle şehir seçimini yapması gerekmektedir. Bu işlemi ardından bu şehre ait İlçe bilgilerini yandaki “İlçe” kısmına tıklayarak seçebilir. Diğer alanları da doldurduktan sonra Ekle butonuna tıklayarak Adres ekleme işlemini gerçekleştirebilir. Ekleme işlemi başarılı olduğunda kullanıcının bir önceki sayfada eklenen adresi görebilmesi için yönlendirme işlemi yapılır.



**4.2 VERİTABANI İŞLEMLERİ**

4.2.1. CRUD (CREATE, READ, UPDATE, DELETE) İŞLEMLERİ

Uygulamanın birçok kısmında CRUD işlemleri kullanılmıştır.

1. **Ürünler**: Ürün ekleme, güncelleme ve silme işlemleri.

2. **Kullanıcılar**: Kullanıcı kayıt, profil güncelleme ve silme işlemleri.

3. **Siparişler**: Sipariş oluşturma, sipariş durumunu güncelleme ve silme işlemleri.

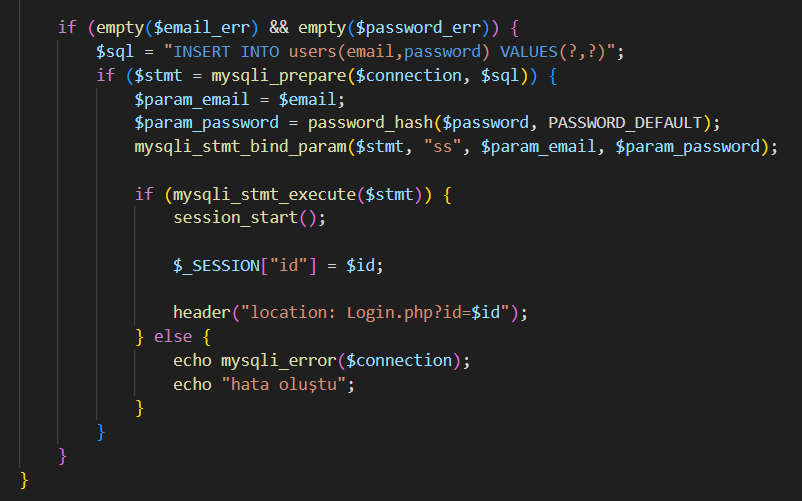
4. **Mesajlar**: Mesaj gönderme, okuma, güncelleme ve silme işlemleri.

Kullanıcının kaydolma, giriş ve profil işlemleri üzerinden bir örnek verilmiştir.

CREATE

**Örnek:** Kullanıcı kayıt işlemi.

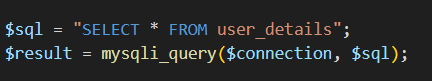
Kullanıcı kayıt formunda yeni bir kullanıcı oluşturma işlemi yapılır.



READ

**Örnek:** Kullanıcı giriş işlemi.

Giriş yapan kullanıcının bilgilerini veritabanından okuma işlemi yapılır.

