



# SonarQube SAST Report

generated by Reflect Sonar

**Date:** 2025-09-24 23:59:58

**SonarQube Project Name:** vulnado

<b>Security</b> <b>0</b> Open Issues	A	<b>Reliability</b> <b>10</b> Open Issues	D	<b>Maintainability</b> <b>54</b> Open Issues	A
<b>Accepted Issues</b> <b>0</b> <small>Valid issues that were not fixed</small>	C	<b>Coverage</b> <b>0.0%</b> <small>on 205 lines to cover</small>	B	<b>Duplications</b> <b>0.0%</b> <small>on 571 lines</small>	D
<b>Security Hotspots</b> <b>18</b> Open Issues					

*This report is generated by ReflectSonar, an open-source tool to add the report generation mechanism to SonarQube Community and Developer Edition. It is not affiliated with SonarSource. The report is generated based on SonarQube instance that its information is provided. All data is fetched from SonarQube API. ReflectSonar just provides a way to generate the report.*



## Security Issues

Total: 0 issues

*No issues found in this category. This indicates good code quality in this area.*

## Reliability Issues

Total: 10 issues (High: 10)

Severity	File Path	Rule & Message
	src/main/java/com/scalesec/vulnado Comment.java (Line 41)	<b>S2095</b> Use try-with-resources or close this "Statement" in a "finally" clause.
<b>■ Problematic Code:</b> <pre> 38:     List&lt;Comment&gt; comments = new ArrayList(); 39:     try { 40:         Connection cxn = Postgres.connection(); &gt;&gt;&gt; 41:         stmt = cxn.createStatement(); 42: 43:         String query = "select * from comments;"; 44:         ResultSet rs = stmt.executeQuery(query); </pre>		
	src/main/java/com/scalesec/vulnado Comment.java (Line 58)	<b>S1143</b> Remove this return statement from this finally block.
<b>■ Problematic Code:</b> <pre> 55:         e.printStackTrace(); 56:         System.err.println(e.getClass().getName()+": "+e.getMessage()); 57:     } finally { &gt;&gt;&gt; 58:         return comments; 59:     } 60: } 61: </pre>		
	src/main/java/com/scalesec/vulnado Comment.java (Line 66)	<b>S2095</b> Use try-with-resources or close this "PreparedStatement" in a "finally" clause.
<b>■ Problematic Code:</b> <pre> 63:     try { 64:         String sql = "DELETE FROM comments where id = ?"; 65:         Connection con = Postgres.connection(); &gt;&gt;&gt; 66:         PreparedStatement pStatement = con.prepareStatement(sql); 67:         pStatement.setString(1, id); 68:         return 1 == pStatement.executeUpdate(); 69:     } catch(Exception e) { </pre>		
	src/main/java/com/scalesec/vulnado Comment.java (Line 72)	<b>S1143</b> Remove this return statement from this finally block.

<p>■ <b>Problematic Code:</b></p> <pre> 69:         } catch(Exception e) { 70:             e.printStackTrace(); 71:         } finally { &gt;&gt;&gt; 72:             return false; 73:         } 74:     } 75: </pre>		
<div>H</div>	src/main/java/com/scalesec/vulnado Comment.java (Line 79)	<b>S2095</b> Use try-with-resources or close this "PreparedStatement" in a "finally" clause.
<p>■ <b>Problematic Code:</b></p> <pre> 76:     private Boolean commit() throws SQLException { 77:         String sql = "INSERT INTO comments (id, username, body, created_on) VALUES (?,?,?,?)"; 78:         Connection con = Postgres.connection(); &gt;&gt;&gt; 79:         PreparedStatement pStatement = con.prepareStatement(sql); 80:         pStatement.setString(1, this.id); 81:         pStatement.setString(2, this.username); 82:         pStatement.setString(3, this.body); </pre>		
<div>H</div>	src/main/java/com/scalesec/vulnado Postgres.java (Line 94)	<b>S2095</b> Use try-with-resources or close this "PreparedStatement" in a "finally" clause.
<p>■ <b>Problematic Code:</b></p> <pre> 91:         String sql = "INSERT INTO users (user_id, username, password, created_on) VALUES (?, ?, ?, current_timestamp)"; 92:         PreparedStatement pStatement = null; 93:         try { &gt;&gt;&gt; 94:             pStatement = connection().prepareStatement(sql); 95:             pStatement.setString(1, UUID.randomUUID().toString()); 96:             pStatement.setString(2, username); 97:             pStatement.setString(3, md5(password)); </pre>		
<div>H</div>	src/main/java/com/scalesec/vulnado Postgres.java (Line 108)	<b>S2095</b> Use try-with-resources or close this "PreparedStatement" in a "finally" clause.
<p>■ <b>Problematic Code:</b></p> <pre> 105:         String sql = "INSERT INTO comments (id, username, body, created_on) VALUES (?, ?, ?, current_timestamp)"; 106:         PreparedStatement pStatement = null; 107:         try { &gt;&gt;&gt; 108:             pStatement = connection().prepareStatement(sql); 109:             pStatement.setString(1, UUID.randomUUID().toString()); 110:             pStatement.setString(2, username); 111:             pStatement.setString(3, body); </pre>		
<div>H</div>	src/main/java/com/scalesec/vulnado Postgres.java (Line 35)	<b>S2095</b> Use try-with-resources or close this "Statement" in a "finally" clause.

