Application Security Lab

Name: Swanand Garge

Div : D(D2)

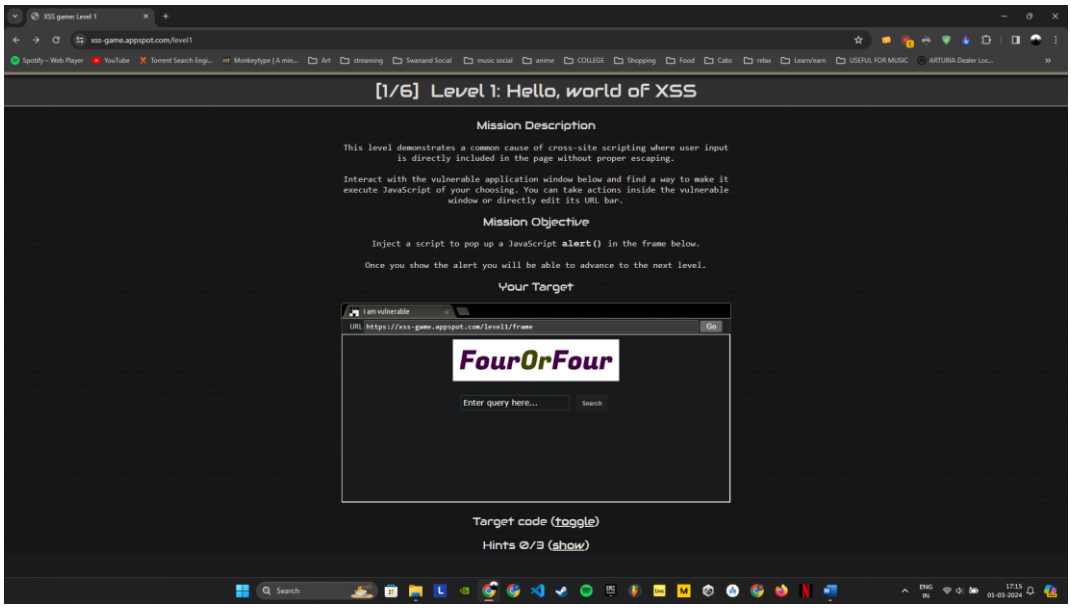
Roll no :42

SRN: 202201589

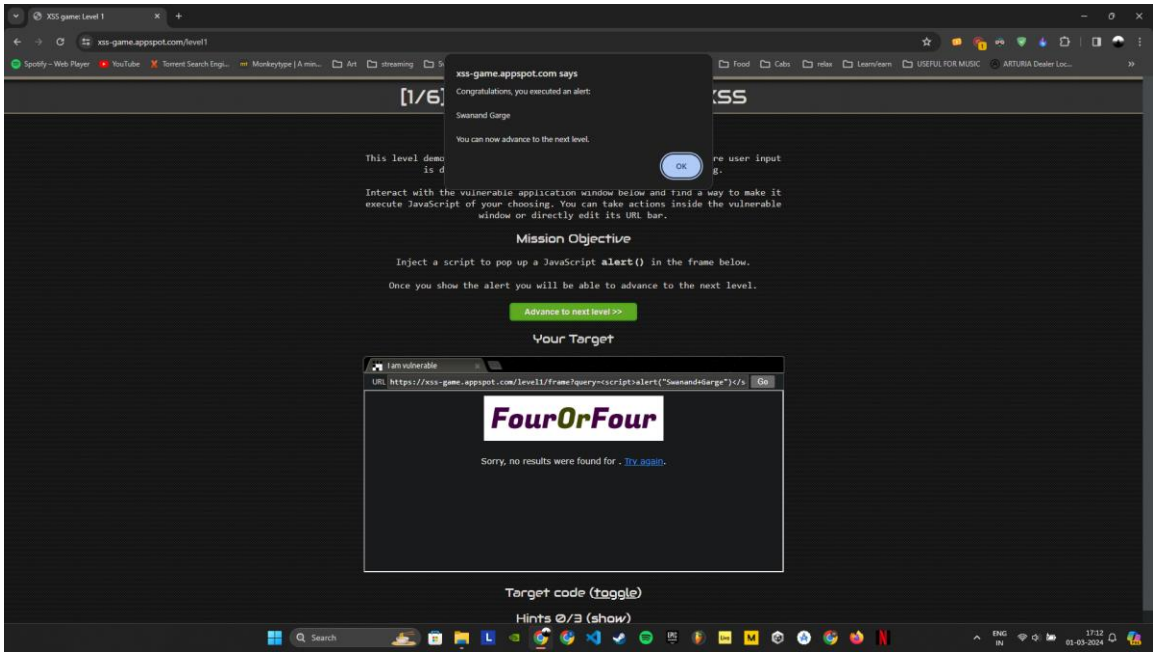
Q . ***Complete all the 6 stages of the xss game from***