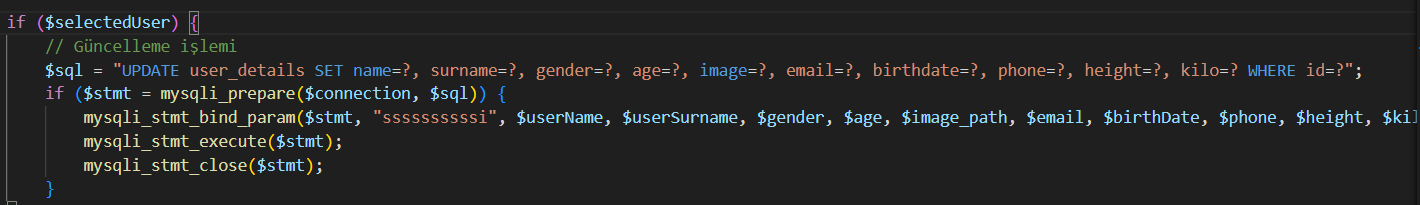




UPDATE

**Örnek:** Kullanıcı profilini güncelleme.

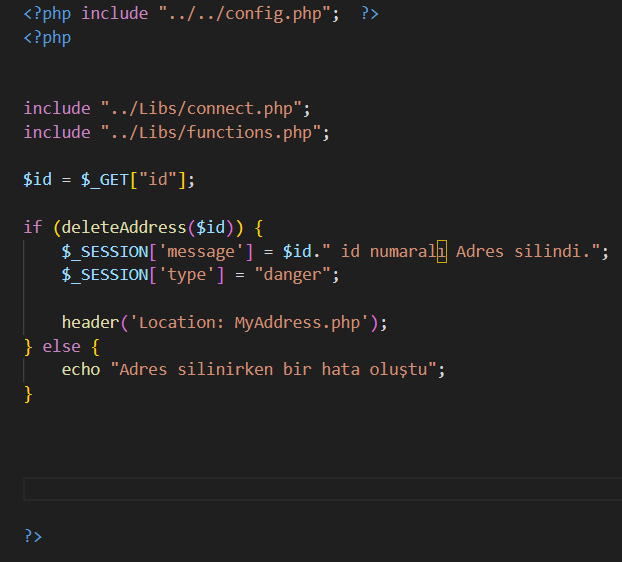
Kullanıcının profil bilgilerini güncelleme işlemi yapılır.



DELETE

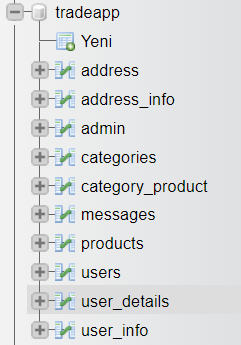
**Örnek:** Kullanıcı adres bilgisini silme.

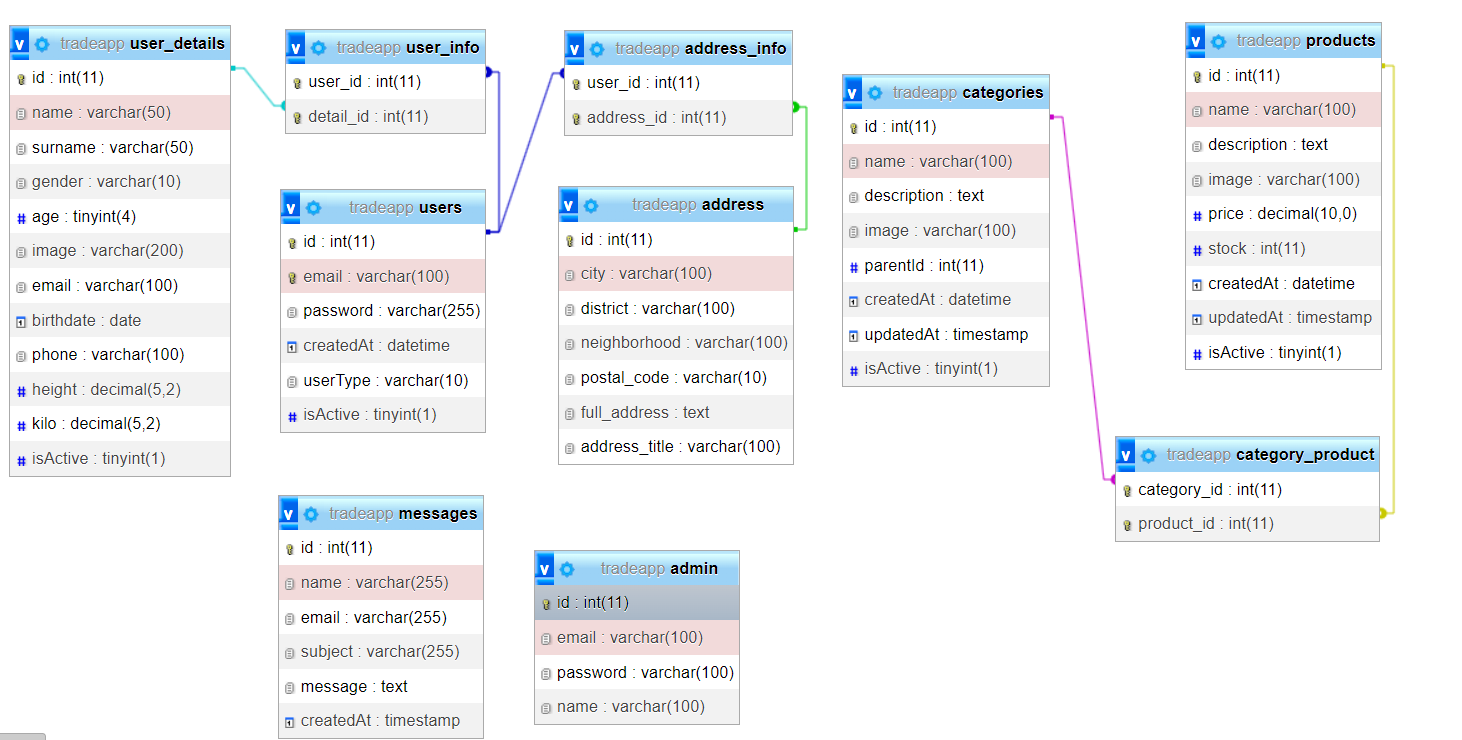
Kullanıcının adres bilgisini veritabanından silinmesi işlemi.