#### ■ Problematic Code:

```

32:         try {
33:             System.out.println("Setting up Database...");
34:             Connection c = connection();
>>> 35:             Statement stmt = c.createStatement();
36:
37:             // Create Schema
38:             stmt.executeUpdate("CREATE TABLE IF NOT EXISTS users(user_id VARCHAR (36)
PRIMARY KEY, username VARCHAR (50) UNIQUE NOT NULL, password VARCHAR (50) NOT NULL, created_on
TIMESTAMP NOT NULL, last_login TIMESTAMP)");

```

H

src/main/java/com/scalesec/vulnado  
User.java  
(Line 44)

#### S2095

Use try-with-resources or close this "Statement" in a "finally" clause.

#### ■ Problematic Code:

```

41:         User user = null;
42:         try {
43:             Connection cxn = Postgres.connection();
>>> 44:             stmt = cxn.createStatement();
45:             System.out.println("Opened database successfully");
46:
47:             String query = "select * from users where username = '" + un + "' limit 1";

```

H

src/main/java/com/scalesec/vulnado  
User.java  
(Line 61)

#### S1143

Remove this return statement from this finally block.

#### ■ Problematic Code:

```

58:         e.printStackTrace();
59:         System.err.println(e.getClass().getName()+": "+e.getMessage());
60:     } finally {
>>> 61:         return user;
62:     }
63: }
64: }

```

## Maintainability Issues

Total: 54 issues (High: 2, Medium: 16, Low: 36)

Severity	File Path	Rule & Message
<b>H</b>	src/main/java/com/scalesec/vulnado Comment.java (Line 62)	<b>S3516</b> Refactor this method to not always return the same value.
<b>■ Problematic Code:</b> <pre> 59:     } 60:   } 61: &gt;&gt;&gt; 62:   public static Boolean delete(String id) { 63:       try { 64:           String sql = "DELETE FROM comments where id = ?"; 65:           Connection con = Postgres.connection(); </pre>		
<b>H</b>	src/test/java/com/scalesec/vulnado VulnadoApplicationTests.java (Line 13)	<b>S1186</b> Add a nested comment explaining why this method is empty, throw an UnsupportedOperationException or complete the implementation.
<b>■ Problematic Code:</b> <pre> 10: public class VulnadoApplicationTests { 11: 12:     @Test &gt;&gt;&gt; 13:     public void contextLoads() { 14:     } 15: 16: } </pre>		
<b>M</b>	src/main/java/com/scalesec/vulnado LinkLister.java (Line 28)	<b>S106</b> Replace this use of System.out by a logger.
<b>■ Problematic Code:</b> <pre> 25:     try { 26:         URL aUrl= new URL(url); 27:         String host = aUrl.getHost(); &gt;&gt;&gt; 28:         System.out.println(host); 29:         if (host.startsWith("172.")    host.startsWith("192.168")    host.startsWith("10.")){ 30:             throw new BadRequest("Use of Private IP"); 31:         } else { </pre>		
<b>M</b>	src/main/java/com/scalesec/vulnado Comment.java (Line 38)	<b>S3740</b> Provide the parametrized type for this generic.

