

OWASP Top 10: Lab

Student worksheet · OWASP Juice Shop

You will read another user's basket (A01) and find files and errors the site leaks (A02).

Open **Juice Shop**: `http://localhost:3000`. Open the **Score Board** too: `http://localhost:3000/#/score-board`. Each challenge you solve turns green there. First, register a free account (**Account** → **Login** → *Not yet a customer?*) and log in.

Rule: do this only on our own Juice Shop. The same steps on a real website, without written permission, are an offence under the IT Act.

1. View Basket (A01, 2 stars)

The server checks that you are logged in, but not that the basket is *yours*.

1. Add any product to your basket, then open **Your Basket** (top right).
2. Open the developer tools (`F12`) → **Application** (Firefox: **Storage**) → **Session Storage** → `http://localhost:3000`. Find the key `bid` (your basket number).
3. Double-click the value of `bid`, change it to `1`, and press Enter. Reload the basket page.
4. You now see someone else's basket (basket `1` is the admin's). The Score Board shows **View Basket** solved.

Why: the basket number travels from your browser, and the server loads whatever number you send without checking the owner. That is broken access control: being logged in is not the same as being allowed.

If stuck: No `bid` key: open **Your Basket** first. Basket did not change: reload after pressing Enter. Try other small numbers too.

2. Confidential Document (A02)

1. In the address bar, open `http://localhost:3000/ftp`. A list of files appears.
2. Click `acquisitions.md`. It opens a document marked "confidential". The Score Board shows **Confidential Document** solved.

Why: a server setting lists every file in the `/ftp` folder, so anyone can see and download files that were never meant to be public. No bug in the code, just an unsafe setting.

If stuck: The shop's home page opens instead of a file list: you typed `#/ftp`; the address is `/ftp`, without `#`.

3. Error Handling (A02)

1. Open `http://localhost:3000/rest/no-such-thing`.
2. You get a full error page showing the framework and version (`Express ^4.22.1`) and file paths inside the server. The Score Board shows **Error Handling** solved.

Why: a developer error page was left on in production, so every visitor sees internal details that help an attacker plan the next step. A real site should show a short, plain error message.

What you did

Three findings, two categories. In View Basket (A01) the server trusted a number from your browser. In Confidential Document and Error Handling (A02) safe-looking *settings* leaked files and internals.

The lessons: decide access on the server (not from the request), and turn off features and error pages you do not need in production.