University of South Australia

Assignment Specification

Course coordinator: Jakob Pennington

University of South Australia

Introduction

For this assignment, you will do a deep dive into a security vulnerability of your choosing, with opportunities to learn about a vulnerability in theory and in practice. You have been granted access to the codebase of a vulnerable web application, Guardian Store, which will serve as the basis for the assignment.

In this assignment, you will:

- Find and exploit your chosen security vulnerability.
- Research your vulnerability, building a deeper understanding beyond what you have learned in seminars and practicals.
- Implement a patch to resolve your vulnerability.
- Research, select and implement automated security testing that aims to identify occurrences of your chosen vulnerability.
- Investigate opportunities for defence-in-depth and secure development practices that could help identify and prevent similar vulnerabilities in future.
- Share the findings of your investigation in a report format.

Similar to the seminars and practicals in this course, the assignment is based on a scenario where you are an employee in the Guardian Protect workforce in the fictitious company Guardian. Your ability to produce deliverables which align with this scenario will impact your grade for the assessment. See the scenario section of this document for more details.

Assessment weighting: 30% of overall grade

Assignment type: Individual

Deliverables:

- Report (submitted via LearnOnline)
- Code (Submitted via pull-request on GitHub)

Word count: 2500 words (approximately)

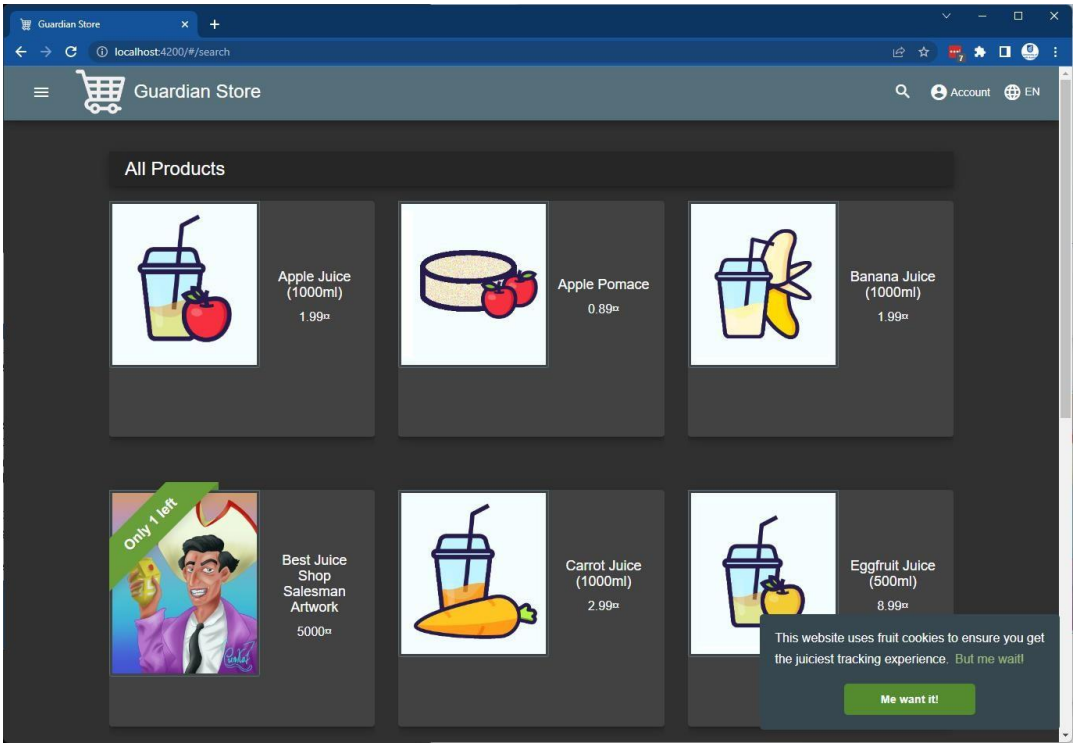
Due date: 30th October 2023

Scenario

Guardian Store is an online storefront where Guardian sells its products, accessories, service subscriptions and merchandise. The store was developed in the early days of Guardian when they were a small start-up, a time when rapid feature development was prioritised over quality and security. Guardian Store was not

developed with the same secure development practices used to develop their newer products, and no security reviews or testing was performed during the SDLC.