<b>■ Problematic Code:</b> <pre> 35: 36:  public static List&lt;Comment&gt; fetch_all() { 37:      Statement stmt = null; &gt;&gt;&gt; 38:      List&lt;Comment&gt; comments = new ArrayList(); 39:      try { 40:          Connection cxn = Postgres.connection(); 41:          stmt = cxn.createStatement(); </pre>		
	src/main/java/com/scalesec/vulnado Comment.java (Line 43)	<b>S6905</b> Don't use the query "SELECT *".
<b>■ Problematic Code:</b> <pre> 40:      Connection cxn = Postgres.connection(); 41:      stmt = cxn.createStatement(); 42: &gt;&gt;&gt; 43:      String query = "select * from comments;"; 44:      ResultSet rs = stmt.executeQuery(query); 45:      while (rs.next()) { 46:          String id = rs.getString("id"); </pre>		
	src/main/java/com/scalesec/vulnado Comment.java (Line 56)	<b>S106</b> Replace this use of System.err by a logger.
<b>■ Problematic Code:</b> <pre> 53:      cxn.close(); 54:  } catch (Exception e) { 55:      e.printStackTrace(); &gt;&gt;&gt; 56:      System.err.println(e.getClass().getName()+" : "+e.getMessage()); 57:  } finally { 58:      return comments; 59:  } </pre>		
	src/main/java/com/scalesec/vulnado Cowsay.java (Line 6)	<b>S1118</b> Add a private constructor to hide the implicit public one.
<b>■ Problematic Code:</b> <pre> 3: import java.io.BufferedReader; 4: import java.io.InputStreamReader; 5: &gt;&gt;&gt; 6: public class Cowsay { 7:     public static String run(String input) { 8:         ProcessBuilder processBuilder = new ProcessBuilder(); 9:         String cmd = "/usr/games/cowsay '" + input + "'"; </pre>		
	src/main/java/com/scalesec/vulnado Cowsay.java (Line 10)	<b>S106</b> Replace this use of System.out by a logger.

<p>■ <b>Problematic Code:</b></p> <pre> 7:   public static String run(String input) { 8:       ProcessBuilder processBuilder = new ProcessBuilder(); 9:       String cmd = "/usr/games/cowsay '" + input + "'"; &gt;&gt;&gt; 10:   System.out.println(cmd); 11:       processBuilder.command("bash", "-c", cmd); 12: 13:       StringBuilder output = new StringBuilder(); </pre>		
M	src/main/java/com/scalesec/vulnado LinkLister.java (Line 13)	<b>S1118</b> Add a private constructor to hide the implicit public one.
<p>■ <b>Problematic Code:</b></p> <pre> 10: import java.net.*; 11: 12: &gt;&gt;&gt; 13: public class LinkLister { 14:     public static List&lt;String&gt; getLinks(String url) throws IOException { 15:         List&lt;String&gt; result = new ArrayList&lt;String&gt;(); 16:         Document doc = Jsoup.connect(url).get(); </pre>		
M	src/main/java/com/scalesec/vulnado Postgres.java (Line 33)	<b>S106</b> Replace this use of System.out by a logger.
<p>■ <b>Problematic Code:</b></p> <pre> 30:     } 31:     public static void setup(){ 32:         try { &gt;&gt;&gt; 33:             System.out.println("Setting up Database..."); 34:             Connection c = connection(); 35:             Statement stmt = c.createStatement(); 36: </pre>		
M	src/main/java/com/scalesec/vulnado Postgres.java (Line 56)	<b>S106</b> Replace this use of System.out by a logger.
<p>■ <b>Problematic Code:</b></p> <pre> 53:         insertComment("alice", "OMG so cute!"); 54:         c.close(); 55:     } catch (Exception e) { &gt;&gt;&gt; 56:         System.out.println(e); 57:         System.exit(1); 58:     } 59: } </pre>		
M	src/main/java/com/scalesec/vulnado Postgres.java (Line 12)	<b>S1118</b> Add a private constructor to hide the implicit public one.