4.2.2. VERİTABANI TASARIMI VE İLİŞKİSEL VERİTABANI YÖNETİMİ

Uygulamada genel olarak Kullanıcıların tutulduğu users tablosu, kullanıcı detay bilgilerinin tutulduğu user\_details tablosu bu iki tablonun ilişkisel olarak id değerlerinin tutulduğu user\_info tablosu , adreslerin tutulduğu address tablosu yine aynı şekilde adres bilgileri ile kullanıcı bilgilerinin ilişkisel olarak tutulduğu address\_info tablosu, kategorilerin tutulduğu categories tablosu, ürünlerin tutulduğu products tablosu , ürünler ile kategorilerin ilişkisel olarak tutulduğu category\_product tablosu , mesajların tutulduğu messages tablosu ve son olarak admin bilgilerinin tutulduğu admin tablosu bulunmaktadır.





4.2.3. MYSQL KULLANARAK VERİTABANI İŞLEMLERİNİN GERÇEKLEŞTİRİLMESİ

Uygulamada MYSQL kullanılarak veritabanı işlemleri gerçekleştirilmiştir.

* **Yeni bir kullanıcının kaydedilmesi işlemi MySQL veritabanı kullanılarak yapılır. Bu işlem INSERT sorgusu kullanılır.**

*Örneğin:*

*$sql = "INSERT INTO messages (name, email, subject, message) VALUES (?, ?, ?, ?)";*

* **Kullanıcı girişi için veritabanında kayıtlı kullanıcı bilgileri kontrol edilir. Bu işlem SELECT sorgusunu kullanır.**

*Örneğin:*

*$sql = "SELECT id, name, email, subject, message, createdAt FROM messages ORDER BY createdAt DESC";*

* **Kullanıcının profil bilgilerini güncellemesinde UPDATE sorgusu kullanılır.**

*Örneğin:*

*$query = "UPDATE categories SET name='$categoryName',description =' $categoryDescription',image ='$categoryImage', isActive=$isActive WHERE id=$id";*

* **Adres silme işlemlerinde DELETE sorgusu kullanılır.**

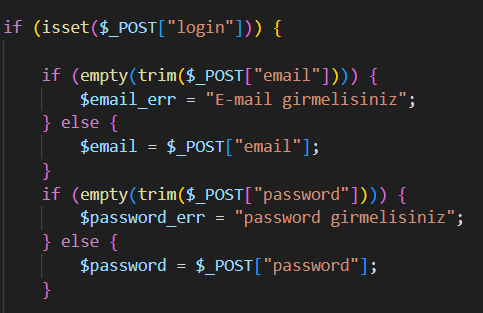
*Örneğin:*

*$query = "DELETE FROM categories WHERE id=$id";*

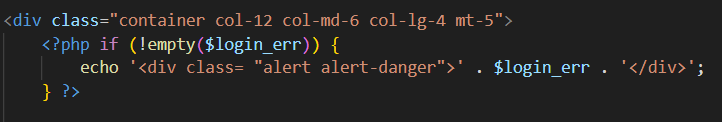
**4.3 FORM YÖNETİMİ**

4.3.1. VERİ DOĞRULAMA VE HATA YÖNETİMİ

Uygulamada kullanıcının girdiği email ve password bilgilerinin boş olup olmadığı, girilen bilgilerin doğru olup olmadığı kontrol edilmektedir. E-posta ve şifre hatası yoksa giriş işlemi yapılması sağlanmaktadır.



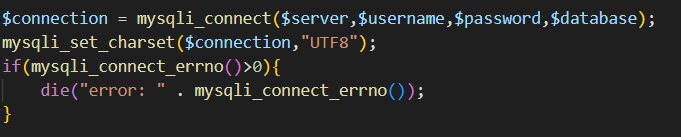
Login form alınan hatalar için de kullanıcıyı bilgilendirmek amaçlı kullanılan hata kontrolü mekanizması aşağıdaki gibidir.



