SQL Queries

Af Kasper

Alle metoder til at behandle SQL statements til vore MySQL database findes i DataAccessObject-klassen. Vi benytter os af PreparedStatements for at gøre systemet hurtigere og mere stabilt, og for at sikre os mod SQL-injection angreb.

public User getUserByUsername(String username) throws SQLException {

User user = null;

PreparedStatement stmt = null;

try {

stmt = dbcon.getConnection().prepareStatement("SELECT \* FROM users WHERE uname = ?;");

stmt.setString(1, username);

ResultSet rs = stmt.executeQuery();

if (rs.next()) {

int UID = rs.getInt("uid");

String usernameRetrieved = rs.getString("uname");

String passwordRetrieved = rs.getString("password");

String saltRetrieved = rs.getString("salt");

String emailRetrieved = rs.getString("email");

String userString = rs.getString("userstring");

String carportRetrieved = rs.getString("carport");

user = new User(UID, usernameRetrieved, passwordRetrieved, saltRetrieved, emailRetrieved, userString, carportRetrieved);

}

} finally {

try {

if (stmt != null) {

stmt.close();

}

} catch (Exception e) {

}

}

return user;

}

PreparedStatements indsætter de strings der bliver givet med til metoden der hvor spørgsmålstegnene står. Dette er ligegyldigt hvad der står i den string der kommer med, derfor bliver SQL-injection umuliggjort her.

MySQL-Databasen er opbygget specifikt efter vores egne behov. Derfor er den ikke særlig fleksibel. Dog er tabellen med dele til carporten bygget op så både skruer, brædder, søm osv. kan være i samme tabel. dette er gjort ved at definere forskellige dele-id'er som markerer forskellige typer materialer.