<p>■ <b>Problematic Code:</b></p> <pre> 9: import java.sql.Statement; 10: import java.util.UUID; 11: &gt;&gt;&gt; 12: public class Postgres { 13: 14:     public static Connection connection() { 15:         try { </pre>		
M	src/main/java/com/scalesec/vulnado Postgres.java (Line 16)	<b>S4925</b> Remove this "Class.forName()", it is useless.
<p>■ <b>Problematic Code:</b></p> <pre> 13: 14:     public static Connection connection() { 15:         try { &gt;&gt;&gt; 16:             Class.forName("org.postgresql.Driver"); 17:             String url = new StringBuilder() 18:                 .append("jdbc:postgresql://") 19:                 .append(System.getenv("PGHOST")) </pre>		
M	src/main/java/com/scalesec/vulnado Postgres.java (Line 26)	<b>S106</b> Replace this use of System.err by a logger.
<p>■ <b>Problematic Code:</b></p> <pre> 23:             System.getenv("PGUSER"), System.getenv("PGPASSWORD")); 24:         } catch (Exception e) { 25:             e.printStackTrace(); &gt;&gt;&gt; 26:             System.err.println(e.getClass().getName()+": "+e.getMessage()); 27:             System.exit(1); 28:         } 29:         return null; </pre>		
M	src/main/java/com/scalesec/vulnado Postgres.java (Line 86)	<b>S112</b> Define and throw a dedicated exception instead of using a generic one.
<p>■ <b>Problematic Code:</b></p> <pre> 83: 84:         // For specifying wrong message digest algorithms 85:         catch (NoSuchAlgorithmException e) { &gt;&gt;&gt; 86:             throw new RuntimeException(e); 87:         } 88:     } 89: </pre>		
M	src/main/java/com/scalesec/vulnado User.java (Line 45)	<b>S106</b> Replace this use of System.out by a logger.

<b>■ Problematic Code:</b> <pre> 42:     try { 43:         Connection cxn = Postgres.connection(); 44:         stmt = cxn.createStatement(); &gt;&gt;&gt; 45:         System.out.println("Opened database successfully"); 46: 47:         String query = "select * from users where username = '" + un + "' limit 1"; 48:         System.out.println(query); </pre>		
M	src/main/java/com/scalesec/vulnado User.java (Line 48)	<b>S106</b> Replace this use of System.out by a logger.
<b>■ Problematic Code:</b> <pre> 45:         System.out.println("Opened database successfully"); 46: 47:         String query = "select * from users where username = '" + un + "' limit 1"; &gt;&gt;&gt; 48:         System.out.println(query); 49:         ResultSet rs = stmt.executeQuery(query); 50:         if (rs.next()) { 51:             String user_id = rs.getString("user_id"); </pre>		
M	src/main/java/com/scalesec/vulnado User.java (Line 59)	<b>S106</b> Replace this use of System.err by a logger.
<b>■ Problematic Code:</b> <pre> 56:         cxn.close(); 57:     } catch (Exception e) { 58:         e.printStackTrace(); &gt;&gt;&gt; 59:         System.err.println(e.getClass().getName()+" : "+e.getMessage()); 60:     } finally { 61:         return user; 62:     } </pre>		
L	src/main/java/com/scalesec/vulnado Comment.java (Line 3)	<b>S1128</b> Remove this unused import 'org.apache.catalina.Server'.
<b>■ Problematic Code:</b> <pre> 1: package com.scalesec.vulnado; 2: &gt;&gt;&gt; 3: import org.apache.catalina.Server; 4: import java.sql.*; 5: import java.util.Date; 6: import java.util.List; </pre>		
L	src/main/java/com/scalesec/vulnado Comment.java (Line 11)	<b>S1104</b> Make id a static final constant or non-public and provide accessors if needed.

