Vulnerability Assessment and Penetration Testing (VAPT) on Web Application

Submitted by:
Harsh Shukla
B.Tech 1st year
United College of Engineering And Research

Prayagraj

Supervisor: Mr. Ayush Sir
Date: 30 July 2025

Abstract

In today's digital era, web applications play a very important role for businesses, education, and personal use. Almost every organization has its own website or application, which makes them a major target for cyber-attacks. This project focuses on performing Vulnerability Assessment and Penetration Testing (VAPT) on a purposely vulnerable web application called OWASP Juice Shop.

The purpose of this project was to understand how attackers exploit weaknesses in web applications and how such attacks can be prevented. I tested for three major vulnerabilities: SQL Injection, Cross-Site Scripting (XSS), and Broken Access Control. Each vulnerability was carefully tested using manual techniques so that I could learn the fundamentals rather than relying on automated tools.

The project helped me understand the importance of secure coding, proper validation of user input, and role-based access control in web applications. I also faced multiple challenges during testing, such as payloads not working and needing to research the right techniques. This hands-on experience was extremely valuable because it showed me how a real-world attacker could think and act.

The final result of the project was a clear understanding of how web vulnerabilities can lead to serious security breaches and what measures can be taken to fix them.

Table of Contents

1. Introduction

Web applications have become the backbone of almost every service. They store sensitive data like usernames, passwords, banking details, and other personal information. This makes them a prime target for hackers. Many people think only large companies get hacked, but small websites are attacked even more because their security is often weak.

As a first-year student interested in cybersecurity, I wanted to understand how websites get hacked and how to stop it. That is why I chose the project 'VAPT on Web Application'. VAPT stands for Vulnerability Assessment and Penetration Testing.

Vulnerability Assessment is the process of scanning a website and finding weaknesses. Penetration Testing is like acting as a hacker and trying to exploit those weaknesses to see how much damage they can cause.

In this project, I used OWASP Juice Shop, which is an open-source web application made for students and professionals to practice hacking legally.

2. Literature Review

Before starting this project, I studied OWASP Top 10, which is a list of the most common web application vulnerabilities. I also went through documentation and YouTube tutorials about SQL Injection and XSS.

- SQL Injection: A method where attackers insert malicious SQL queries into input fields to bypass authentication or extract data from databases.
- Cross-Site Scripting (XSS): Allows attackers to inject scripts into web pages, which can then steal cookies or redirect users to malicious sites.
- Broken Access Control: Happens when users can access data or pages they are not authorized to see.

I learned that even small mistakes in code can lead to major security risks.

3. Methodology

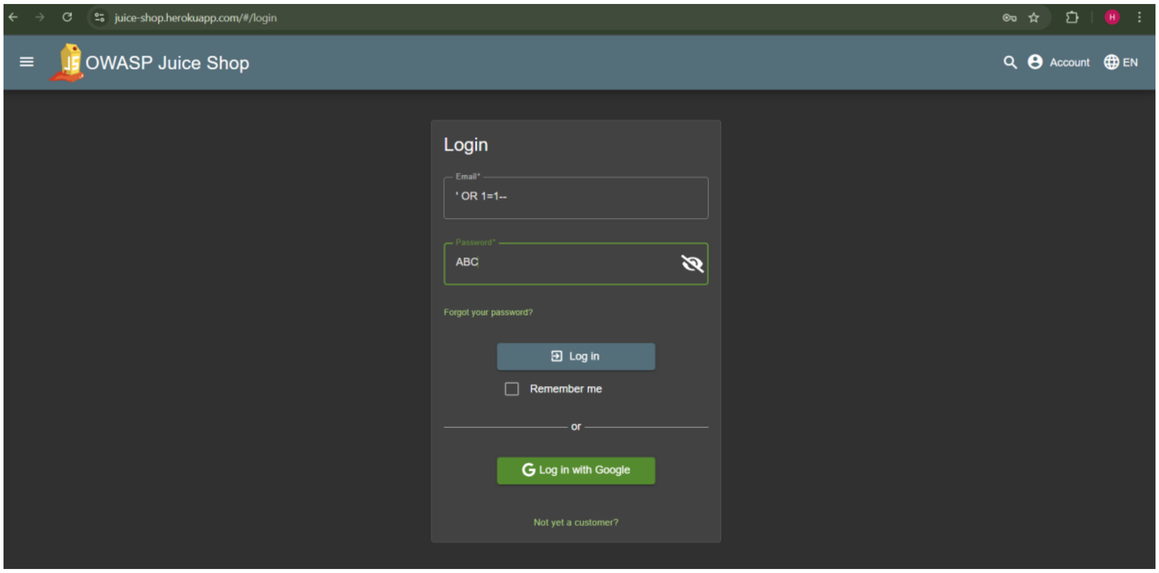
Tools and Environment:

- Tested Application: OWASP Juice Shop (https://owasp-juice.shop)
- Browser: Google Chrome
- Testing Style: Manual testing (no automated tools used)
- Techniques Used: Payload injection, URL manipulation, observing error messages

Step 1: SQL Injection

- Went to the login page and entered ' OR 1=1-- as the username and any random password.
- This payload tricks the database into always returning true, bypassing login authentication.
- Successfully logged in without knowing the actual username or password.

(Screenshot 1: SQL Injection login bypass)



juice-shop.herokuapp.com/#/login

Login

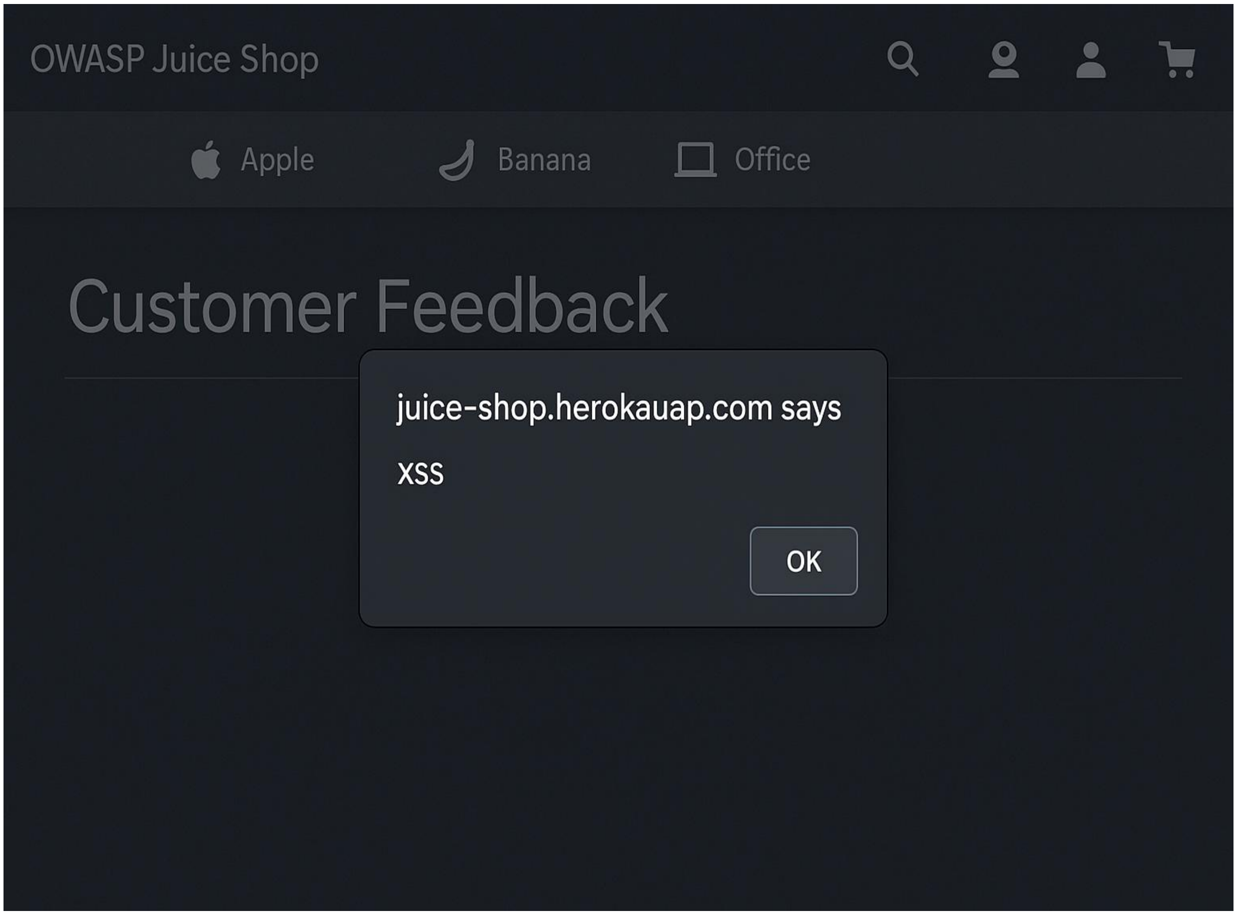
Email*: ' OR 1=1--

Password*: ABC

Step 2: Cross-Site Scripting (XSS)

- Opened the feedback form and inserted <script>alert('XSS')</script> in the input field.
- When the page loaded, an alert box popped up showing the text 'XSS'.

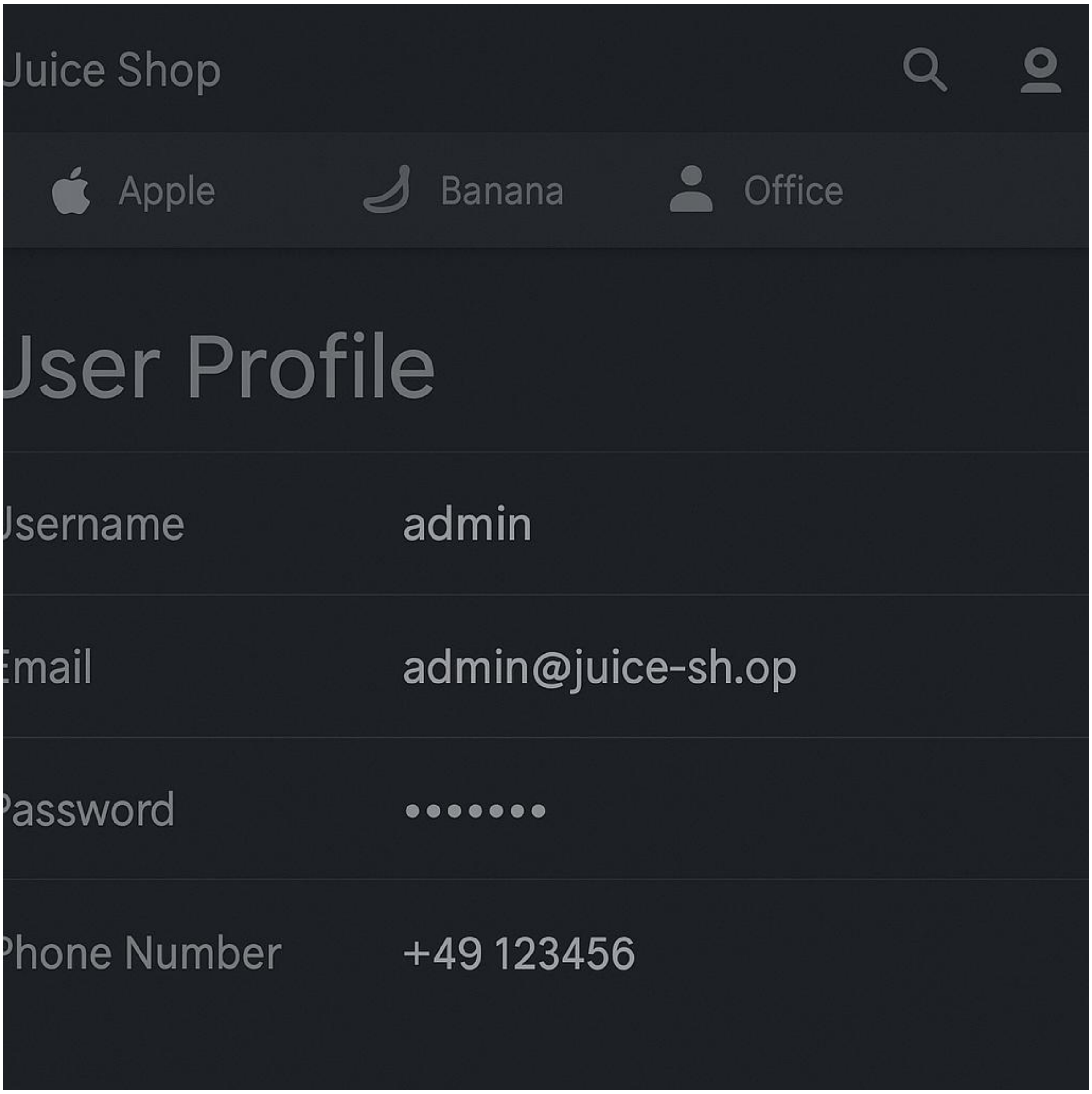
(Screenshot 2: XSS alert box)



