**Güvenli Yazılım Geliştirme Final Projesi**

**SoftSafety Uygulaması**

**Adı ve Soyadı** : Gültekin Bütün

**Numara** : 1350D08251

**E-posta** : gltknbtn@gmail.com

# **SoftSafety v1.0**

### **Tanım**

SoftSafety v1.0, personel ekleme, güncelleme, silme ve arama işlemlerini gerçekleştiren web tabanlı bir uygulamadır. Kullanıcılar login işlemi sonrasında belirtilen işlevsellikleri sağlayabilmektedir. İlk aşamada uygulama, herhangi bir güvenlik tedbiri alınmaksızın SQL enjeksiyonu, kullanıcı parolalarının şifrelenmeden saklanması ve hata mesajlarında sistem veritabanı ile ilgili bilgiler vererek çeşitli zafiyetlere açık durumda geliştirilmişti. Bunlardan birkaç örnek aşağıda belirtilmiştir.

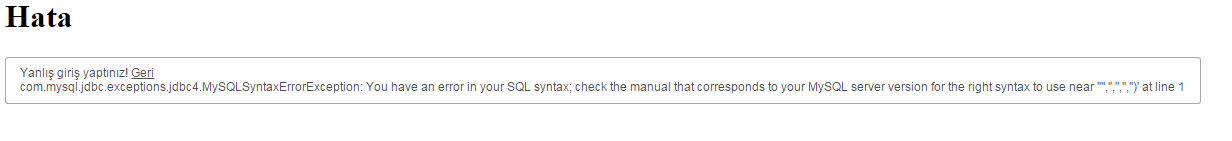
SQL enjeksiyonu ile sisteme giriş:

İlk aşamada Kullanıcı, ' or '1'='1 ifadesini “User Name” ve “Password” alanlarına yazarak sisteme başarılı bir şekilde giriş yapmaktaydı:

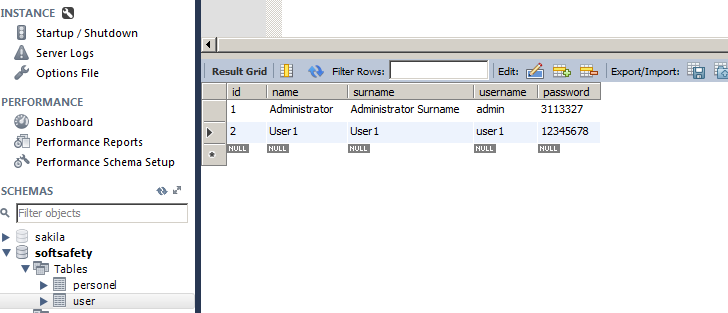


Hata mesajlarının sistem veritabanı ile ilgili kritik bilgileri belirtmesi:

Kullanıcı veritabanı ile ilgili keyword olarak bilinen karakter veya ifadeleri kullandığında ortaya çıkarılan hata mesajları sistemin kritik bilgilerini barındırmaktaydı:



Kullanıcı hesaplarına ait parolalar veritabanında şifrelenmemiş halde tutulmaktaydı:



### **Açıklıkların Düzeltilmesi**

SQL enjeksiyonlarının kapatılması: Aşağıda belirtilen ve login işlemini doğrulayan metot görüldüğü gibi düz SQL ifadesini barındıran “Statement” objesi kullanılarak oluşturulmuş böyle bir durumda “username” ve “password” alanına özel karakterler kullanılarak yazılan sql cümlecikleriyle sisteme giriş yapılabilmektedir.

**public** **boolean** validateLogin(String username, String password)

**throws** SQLException {

**boolean** auth = **false**;

ResultSet result = **null**;

Statement stmt = **null**;

Connection con = getConnection();

String sql = "select \* from User where username='" + username

+ "' and password='" + password + "'";

System.***out***.println("sql: " + sql);

stmt = con.createStatement();

result = stmt.executeQuery(sql);

**while** (result.next()) {

User user = **new** User();

user.setId(result.getInt("id"));

user.setUsername(result.getString("username"));

user.setPassword(result.getString("password"));

auth = **true**;

}

**return** auth;

}

Düzeltilmiş Durum: SQL cümleciklerini barındıran ifade “PreparedStatement” ifadesi kullanılarak oluşturulmuş ve sızıntının önüne geçilmiştir.

Düzeltilmiş ifade:

……

PreparedStatement statement = conn.prepareStatement( "SELECT \* FROM User WHERE username= ? AND password= ? " );

statement.setString(1, username);

statement.setString( 2, password );

result = statement.executeQuery();

……

Bir diğer açıklık kullanıcı ekleme işlemi yapılırken veritabanı ile ilgili özel karakterlerin ve ifadelerin kullanılmasıyla sistemin veritabanı hakkında bilgilerin kullanıcıya sunulmasıydı. Sistem bu durumdayken aşağıda belirtilen kod parçacığında belirtildiği üzere kullanıcıya direkt sistem tarafından üretilen hata mesajı sunulmaktadır:

**public** **void** savePersonel() {

**try** {

.

.

.

isSucces = service.save(personel);

**if** (isSucces) {

System.***out***.println(personel.getName()+" eklendi.");

personelList = service.getPersonelList();

clearFields();

}**else** System.***out***.println("Ekleme işleminde hata");

} **catch** (Exception e) {

errorMessage = e.toString();

e.printStackTrace();

FacesContext.*getCurrentInstance*().getApplication().getNavigationHandler().handleNavigation(FacesContext.*getCurrentInstance*(), **null**,"/hata?faces-redirect=true");

}

}

Üretilen mesaj handle edilerek kullanıcıya uygun hata mesajı gösterilmektedir.

Kullanıcılara ait bilgilerden parola plaintext olarak tutulmaktaydı. Bu güvenlik açığının giderilmesi için aşağıda belirtilen kod parçacığının yardımıyla parolaların hash bilgisi veritabanında tutulmaktadır.

**public** **static** String createPasswordHash(String pass) **throws** NoSuchAlgorithmException{

MessageDigest md = MessageDigest.*getInstance*("MD5");

md.update(pass.getBytes());

**byte**[] bytes = md.digest();

StringBuilder sb = **new** StringBuilder();

**for** (**int** i = 0; i < bytes.length; i++) {

sb.append(Integer.*toString*((bytes[i]&0xff)+0x100,16).substring(1));

}

String passwordHashCode = sb.toString();

System.***out***.println(passwordHashCode);

**return** passwordHashCode;

}

Nihai durumda kullanıcı bilgilerinin tutulduğu veritabanındaki parola durumu aşağıda belirtilmektedir.