<div> <div>■ Problematic Code:</div> <pre> 8: import java.util.UUID; 9: 10: public class Comment { &gt;&gt;&gt; 11:   public String id, username, body; 12:   public Timestamp created_on; 13: 14:   public Comment(String id, String username, String body, Timestamp created_on) { </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado Comment.java (Line 11)	<b>S1104</b> Make username a static final constant or non-public and provide accessors if needed.
<div> <div>■ Problematic Code:</div> <pre> 8: import java.util.UUID; 9: 10: public class Comment { &gt;&gt;&gt; 11:   public String id, username, body; 12:   public Timestamp created_on; 13: 14:   public Comment(String id, String username, String body, Timestamp created_on) { </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado Comment.java (Line 11)	<b>S1104</b> Make body a static final constant or non-public and provide accessors if needed.
<div> <div>■ Problematic Code:</div> <pre> 8: import java.util.UUID; 9: 10: public class Comment { &gt;&gt;&gt; 11:   public String id, username, body; 12:   public Timestamp created_on; 13: 14:   public Comment(String id, String username, String body, Timestamp created_on) { </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado Comment.java (Line 11)	<b>S1659</b> Declare "username" and all following declarations on a separate line.
<div> <div>■ Problematic Code:</div> <pre> 8: import java.util.UUID; 9: 10: public class Comment { &gt;&gt;&gt; 11:   public String id, username, body; 12:   public Timestamp created_on; 13: 14:   public Comment(String id, String username, String body, Timestamp created_on) { </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado Comment.java (Line 12)	<b>S1104</b> Make created_on a static final constant or non-public and provide accessors if needed.

<p>■ <b>Problematic Code:</b></p> <pre> 9: 10: public class Comment { 11:   public String id, username, body; &gt;&gt;&gt; 12:   public Timestamp created_on; 13: 14:   public Comment(String id, String username, String body, Timestamp created_on) { 15:     this.id = id; </pre>		
<p>L</p>	<p>src/main/java/com/scalesec/vulnado Comment.java (Line 12)</p>	<p><b>S116</b> Rename this field "created_on" to match the regular expression <code>^[a-z][a-zA-Z0-9]*\$</code>.</p>
<p>■ <b>Problematic Code:</b></p> <pre> 9: 10: public class Comment { 11:   public String id, username, body; &gt;&gt;&gt; 12:   public Timestamp created_on; 13: 14:   public Comment(String id, String username, String body, Timestamp created_on) { 15:     this.id = id; </pre>		
<p>L</p>	<p>src/main/java/com/scalesec/vulnado Comment.java (Line 14)</p>	<p><b>S117</b> Rename this local variable to match the regular expression <code>^[a-z][a-zA-Z0-9]*\$</code>.</p>
<p>■ <b>Problematic Code:</b></p> <pre> 11:   public String id, username, body; 12:   public Timestamp created_on; 13: &gt;&gt;&gt; 14:   public Comment(String id, String username, String body, Timestamp created_on) { 15:     this.id = id; 16:     this.username = username; 17:     this.body = body; </pre>		
<p>L</p>	<p>src/main/java/com/scalesec/vulnado Comment.java (Line 26)</p>	<p><b>S5411</b> Use a primitive boolean expression here.</p>
<p>■ <b>Problematic Code:</b></p> <pre> 23:   Timestamp timestamp = new Timestamp(time); 24:   Comment comment = new Comment(UUID.randomUUID().toString(), username, body, timestamp); 25:   try { &gt;&gt;&gt; 26:     if (comment.commit()) { 27:       return comment; 28:     } else { 29:       throw new BadRequest("Unable to save comment"); </pre>		
<p>L</p>	<p>src/main/java/com/scalesec/vulnado Comment.java (Line 36)</p>	<p><b>S100</b> Rename this method name to match the regular expression <code>^[a-z][a-zA-Z0-9]*\$</code>.</p>

