

## Cibersegurança – Módulo 2

### Terceira série de exercícios

**Entrega:**

Relatório com evidências ou explicações pedidas em cada questão, e o URL da aplicação *juice shop* do grupo.

**❖ Preparação**

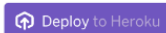
1. Pretende-se alojar a aplicação “Juice Shop” no serviço *cloud* Heroku. Este serviço corre aplicações web, escritas em diferentes linguagens, dentro de um ambiente de execução virtual designado *dyno*.

i. Crie uma conta gratuita no serviço Heroku e faça *login*

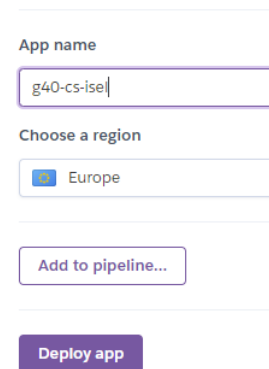
ii. Aceda ao repositório onde está o código fonte da aplicação, <https://github.com/bkimminich/juice-shop>, e use a opção “Deploy to Heroku”

Deploy on Heroku (free (\$0/month) dyno)

1. [Sign up to Heroku](#) and [log in to your account](#)
2. Click the button below and follow the instructions

 Deploy to Heroku

iii. Indique o nome da aplicação segundo a regra  $g<NN>-cs-isel$ , onde  $<NN>$  é o número do grupo (01, 02, 03, ...), e faça *deploy* da aplicação:



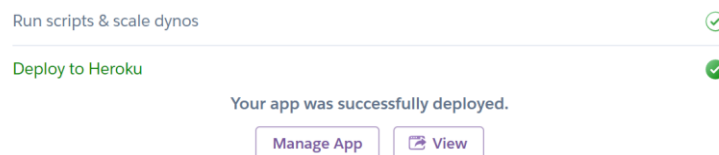
App name  
g40-cs-isel

Choose a region  
Europe

Add to pipeline...

Deploy app

iv. O processo demora alguns minutos. No final, consulte o seu *dashboard* com a opção “Manage App” ou siga para a aplicação escolhendo “View”:



Run scripts & scale dynos ✓

Deploy to Heroku ✓

Your app was successfully deployed.

Manage App View

A arquitetura da Juice Shop segue um desenho moderno de aplicações web, como exemplificado na Figura 1 (<https://owasp.org/www-project-juice-shop/>).

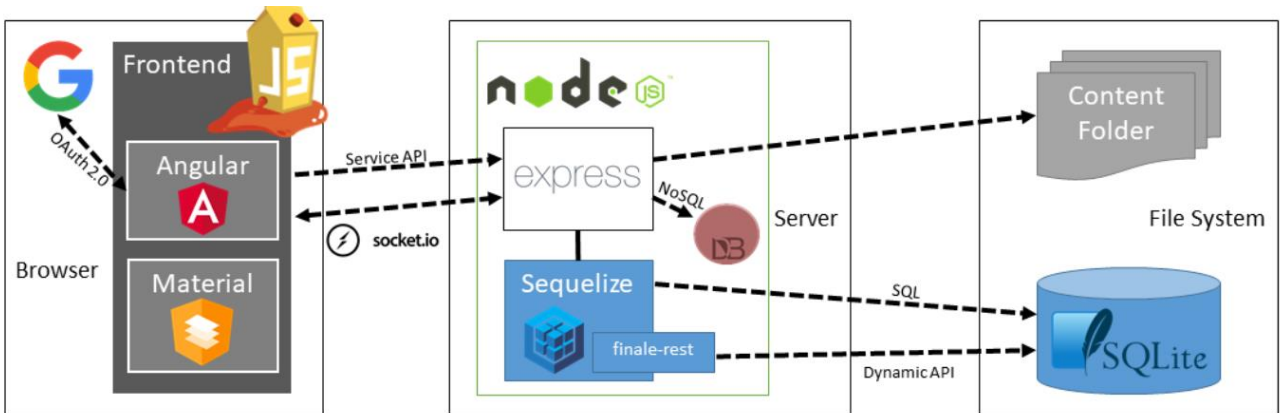


Figura 1: Arquitetura da Juice Shop

A aplicação tem várias vulnerabilidades propositadas as quais estão associadas a desafios que o utilizador pode ir resolvendo. Cada instância da aplicação mantém uma lista dos desafios já resolvidos, organizados pelo número de estrelas de cada desafio (de 1 a 6).