juice-shop.herokauap.com says

XSS

Step 3: Broken Access Control
- Tried to open the admin page directly by typing /#/administration in the browser URL bar.
- Surprisingly, the admin page opened without proper login.
(Screenshot 3: Admin page access without login)



User Profile

Username	admin
Email	admin@juice-sh.op
Password	••••••••
Phone Number	+49 123456

4. Results & Discussion

Vulnerabilities Found:

1. SQL Injection
- Location: Login Page
- Payload: ' OR 1=1--
- Result: Login bypass
- Fix: Use parameterized queries

2. Cross-Site Scripting (XSS)
- Location: Feedback Form
- Payload: <script>alert('XSS')</script>
- Result: Alert appeared
- Fix: Validate and encode user inputs

3. Broken Access Control
- Location: Admin Page
- Method: Direct URL access
- Result: Admin page exposed
- Fix: Implement role-based access

5. Challenges Faced

As a beginner, I struggled at many points:

- Initially, none of my payloads worked because I didn’t know where to enter them.
- Some errors confused me, and I had to search for solutions online.
- Broken Access Control was tricky because I didn’t expect a URL to open directly.
- I had to retry tests many times before they worked.

6. Learning Outcome

This project gave me a clear idea of how hackers think. I learned:

- How SQL Injection bypasses authentication.
- How XSS can inject malicious scripts.
- The importance of sanitizing user input.
- Why access control is necessary.
- How to document findings in a professional way.

7. Conclusion & Future Work

The project helped me understand that even a small website can be hacked if developers do not secure it.

Future Work:

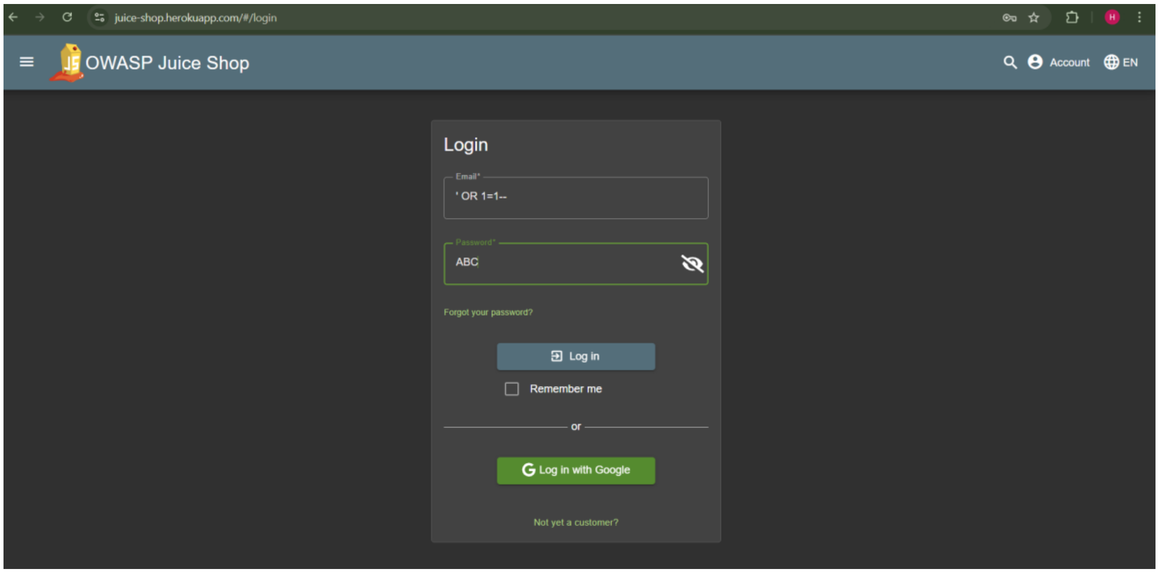
- Learn to use professional tools like Burp Suite and OWASP ZAP.
- Explore more vulnerabilities from OWASP Top 10.
- Learn how to fix vulnerabilities as a developer.