<p>■ <b>Problematic Code:</b></p> <pre> 33:     } 34: } 35: &gt;&gt;&gt; 36: public static List&lt;Comment&gt; fetch_all() { 37:     Statement stmt = null; 38:     List&lt;Comment&gt; comments = new ArrayList(); 39:     try { </pre>		
L	src/main/java/com/scalesec/vulnado Comment.java (Line 49)	<b>S117</b> Rename this local variable to match the regular expression '[a-z][a-zA-Z0-9]*\$'.
<p>■ <b>Problematic Code:</b></p> <pre> 46:         String id = rs.getString("id"); 47:         String username = rs.getString("username"); 48:         String body = rs.getString("body"); &gt;&gt;&gt; 49:         Timestamp created_on = rs.getTimestamp("created_on"); 50:         Comment c = new Comment(id, username, body, created_on); 51:         comments.add(c); 52:     } </pre>		
L	src/main/java/com/scalesec/vulnado CommentsController.java (Line 17)	<b>S4488</b> Replace "@RequestMapping(method = RequestMethod.GET)" with "@GetMapping"
<p>■ <b>Problematic Code:</b></p> <pre> 14: private String secret; 15: 16: @CrossOrigin(origins = "*") &gt;&gt;&gt; 17: @RequestMapping(value = "/comments", method = RequestMethod.GET, produces = "application/json") 18: List&lt;Comment&gt; comments(@RequestHeader(value="x-auth-token") String token) { 19:     User.assertAuth(secret, token); 20:     return Comment.fetch_all(); </pre>		
L	src/main/java/com/scalesec/vulnado CommentsController.java (Line 24)	<b>S4488</b> Replace "@RequestMapping(method = RequestMethod.POST)" with "@PostMapping"
<p>■ <b>Problematic Code:</b></p> <pre> 21:     } 22: 23: @CrossOrigin(origins = "*") &gt;&gt;&gt; 24: @RequestMapping(value = "/comments", method = RequestMethod.POST, produces = "application/json", consumes = "application/json") 25: Comment createComment(@RequestHeader(value="x-auth-token") String token, @RequestBody CommentRequest input) { 26:     return Comment.create(input.username, input.body); 27: } </pre>		
L	src/main/java/com/scalesec/vulnado CommentsController.java (Line 30)	<b>S4488</b> Replace "@RequestMapping(method = RequestMethod.DELETE)" with "@DeleteMapping"

<div> <div>■ Problematic Code:</div> <pre> 27:     } 28: 29:     @CrossOrigin(origins = "*") &gt;&gt;&gt; 30:     @RequestMapping(value = "/comments/{id}", method = RequestMethod.DELETE, produces = "application/json") 31:     Boolean deleteComment(@RequestHeader(value="x-auth-token") String token, @PathVariable("id") String id) { 32:         return Comment.delete(id); 33:     } </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado CommentsController.java (Line 37)	<b>S1104</b> Make username a static final constant or non-public and provide accessors if needed.
<div> <div>■ Problematic Code:</div> <pre> 34: } 35: 36: class CommentRequest implements Serializable { &gt;&gt;&gt; 37:     public String username; 38:     public String body; 39: } 40: </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado CommentsController.java (Line 38)	<b>S1104</b> Make body a static final constant or non-public and provide accessors if needed.
<div> <div>■ Problematic Code:</div> <pre> 35: 36: class CommentRequest implements Serializable { 37:     public String username; &gt;&gt;&gt; 38:     public String body; 39: } 40: 41: @ResponseStatus(HttpStatus.BAD_REQUEST) </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado LoginController.java (Line 7)	<b>S1128</b> Remove this unused import 'org.springframework.stereotype'.
<div> <div>■ Problematic Code:</div> <pre> 4: import org.springframework.http.HttpStatus; 5: import org.springframework.web.bind.annotation.*; 6: import org.springframework.boot.autoconfigure.*; &gt;&gt;&gt; 7: import org.springframework.stereotype.*; 8: import org.springframework.beans.factory.annotation.*; 9: import java.io.Serializable; 10: </pre> </div>		
<div>L</div>	src/main/java/com/scalesec/vulnado LoginController.java (Line 35)	<b>S1104</b> Make token a static final constant or non-public and provide accessors if needed.