2. Instale e verifique o correto funcionamento do Zed Attack Proxy (ZAP) disponível aqui <https://owasp.org/www-project-zap/>:
  - i. Na opção “Quick start” escolha a opção “Manual Explore”, e indique o site <https://www.example.org>, com HUD ativo;
  - ii. Aceda à aplicação Juice Shop.

## ❖ Exercícios

0. Usando as ferramentas de programador do *browser* edite o título da página de entrada para “OWASP Juice Shop G<NN>”. Apresente o *screenshot* da modificação. O *refresh* à página mantém a alteração?
1. O código a correr no elemento “server” identificado na Figura 1 está escrito em Node.js, um ambiente de execução para *JavaScript* e cujas bibliotecas de dependências são geridas com a ferramenta npm (<https://docs.npmjs.com/about-npm/>). Instale o cliente de linha de comandos (*command line interface*) do serviço Heroku: <https://devcenter.heroku.com/articles/heroku-cli>. Ligue-se através da *command line interface* ao *dyno* onde corre a aplicação web:

```
$ heroku login
$ heroku ps:exec -a g<NN>-cs-isel
Verifique a versão do sistema operativo do dyno com o comando:

$ cat /etc/os-release
```

A Juice Shop tem vários problemas dos descritos no OWASP Top 10. Dentro do *dyno*, corra o comando de auditoria:

```
$ npm audit
```

- i. Enquadre as 2 vulnerabilidades críticas nos riscos identificados no OWASP Top 10.
  - ii. Quantas vulnerabilidades de XSS foram identificadas e em que biblioteca?
2. Use a funcionalidade de *spider* do ZAP e descubra o conteúdo do ficheiro escondido `acquisitions.md`. Apresente as primeiras 3 linhas do ficheiro. Esta questão corresponde ao desafio “Access a confidential document.” - <https://bkimminich.gitbooks.io/pwning-owasp-juice-shop/content/part2/sensitive-data-exposure.html#access-a-confidential-document>
3. Com as ferramentas de programador do browser (F12 no *chrome* ou *firefox*) descubra o caminho que dá acesso à lista de classificações - “score board”, presente num dos ficheiros *javascript* carregados pelo *frontend*. Apresente as ações realizadas. Esta questão corresponde ao desafio “Find the carefully hidden 'Score Board' page” - <https://pwning.owasp-juice.shop/part2/score-board.html>
4. Procure por produtos com a palavra “Lemon”. Repare na barra de endereços. Experimente alterar diretamente na barra de endereços o critério de pesquisa para “Lemon1”. Na página de resultados onde é inserido o texto do critério de pesquisa?
5. Resolva o desafio “DOM XSS”, injetando o texto `<iframe src="javascript:alert(`xss`)">` de maneira a que *browser* tenha de processar uma página com o texto injetado.
6. Descubra a password do utilizador administrador [admin@juice-sh.op](mailto:admin@juice-sh.op). A password começa por “admin” e tem um sufixo numérico de 3 algarismos. Mostre como pode através de fuzzing descobrir a *password* correta.
7. [extra] Mostre como resolver o desafio “Post some feedback in another users name” usando o proxy ZAP: <https://bkimminich.gitbooks.io/pwning-owasp-juice-shop/content/part2/broken-access-control.html#post-some-feedback-in-another-users-name>

8. [*extra*] Mostre como através de um ataque Cross-site scripting (XSS) e de engenharia social um atacante pode obter o *cookie* com nome “token” armazenado no browser da vítima.