https://xss-game.appspot.com

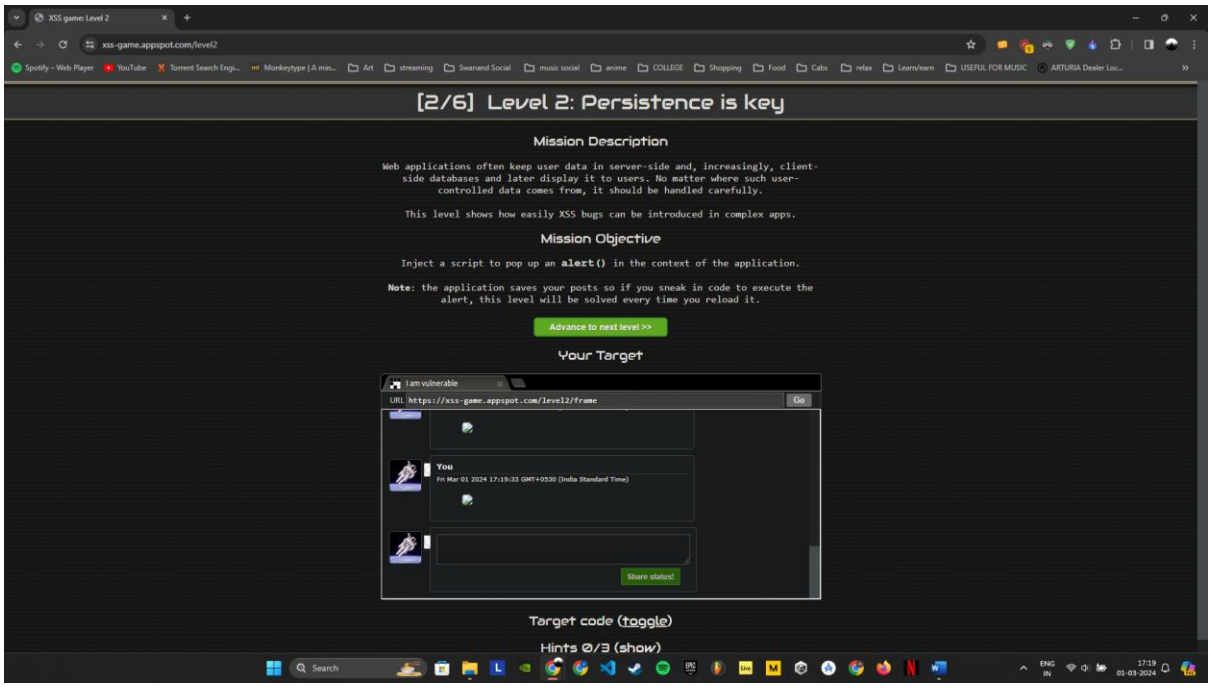
Level 1: Hello, world of XSS



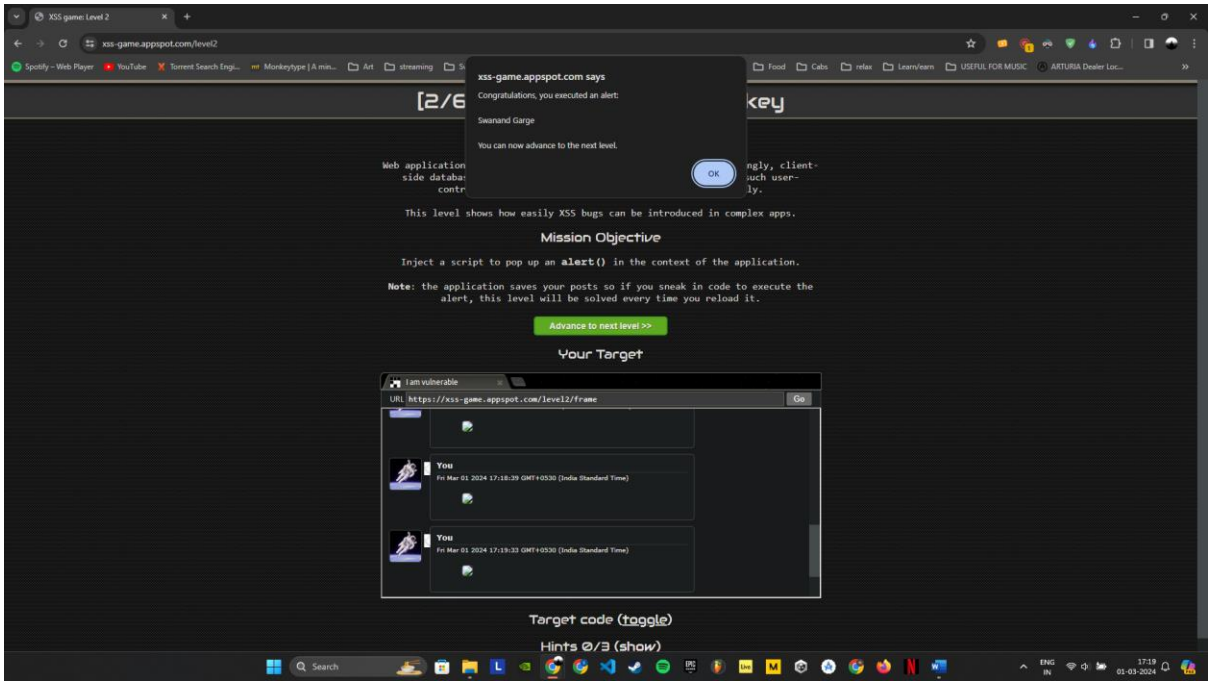
Output:



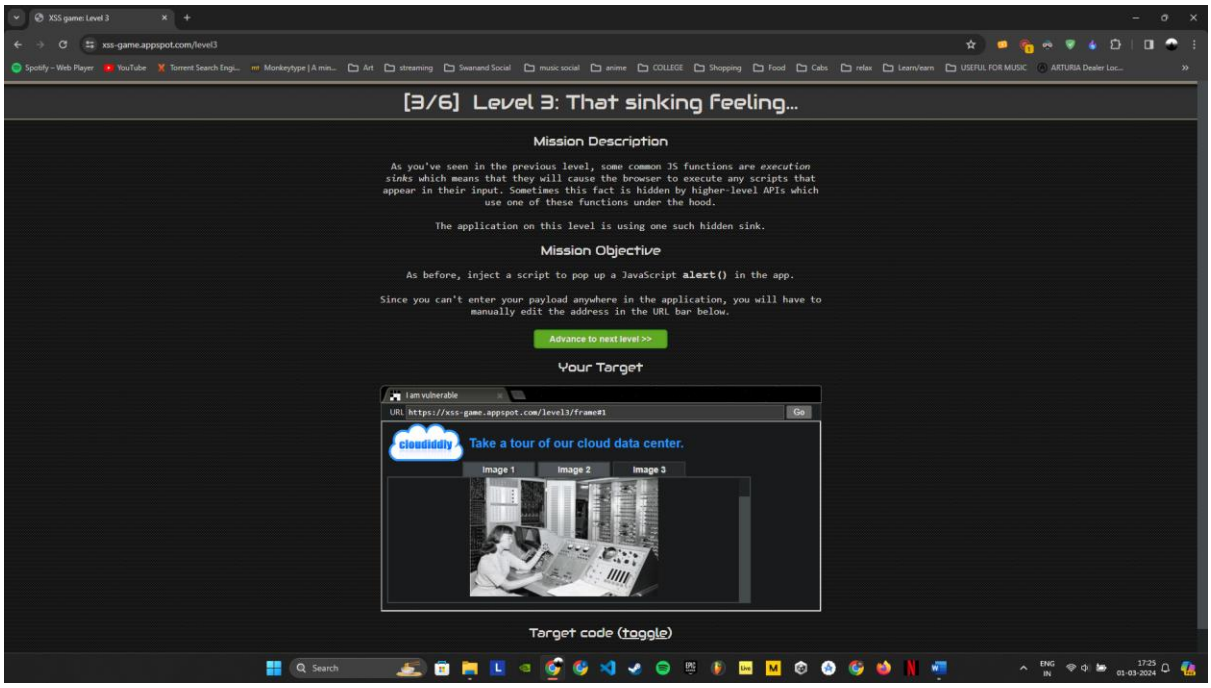
Level 2: Persistence is key



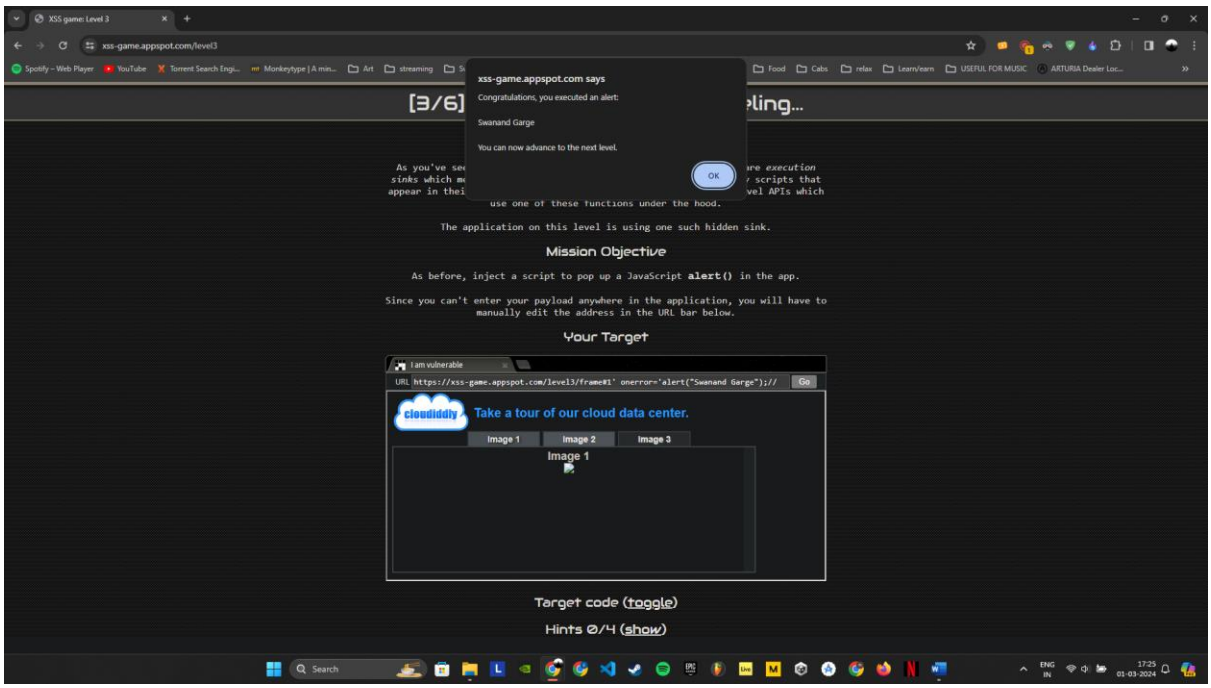