<b>■ Problematic Code:</b> <pre> 32: } 33: 34: class LoginResponse implements Serializable { &gt;&gt;&gt; 35:     public String token; 36:     public LoginResponse(String msg) { this.token = msg; } 37: } 38: </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado User.java (Line 7)	<b>S1128</b> Remove this unused import 'io.jsonwebtoken.JwtParser'.
<b>■ Problematic Code:</b> <pre> 4: import java.sql.Statement; 5: import java.sql.ResultSet; 6: import io.jsonwebtoken.Jwts; &gt;&gt;&gt; 7: import io.jsonwebtoken.JwtParser; 8: import io.jsonwebtoken.SignatureAlgorithm; 9: import io.jsonwebtoken.security.Keys; 10: import javax.crypto.SecretKey; </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado User.java (Line 8)	<b>S1128</b> Remove this unused import 'io.jsonwebtoken.SignatureAlgorithm'.
<b>■ Problematic Code:</b> <pre> 5: import java.sql.ResultSet; 6: import io.jsonwebtoken.Jwts; 7: import io.jsonwebtoken.JwtParser; &gt;&gt;&gt; 8: import io.jsonwebtoken.SignatureAlgorithm; 9: import io.jsonwebtoken.security.Keys; 10: import javax.crypto.SecretKey; 11: </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado User.java (Line 23)	<b>S1488</b> Immediately return this expression instead of assigning it to the temporary variable "jws".
<b>■ Problematic Code:</b> <pre> 20: 21:     public String token(String secret) { 22:         SecretKey key = Keys.hmacShaKeyFor(secret.getBytes()); &gt;&gt;&gt; 23:         String jws = Jwts.builder().setSubject(this.username).signWith(key).compact(); 24:         return jws; 25:     } 26: </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado CowController.java (Line 6)	<b>S1128</b> Remove this unused import 'java.io.Serializable'.

<p>■ <b>Problematic Code:</b></p> <pre> 3: import org.springframework.web.bind.annotation.*; 4: import org.springframework.boot.autoconfigure.*; 5: &gt;&gt;&gt; 6: import java.io.Serializable; 7: 8: @RestController 9: @EnableAutoConfiguration </pre>		
L	src/main/java/com/scalesec/vulnado LinkLister.java (Line 15)	<b>S2293</b> Replace the type specification in this constructor call with the diamond operator ("<>").
<p>■ <b>Problematic Code:</b></p> <pre> 12: 13: public class LinkLister { 14:     public static List&lt;String&gt; getLinks(String url) throws IOException { &gt;&gt;&gt; 15:         List&lt;String&gt; result = new ArrayList&lt;String&gt;(); 16:         Document doc = Jsoup.connect(url).get(); 17:         Elements links = doc.select("a"); 18:         for (Element link : links) { </pre>		
L	src/main/java/com/scalesec/vulnado LinksController.java (Line 3)	<b>S1128</b> Remove this unused import 'org.springframework.boot'.
<p>■ <b>Problematic Code:</b></p> <pre> 1: package com.scalesec.vulnado; 2: &gt;&gt;&gt; 3: import org.springframework.boot.*; 4: import org.springframework.http.HttpStatus; 5: import org.springframework.web.bind.annotation.*; 6: import org.springframework.boot.autoconfigure.*; </pre>		
L	src/main/java/com/scalesec/vulnado LinksController.java (Line 4)	<b>S1128</b> Remove this unused import 'org.springframework.http.HttpStatus'.
<p>■ <b>Problematic Code:</b></p> <pre> 1: package com.scalesec.vulnado; 2: 3: import org.springframework.boot.*; &gt;&gt;&gt; 4: import org.springframework.http.HttpStatus; 5: import org.springframework.web.bind.annotation.*; 6: import org.springframework.boot.autoconfigure.*; 7: import java.util.List; </pre>		
L	src/main/java/com/scalesec/vulnado LinksController.java (Line 8)	<b>S1128</b> Remove this unused import 'java.io.Serializable'.