Aynı şekilde kullanıcının yönetici kişiye mail göndermesi esnasında da bazı doğrulamalar ve hata yönetimi mevcuttur. Kullanıcıların mesaj gönderirken, tüm gerekli alanları doldurduklarından emin olmak için veri doğrulama kullanılır.



Veritabanı işlemleri sırasında oluşabilecek hatalar ve kullanıcıya gösterilecek hata mesajları açısından başka bir örnek de veritabanı bağlantısı sırasında çıkabilecek bağlantı sorunlarına karşı kullanılan hata kontrolüdür.



4.3.2. KULLANICIDAN ALINAN VERİLERİN İŞLENMESİ VE SAKLANMASI

Kullanıcının girdiği bilgiler işlenerek veri tabanına kaydedilmektedir. Örneğin kullanıcı kaydını oluşturduktan sonra giriş yapma sayfasında kullanıcının bilgileri veri tabanında saklanır ve giriş yapacağı zaman bu bilgiler işlenerek giriş yapması sağlanır. Bunun dışında kullanıcı mail göndereceği zaman da yine aynı şekilde kullanıcının verileri veritabanına kaydedilir ve bu bilgiler daha sonrasında *send\_contact.php* sayfasında işlenerek kullanıcının mail göndermesi sağlanır.



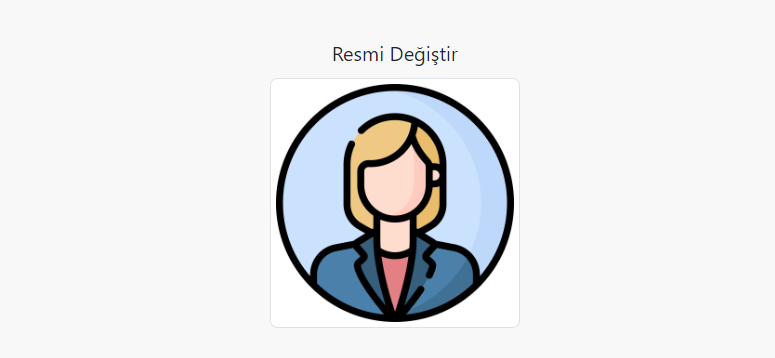
**4.4 DOSYA YÜKLEME**

4.4.1. KULLANICILARIN DOSYA YÜKLEYEBİLMESİ (ÖRNEĞİN RESİM YÜKLEME)

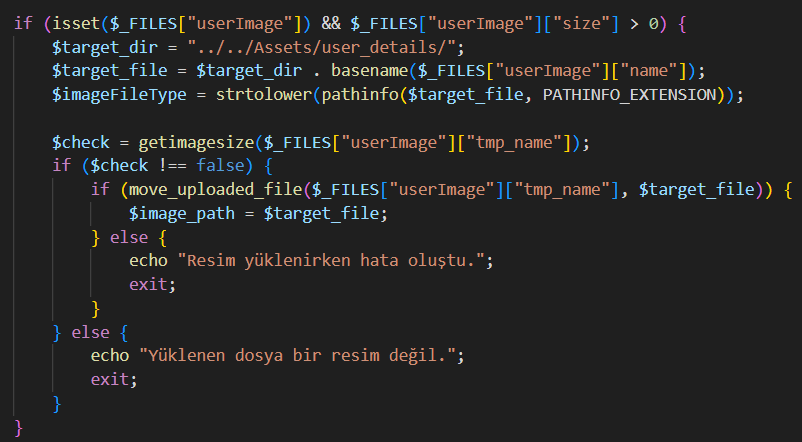
Kullanıcılar sisteme giriş yaptıktan sonra *MyProfile.php* sayfasına yönlendirmeleri gerçekleşir. Bu sayfada kişisel bilgilerinin yanında varsayılan olarak gelen profil resmini güncelleme işlemini gerçekleştirip sisteme fotoğraf yükleyebilmektedirler.



Varsayılan olarak gelen resmi *“Resmi Değiştir”* butonuna tıklayarak resim seçme işlemini yaptıktan sonra aşağıda yer alan *“Kaydet”* butonuna tıklanıldığında resim kaydetme işlemi başarılı bir şekilde yapılmaktadır.



Bu işlemlerin kod yapısı aşağıda verilmiştir.