Output:



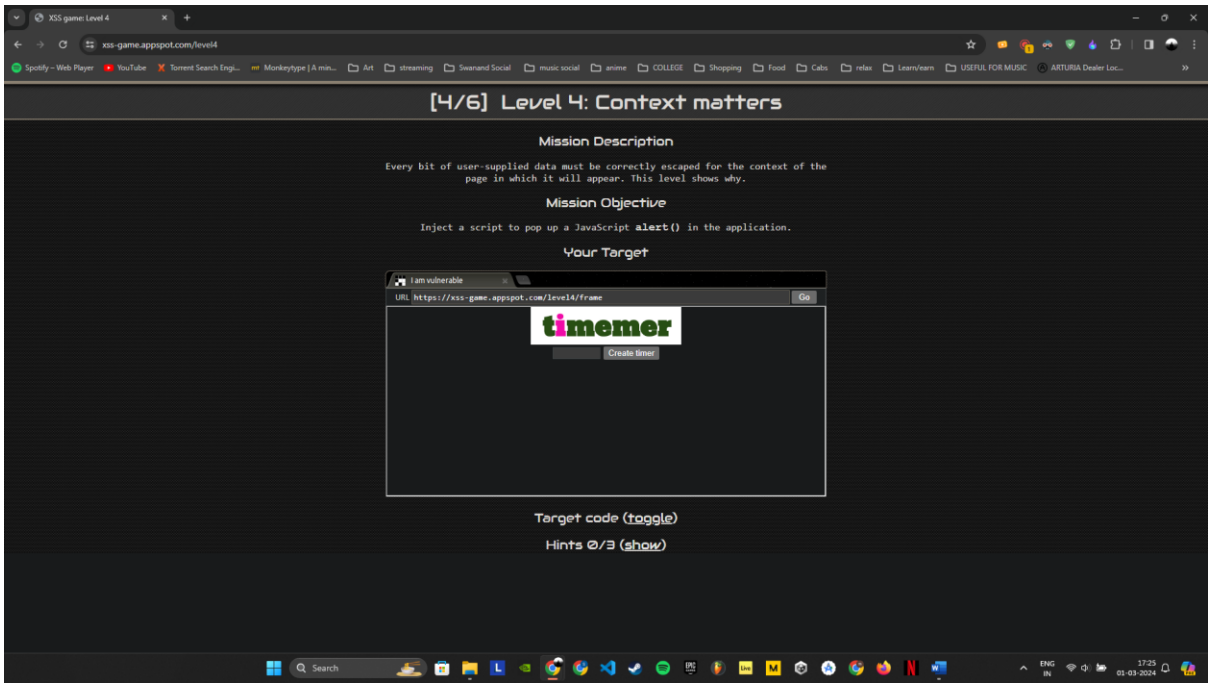
Level 3: That sinking feeling...



