# ISEC 400 Lab 3

For this assignment, you will be discovering vulnerabilities in software using an example system that was written in NodeJS. Although the source code language may not be familiar, the concepts and vulnerabilities in the study of application security are the same. This exercise will use JuiceShop, an application specifically designed to be vulnerable.

## Getting started

Use the GitpPod, HerokuApp, or live version of the JuiceShop platform you setup in the previous labs for this lab.

## Complete the following lessons

1. Using the “PWNING OWASP JUICE SHOP” book as your guide choose one challenge from the **“Injection”** area (pg. 88 for a list of these options, section 3.3) and complete it. Then, pick one challenge from the **“Cross Site Scripting (XSS)”** area (pg. 120 for a list of these options, section 3.10) and also complete that. The book will guide you through these, depending on the challenges you choose you may have to:
   1. View the source of documents in order to look for form elements
   2. Employ the “console” view of a browser (such as the Web Developer toolbar [generally F11 key shortcut]) and modify values that are being submitted in forms. Note that in Chrome, you can search for elements such as “form” or “option” by pressing Ctrl-F in the console window in Elements view.
2. Explain the challenges you chose, what you did for each challenge (take a screen shots along the way to paste it into your solution document) and create a write-up about your experience. Using screenshots here is recommended. A recommended free-for-personal-use screen capture tool is PicPick (<http://www.picpick.org/en/download_free>). This will let you capture just a region of your screen and easily paste the results.

## Submission instructions

Create a properly cited Word and formatted document for your submission that includes the steps you took to complete the challenges you chose, with screenshots along the way for each significant moment of finding. Submit the Word document to the dropbox for the lab.

For the injection area, the challenge I chose is the login Jim where we login with Jim’s user account. We start by looking through the products and finding the review left with Jim’s account as it will give us his email. We type in Jim’s email and then add ‘—to it to create the SQL injection so we can enter anything into the password spot. Then we hit login and we can now access Jim’s account.

A screenshot of a computer

Description automatically generated

Moving into the cross site scripting challenges, I chose the DOM XSS. We start by searching owasp in the search bar and looking at the owasp items. We test for HTML injection by changing our search value to <h1>owasp. After we confirm the addition of our html, we search, <script>alert(xss)</script>. As the script did not run we try the next XSS payload, <iframe src="javascript:alert(`xss`)"> which gives us an alert box that lets us know that the DOM XSS attack worked.

A screenshot of a computer

Description automatically generated

<https://3000-juiceshop-juiceshop-1zq4xwm666r.ws-us92.gitpod.io/#/search>