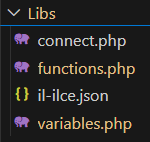
4.4.2. YÜKLENEN DOSYALARIN SUNUCUDA GÜVENLİ BİR ŞEKİLDE SAKLANMASI

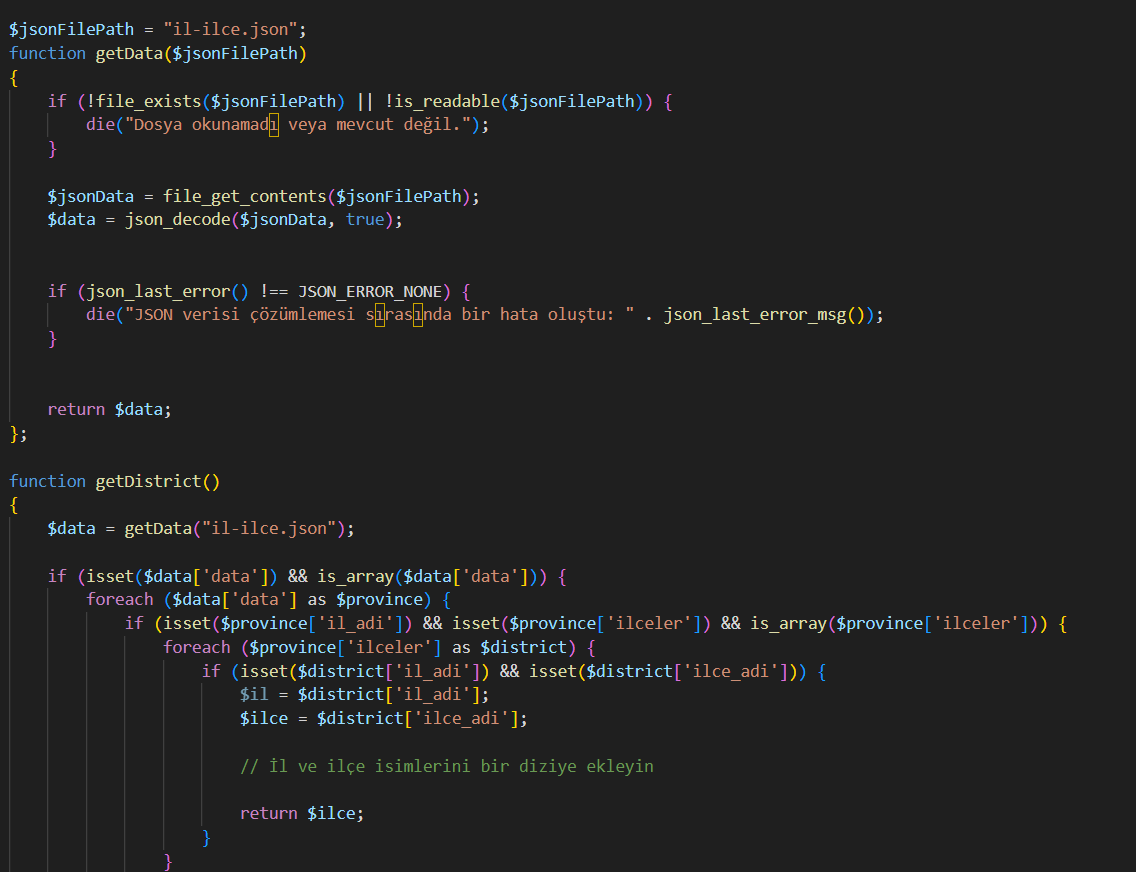
Yüklenen dosyanın güvenli bir şekilde sağlanması ve bazı kontrollerin yapılması *“MyProfile.php”* sayfasında sağlanmaktadır. Bunlardan bazıları, yüklenen dosya için maximum boyut kontrolünün yapılması, yüklenen dosya tipinin belirlenmesi gibi işlemlerdir.