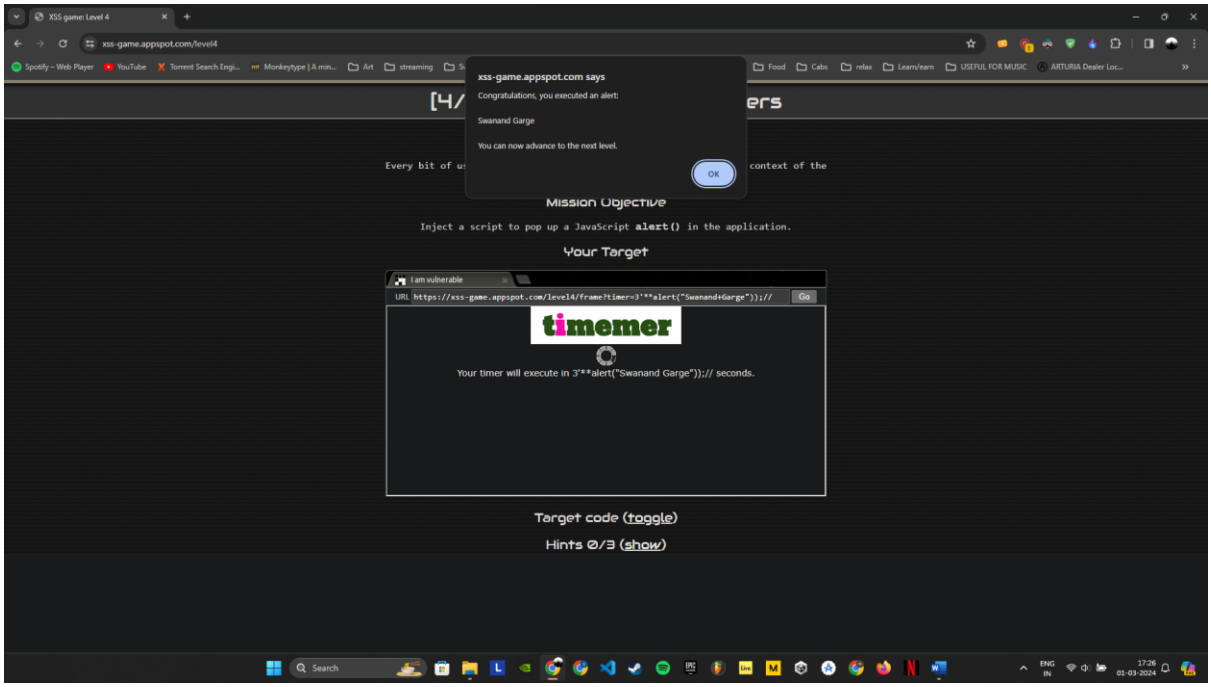
Output:



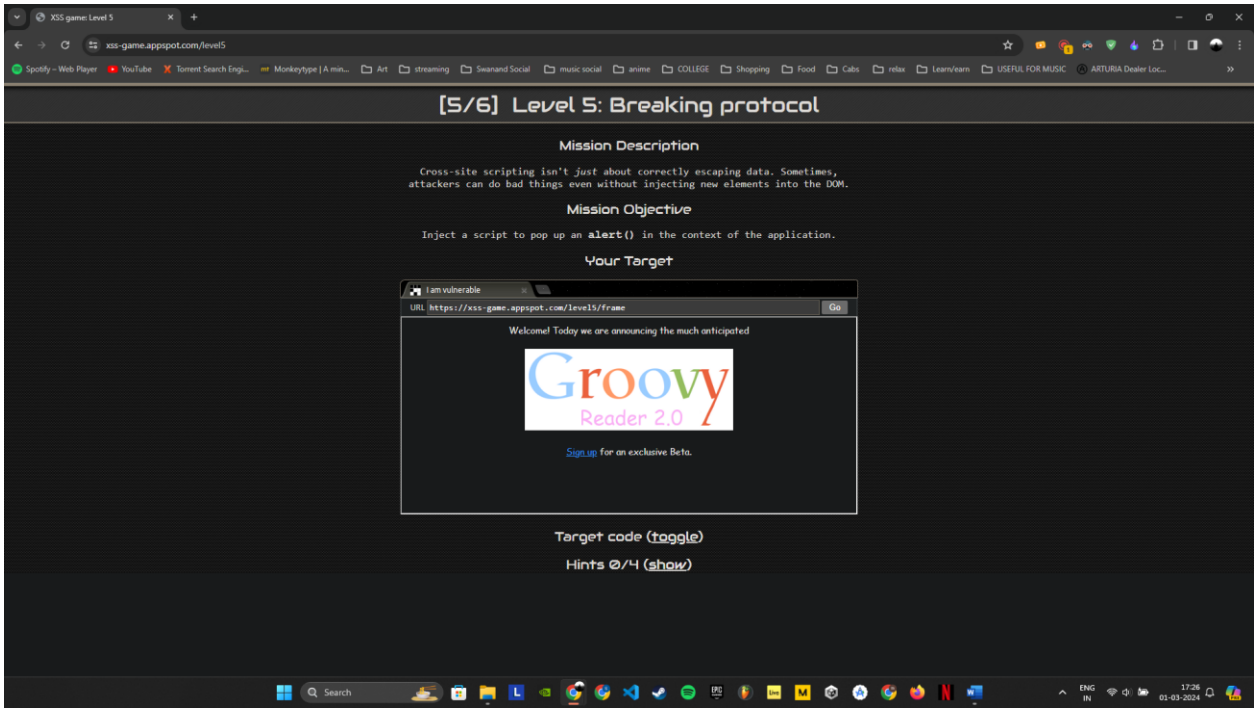
Level 4: Context matters



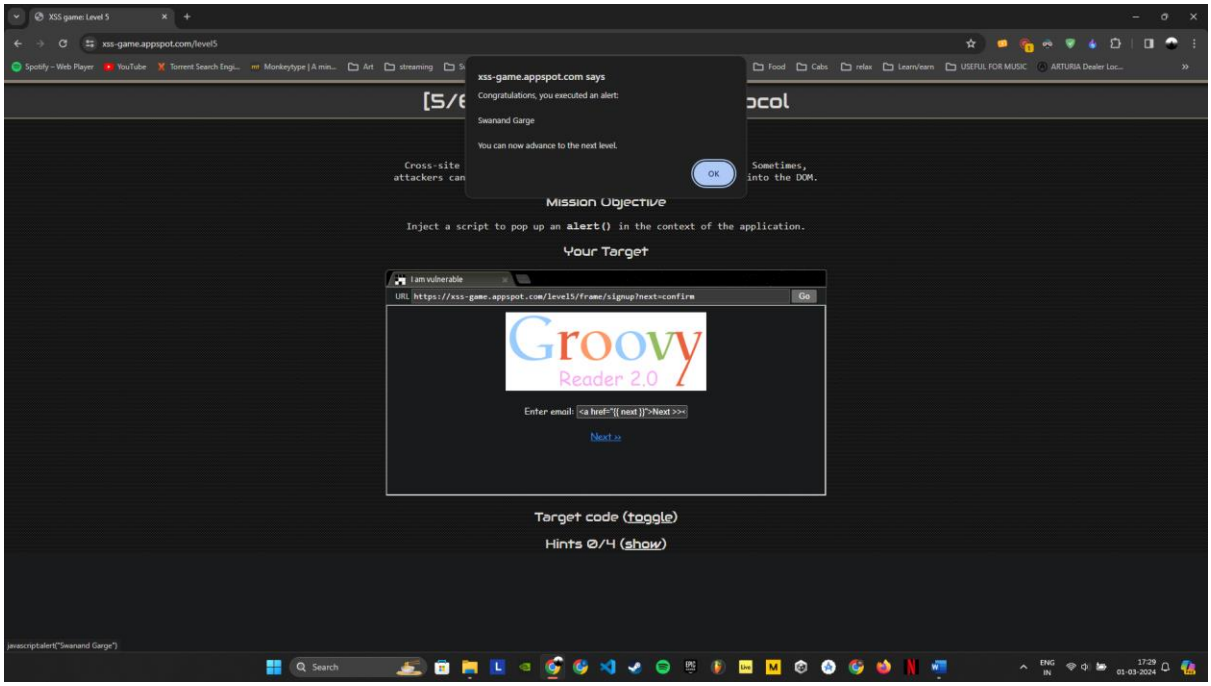
Output:



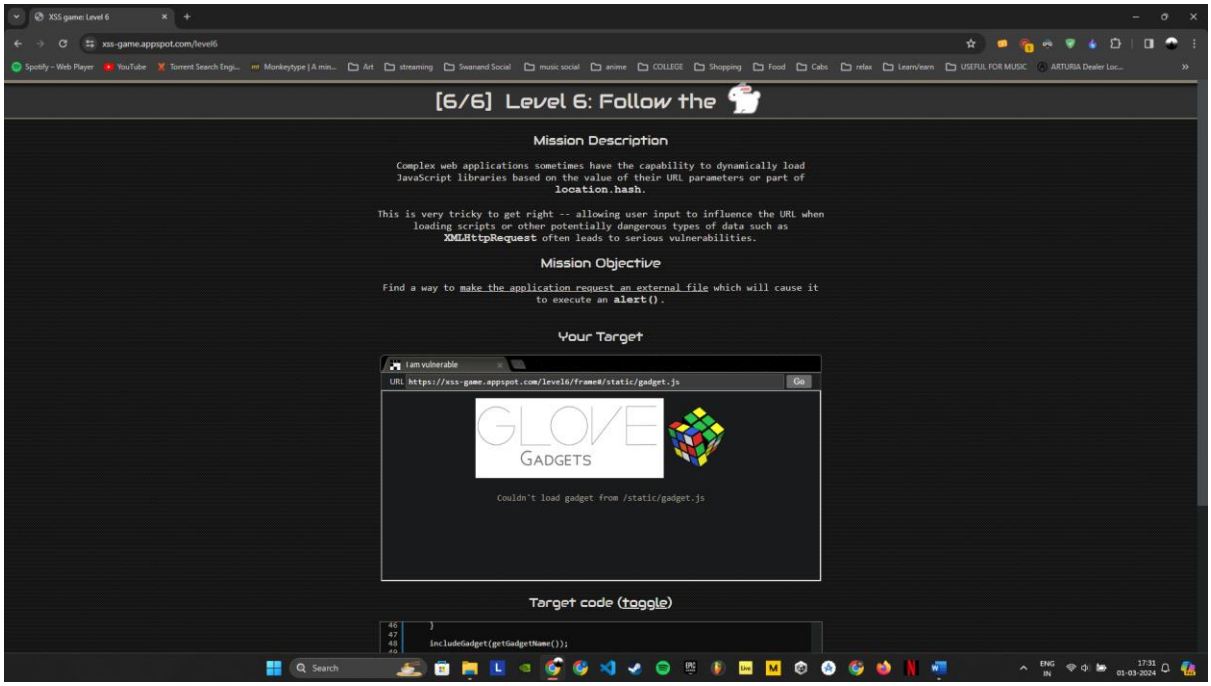
Level 5: Breaking protocol



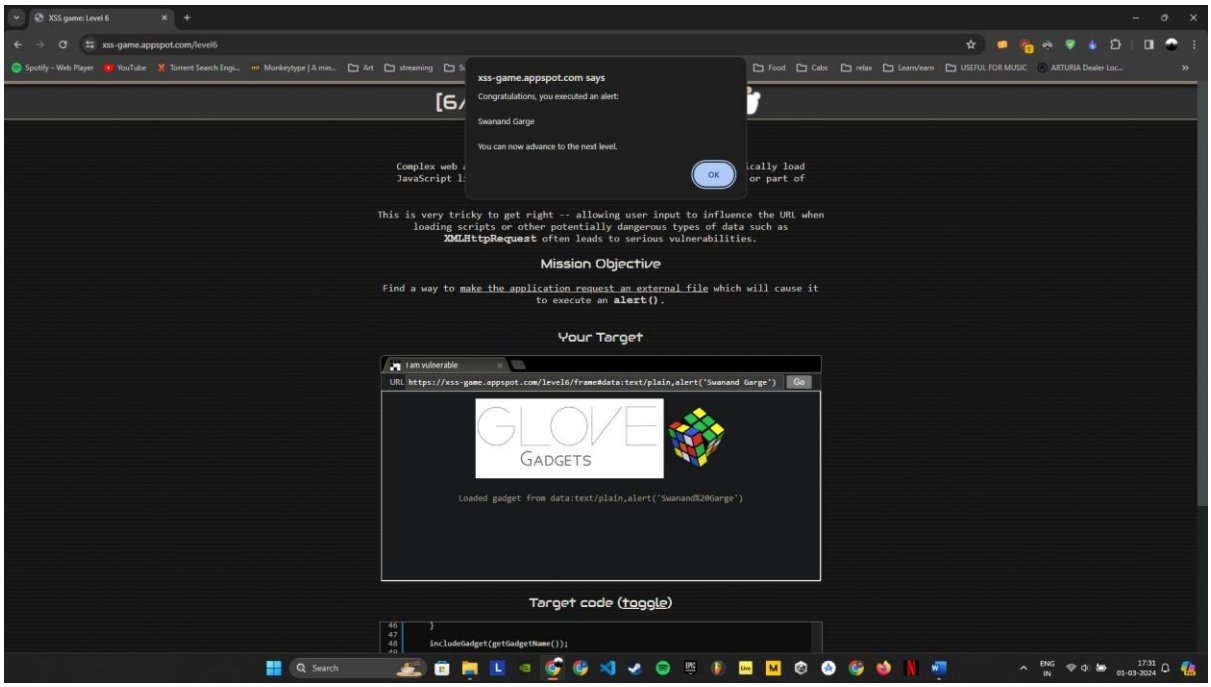
Output:



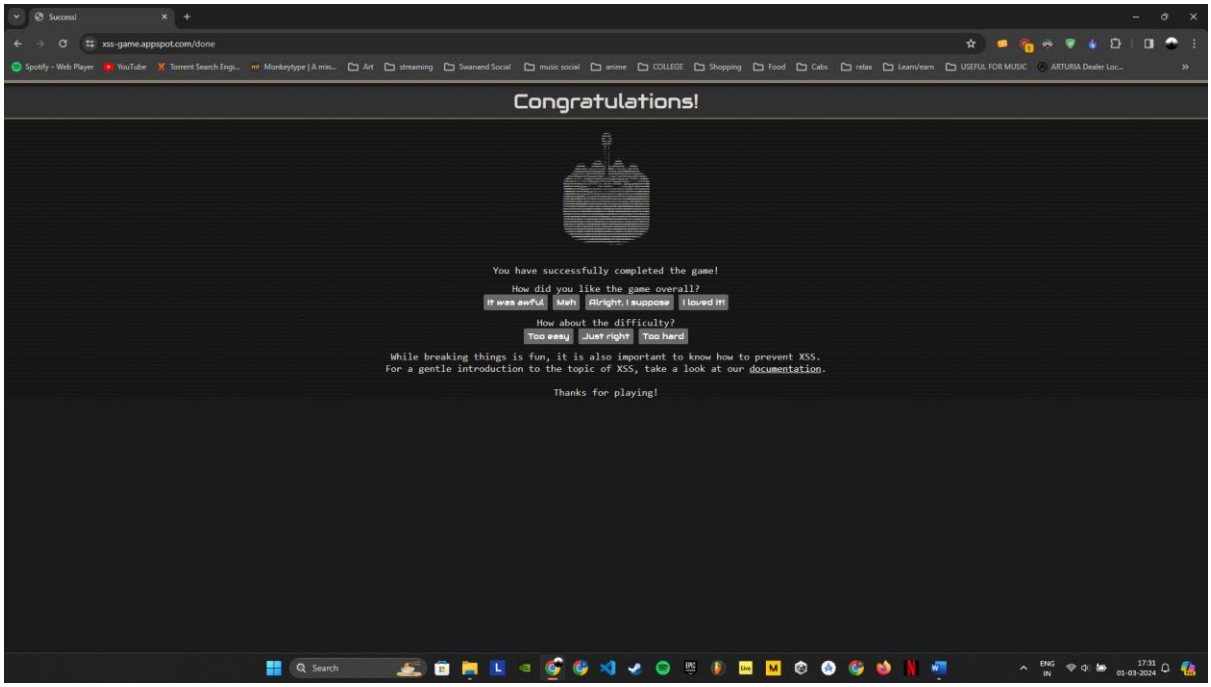
Level 6: Follow the



Output:



Final screen:



Conclusions:

Level 1: Hello, world of XSS

Conclusion: We grasped the fundamentals of XSS by embedding a simple script into an input field, resulting in an alert box popping up. This demonstrates how attackers can inject malicious code into vulnerable web pages.

Answer: Injecting ***<script>alert('Swanand Garge')</script>*** into the input field.

Level 2: Persistence is key

Conclusion: we now understand the concept of persistent XSS, where malicious code remains active even after the page reloads. By storing the script in the website's database, it infects any visitor to that page, emphasizing the importance of secure handling of user inputs.

Answer: Storing ***<img src='x' onerror='alert("Swanand Garge")'>*** as a comment on the page.

Level 3: That sinking feeling...

Conclusion: This level highlighted how URLs can propagate XSS attacks. By manipulating the URL parameters, attackers can insert harmful scripts executed when someone accesses that URL. It's akin to hiding a trap within a link.

Answer: Modifying the URL to ***https://xss-game.appspot.com/level3/frame#1' onerror='alert();//***

Level 4: Context matters

Conclusion: Recognizing the importance of adjusting attack strategies based on different parts of a webpage. Injecting code into text paragraphs requires a different approach than injecting code into the website's code itself, demonstrating the need to tailor attacks to specific contexts.

Answer: Injecting ***3'**alert("Swanand Garge"));//*** into the text box

Level 5: Breaking protocol

Conclusion: You've learned to bypass security measures by exploiting JavaScript event handlers. Understanding how these handlers work and interact with website security protocols enabled you to execute malicious code despite security measures. It's akin to finding a secret backdoor into a locked room.

Answer: change URL to ***https://xss-game.appspot.com/level5/frame/signup?next=javascript:alert("Swanand Garge")***

and write the following in the text box of sign in

<script>setTimeout(function(alert("Swanand Garge")) { window.location = '{{ next}}'; }, 5000);</script>

Level 6: Follow the... red herrings?

Conclusion: This level emphasizes the critical importance of conducting comprehensive security audits and considering all potential attack vectors, even if a website appears secure at first glance. It underscores the need for persistence and thorough examination in identifying hidden vulnerabilities that may be overlooked. By scrutinizing the code meticulously, participants learn valuable lessons in proactive security measures and the continuous vigilance required to safeguard against XSS attacks.

Answer: Change the URL to ***https://xss-game.appspot.com/level6/frame#data:text/plain,alert('xss')***