<b>■ Problematic Code:</b> <pre> 5: import org.springframework.web.bind.annotation.*; 6: import org.springframework.boot.autoconfigure.*; 7: import java.util.List; &gt;&gt;&gt; 8: import java.io.Serializable; 9: import java.io.IOException; 10: 11: </pre>		
	src/main/java/com/scalesec/vulnado LoginController.java <b>(Line 3)</b>	<b>S1128</b> Remove this unused import 'org.springframework.boot'.
<b>■ Problematic Code:</b> <pre> 1: package com.scalesec.vulnado; 2: &gt;&gt;&gt; 3: import org.springframework.boot.*; 4: import org.springframework.http.HttpStatus; 5: import org.springframework.web.bind.annotation.*; 6: import org.springframework.boot.autoconfigure.*; </pre>		
	src/main/java/com/scalesec/vulnado LoginController.java <b>(Line 18)</b>	<b>S4488</b> Replace "@RequestMapping(method = RequestMethod.POST)" with "@PostMapping"
<b>■ Problematic Code:</b> <pre> 15: private String secret; 16: 17: @CrossOrigin(origins = "*") &gt;&gt;&gt; 18: @RequestMapping(value = "/login", method = RequestMethod.POST, produces = "application/json", consumes = "application/json") 19: LoginResponse login(@RequestBody LoginRequest input) { 20:     User user = User.fetch(input.username); 21:     if (Postgres.md5(input.password).equals(user.hashPassword)) { </pre>		
	src/main/java/com/scalesec/vulnado LoginController.java <b>(Line 30)</b>	<b>S1104</b> Make username a static final constant or non-public and provide accessors if needed.
<b>■ Problematic Code:</b> <pre> 27: } 28: 29: class LoginRequest implements Serializable { &gt;&gt;&gt; 30:     public String username; 31:     public String password; 32: } 33: </pre>		
	src/main/java/com/scalesec/vulnado LoginController.java <b>(Line 31)</b>	<b>S1104</b> Make password a static final constant or non-public and provide accessors if needed.

<b>■ Problematic Code:</b> <pre> 28: 29: class LoginRequest implements Serializable { 30:     public String username; &gt;&gt;&gt; 31:     public String password; 32: } 33: 34: class LoginResponse implements Serializable { </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado Postgres.java (Line 79)	<b>S1643</b> Use a StringBuilder instead.
<b>■ Problematic Code:</b> <pre> 76:         // Convert message digest into hex value 77:         String hashtext = no.toString(16); 78:         while (hashtext.length() &lt; 32) { &gt;&gt;&gt; 79:             hashtext = "0" + hashtext; 80:         } 81:         return hashtext; 82:     } </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado User.java (Line 13)	<b>S1104</b> Make id a static final constant or non-public and provide accessors if needed.
<b>■ Problematic Code:</b> <pre> 10: import javax.crypto.SecretKey; 11: 12: public class User { &gt;&gt;&gt; 13:     public String id, username, hashedPassword; 14: 15:     public User(String id, String username, String hashedPassword) { 16:         this.id = id; </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado User.java (Line 13)	<b>S1104</b> Make username a static final constant or non-public and provide accessors if needed.
<b>■ Problematic Code:</b> <pre> 10: import javax.crypto.SecretKey; 11: 12: public class User { &gt;&gt;&gt; 13:     public String id, username, hashedPassword; 14: 15:     public User(String id, String username, String hashedPassword) { 16:         this.id = id; </pre>		
<b>L</b>	src/main/java/com/scalesec/vulnado User.java (Line 13)	<b>S1104</b> Make hashedPassword a static final constant or non-public and provide accessors if needed.

<b>■ Problematic Code:</b> <pre> 10: import javax.crypto.SecretKey; 11: 12: public class User { &gt;&gt;&gt; 13:   public String id, username, hashedPassword; 14: 15:   public User(String id, String username, String hashedPassword) { 16:     this.id = id; </pre>		
	src/main/java/com/scalesec/vulnado User.java (Line 13)	<b>S1659</b> Declare "username" and all following declarations on a separate line.
<b>■ Problematic Code:</b> <pre> 10: import javax.crypto.SecretKey; 11: 12: public class User { &gt;&gt;&gt; 13:   public String id, username, hashedPassword; 14: 15:   public User(String id, String username, String hashedPassword) { 16:     this.id = id; </pre>		
	src/main/java/com/scalesec/vulnado User.java (Line 51)	<b>S117</b> Rename this local variable to match the regular expression <code>^[a-z][a-zA-Z0-9]*\$</code> .
<b>■ Problematic Code:</b> <pre> 48:     System.out.println(query); 49:     ResultSet rs = stmt.executeQuery(query); 50:     if (rs.next()) { &gt;&gt;&gt; 51:       String user_id = rs.getString("user_id"); 52:       String username = rs.getString("username"); 53:       String password = rs.getString("password"); 54:       user = new User(user_id, username, password); </pre>		