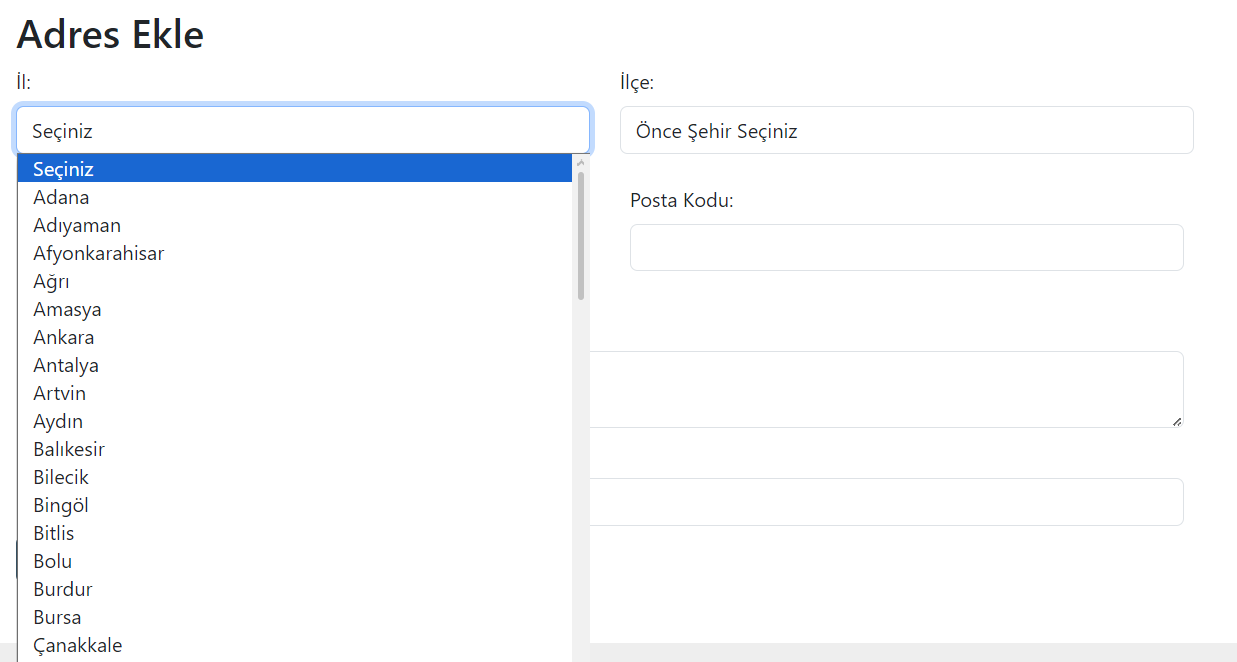


**4.5 JSON**

Uygulamada profil sayfasındaki Adres ekleme alanında Şehir isimleri ve buna bağlı olarak ilçe isimleri *“il-ilce.json”* dosyasından çekilmektedir.



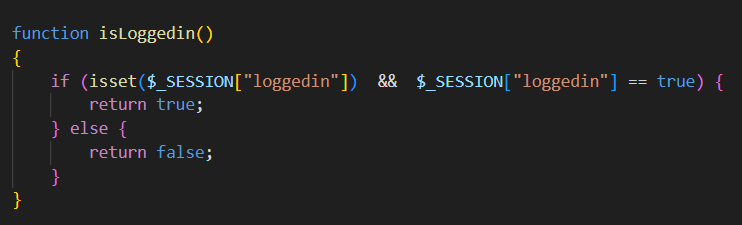




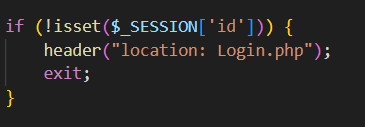
**4.6 OTURUM YÖNETİMİ**

4.6.1. PHP SESSIONS KULLANARAK OTURUM YÖNETİMİ

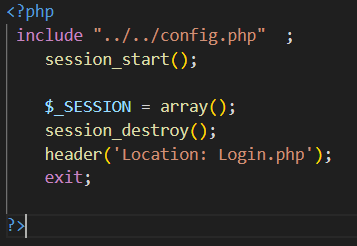
Kullanıcı sisteme giriş yaptığında   *$\_SESSION["loggedin"] = true;*  olur ve *“functions.php”* sayfasında bulunan *isLoggedin()*  fonksiyonu true değer döndürür.



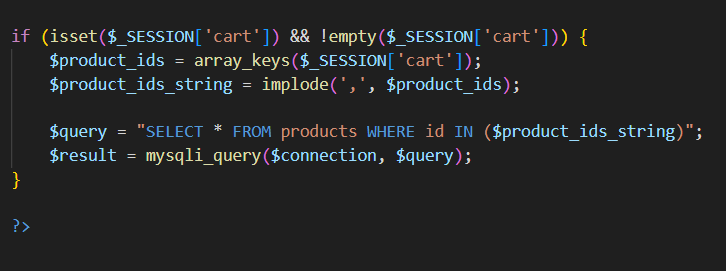
Kullanıcı giriş yapmadığı zaman Profilim sayfasına ulaşamaz ve bunu sağlayan yapı da Sessions ‘lardır. Örneğin:



Yukardaki kod parçasında giriş yapan kullanıcıya ait bir “id” bulunamazsa Profilim sayfası ekranda gözükmemektedir. Giriş yapılmadan görünmemesi gereken sayfaların başına yukardaki kod parçacığı eklenmiştir. Yine çıkış yapma işleminde de Session\_destroy() fonkiyonu yardımıyla oturum kapatılmaktadır.