8. References

- OWASP Juice Shop Documentation
- OWASP Top 10 Security Risks
- TutorialsPoint – SQL Injection, XSS basics
- YouTube: 'SQL Injection explained for beginners'

9. Appendices (Screenshots)

1. Screenshot 1: SQL Injection Login Bypass
2. Screenshot 2: XSS Alert Box
3. Screenshot 3: Admin Page Access Without Login

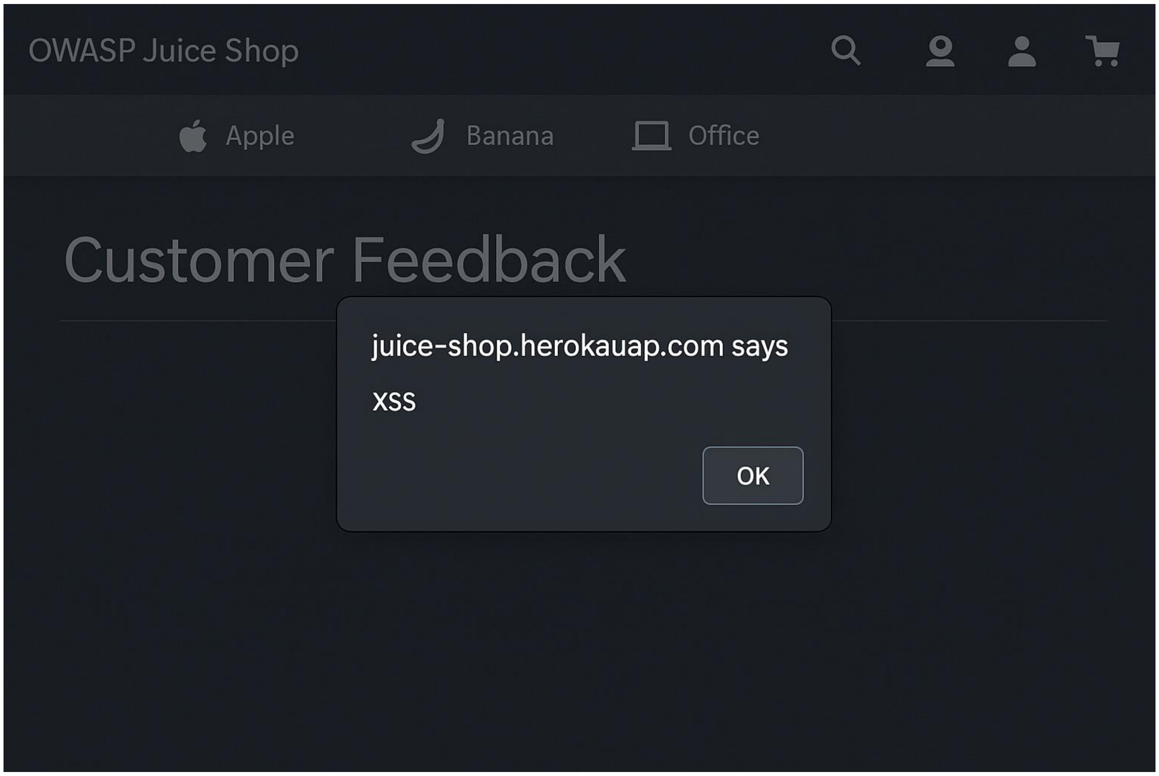


juice-shop.herokuapp.com/#/login

Login

Email*: ' OR 1=1--

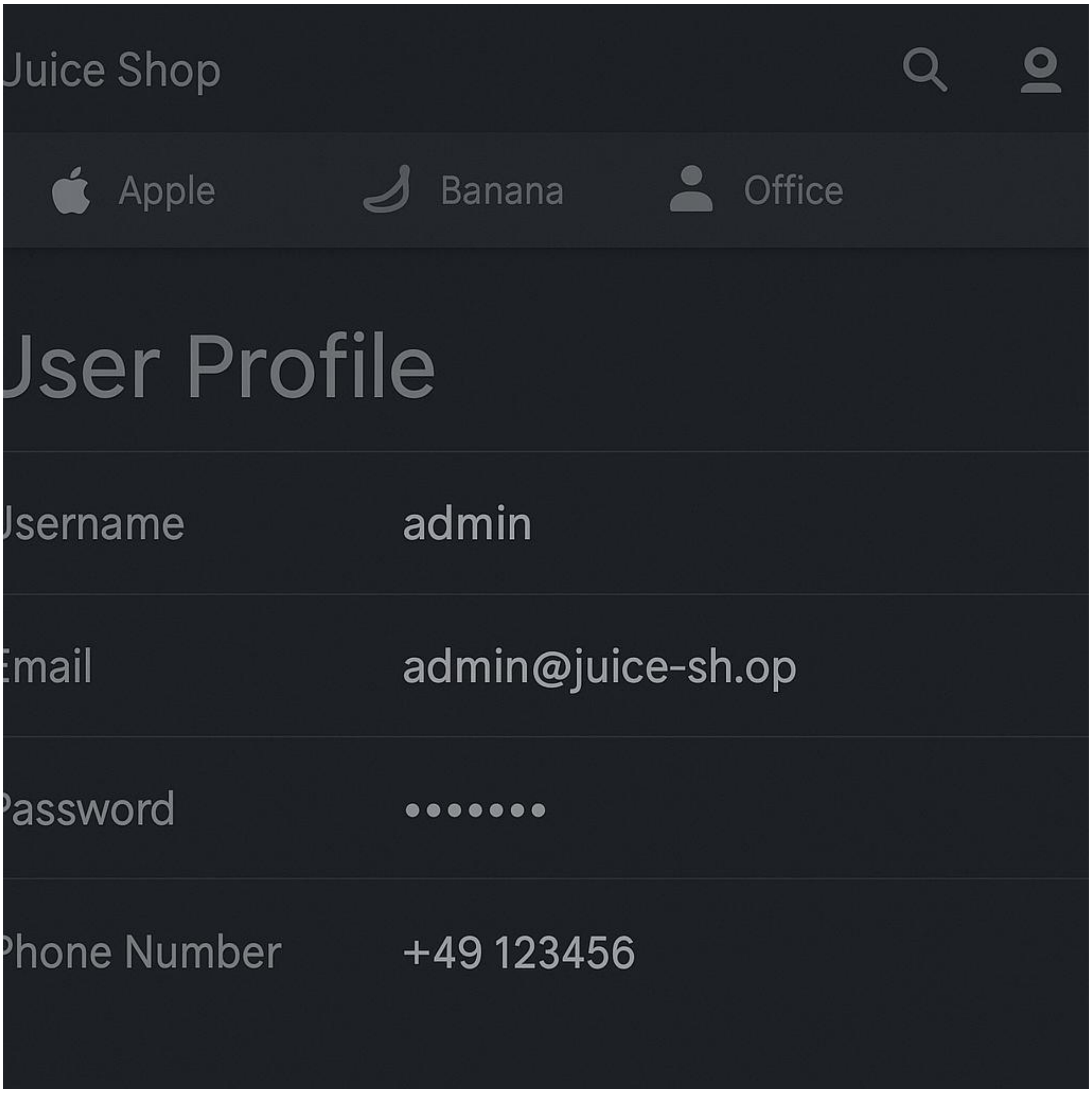
Password*: ABC



Customer Feedback

juice-shop.herokauap.com says

XSS



User Profile

Username	admin
Email	admin@juice-sh.op
Password	•••••••
Phone Number	+49 123456