Bu işlemlerin dışında Sepete ürün ekleme gibi bazı alanlarda da yine Sessions’lar kullanılmıştır.



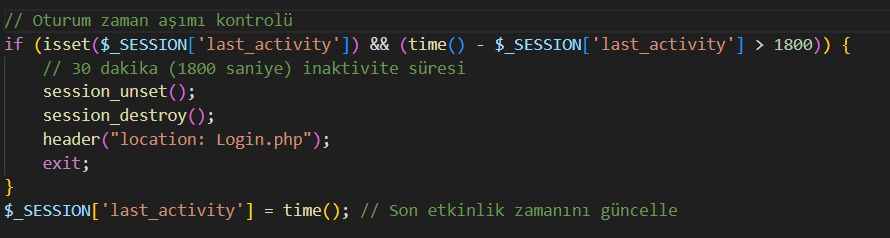
4.6.2. KULLANICI OTURUMLARININ GÜVENLİ BİR ŞEKİLDE YÖNETİLMESİ

Kullanıcı oturumlarının güvenliği açısından yapılan bazı önlemler şunlardır.

1. *“session\_regenerate\_id(true);”* bu fonksiyon geçerli oturum kimliğini yenisiyle değiştirirken oturum bilgisini korur ve böylece oturum sabitleme saldırılarını engeller.

**

1. Oturumda zaman aşımı kontrolü yaparak belli bir süreyi aşan (mesela 30 dk) kullanıcıların oturumunu sonlandırma işlemi yapılır.

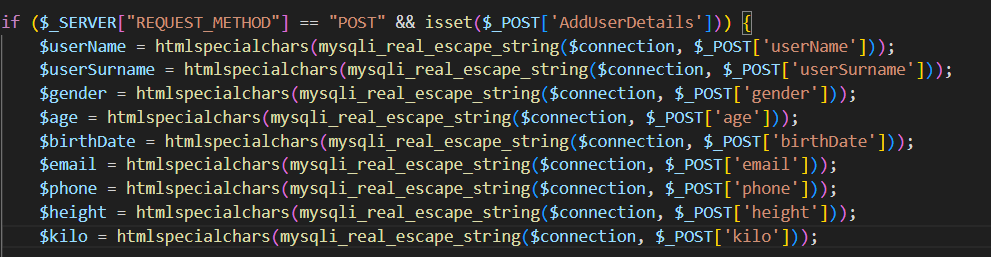


**4.7. GÜVENLİK**

4.7.1. GÜVENLİK ÖNLEMLERİ (ÖRNEĞİN CSRF, XSS KORUMALARI)

**XSS (Cross-Site Scripting) için alınan güvenlik önlemleri**

Kullanıcıdan alınan veriler veritabanına kaydedilmeden önce güvenli bir hale getirmek HTML/JS kodu olarak çalışmasını engellemek için ***“htmlspecialchars()”***ve ***“mysqli\_real\_escape\_string()”***fonksiyonları uygulamanın gerekli yerlerinde kullanılmıştır.



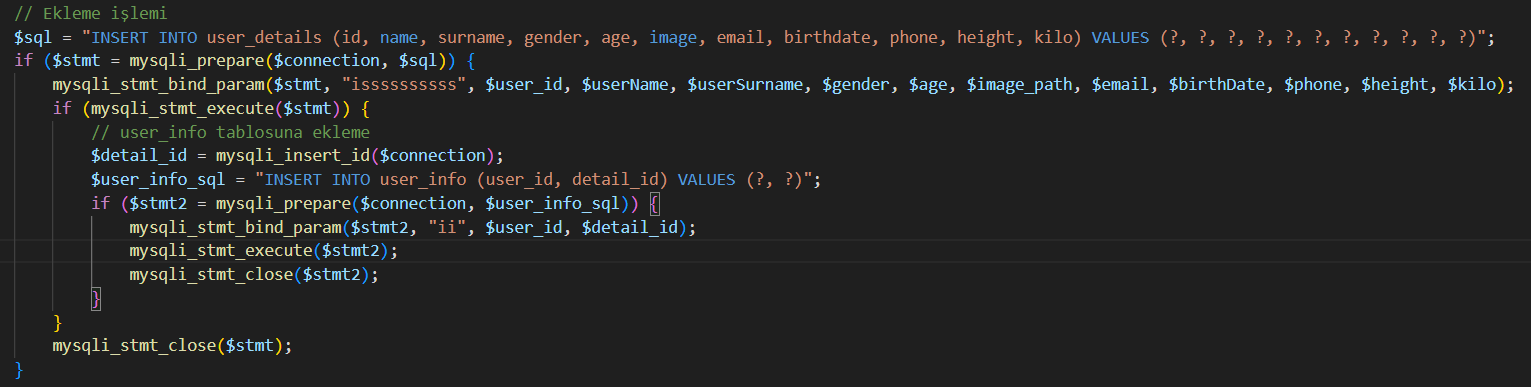
4.7.2. KULLANICI VERİLERİNİN GÜVENLİĞİNİN SAĞLANMASI

Bu aşamada kullanıcı verilerinin ve web uygulamasının güvenliğini sağlamak amacıyla belli bazı güvenlik önlemleri alınmıştır.

1. Veritabanı Güvenliği

**SQL enjeksiyonlarına karşı korunmak:**

Kullanıcıdan gelen verileri doğrudan SQL sorgularında kullanmak, SQL enjeksiyon saldırılarına karşı savunmasız hale getirir. Bunu önlemek için hazırlanmış ifadeler (prepared statements) ve parametreli sorgu yöntemleri kullanılmıştır.



1. Parola Güvenliği

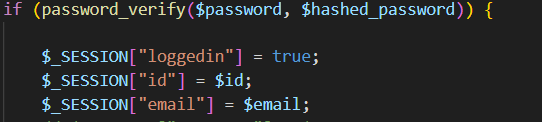
**Parolaların Hashlenmesi:**

Kullanıcının sisteme girdiği parolalar hashlenerek saklanmaktadır. Bunun için *“password\_hash” ve “password\_verify”* fonksiyonları kullanılmıştır.

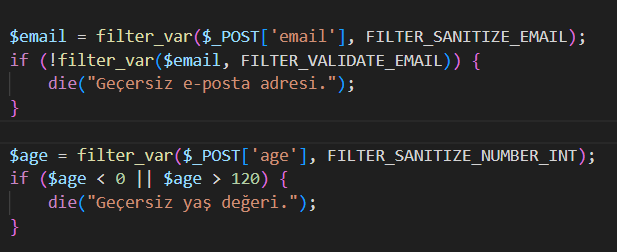
**Parola Hashleme:**



**Parola Doğrulama:**



1. Kullanıcının girdiği verilerin doğrulanması ve temizlenmesi

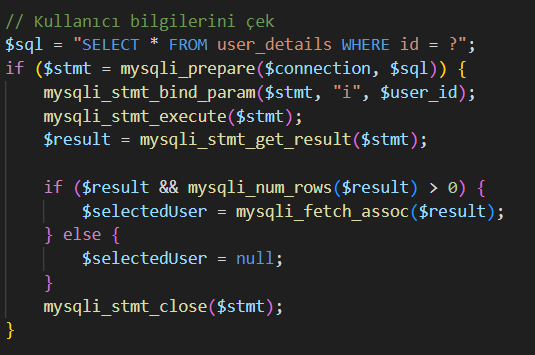


**4.8. PHP OOP ve PDO**

4.8.1. – 4.8.2. NESNE YÖNELİMLİ PROGRAMLAMA (OOP) PRENSİPLERİNE UYGUN KOD YAZMA – PHP PDO KULLANARAK VERİTABANI İŞLEMLERİNİ GERÇEKLEŞTİRME

Uygulamada MySQLi kullanılmıştır.

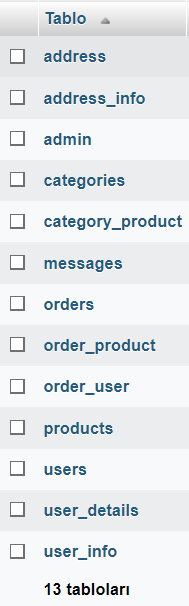
* Prosedürel MySQLi : Fonksiyonlar kullanılarak uygulanmıştır.



**4.9. LARAVEL KULLANIMI**

Uygulamada Laravel kullanılmamıştır.

**GENEL VERİ TABANI TABLOLARI**

****