As part of their work to build a modern application security program, the Guardian Protect workforce contracted a local information security firm to perform a penetration test against Guardian Store. The objective of the penetration test was to understand the current level of risk in the application and to identify any security vulnerabilities that are present. The findings of the penetration test will be used by Guardian Protect to determine where they should focus their efforts.



The Guardian Store landing page.

The penetration test revealed a large number of security vulnerabilities, spanning several vulnerability classes, including:

- Broken authentication
- Broken authorisation
- Business logic flaws
- Cross-site request forgery
- Cross-site scripting
- Cryptography issues
- Insecure file upload
- NoSQL injection
- Sensitive data exposure
- SQL injection

Guardian Protect must now respond to the penetration test. Due to the high volume of vulnerabilities, each member of Guardian Protect is tasked with selecting and addressing **one vulnerability type** within Guardian Store.

Guardian's objective is to not only resolve specific instances of security vulnerabilities; their aim is to systematically select a vulnerability type and eliminate it from all software products across the organisation.

Objective

The objective of this assignment is to help Guardian meet their goal of eliminating your chosen vulnerability type entirely from their suite of products. In order to achieve this goal, you will need to deeply understand your chosen vulnerability type, so you can educate other developers about the vulnerability, how it occurs and how to prevent it, and you will need to provide guidance to Guardian about how they can uplift their secure development practices to prevent the vulnerability from occurring again.

Your grade for this assessment will be based on how well your deliverables meet this objective.

About Guardian Store

Guardian Store is a lightly skinned version of OWASP Juice Shop, which according to their website is "probably the most modern and sophisticated insecure web application". There are plenty of resources, blogs and documentation about OWASP Juice Shop online. You are welcome (and I encourage you) to use these resources, particularly in the early stages of the assignment.

Website: https://owasp.org/www-project-juice-shop/

Documentation: https://pwning.owasp-juice.shop/

GitHub: https://github.com/UniSA-Secure-Software-Development/unisa-guardian-store

Front end: Angular 14 (TypeScript, HTML, CSS)

Back end: Node.js Express (TypeScript)

University of South Australia

Assignment Task

1. **Find and exploit a vulnerability**
   Begin by selecting a vulnerability type that you would like to focus on for your assignment. For example, you may choose to focus on Broken Access Controls, Cross-Site Scripting or SQL Injection. You may choose any vulnerability type that exists within Guardian Store.

   When selecting a vulnerability type, you may choose to:
   a. Select a vulnerability you are interested in, then try to find an instance of that vulnerability in Guardian Store.
   b. Try finding a vulnerability in Guardian Store, then select whichever vulnerability you find.
   c. Select one of the vulnerabilities listed within the OWASP Juice Shop documentation.
   https://pwning.owasp-juice.shop/part2/

   Once you have selected a vulnerability, exploit it. Make sure you understand what the vulnerability is, how the exploit works, and what the impact would be if a malicious user exploited the vulnerability.

   Milestone 1: *Proceed to the next step once you have selected a vulnerability, exploited the vulnerability within Guardian Store, and understand the instance of the vulnerability within Guardian Store.*

2. **Review the vulnerable code**
   Locate the vulnerability in the source code, and review the code to understand how the vulnerability was introduced. Based on this understanding, see if you can find any other instances of the vulnerability in the codebase. Ensure you understand the root cause of the vulnerability, as this will be necessary to implement an effective fix.

   Milestone 2: *Proceed to the next step once you are confident you understand the root cause of the vulnerability in code, that you have found all instances of the vulnerability within Guardian Store, and you could explain the cause of the vulnerability to a classmate.*

3. **Research the vulnerability and remediation approaches**
   Next, research your chosen vulnerability to build a deeper understanding. You should understand what the vulnerability is, how it occurs in software, and what a malicious actor could do if they exploit the vulnerability.

   Research approaches to remediating the vulnerability in the languages and frameworks used to build Guardian Store. Investigate options to apply defence in depth to provide additional layers of protection beyond just fixing

the vulnerability. Also review secure development practices covered throughout the course which could help Guardian prevent or detect your chosen vulnerability type earlier in the SDLC.

Tip: Make sure you collect references during this step, as you will need them when it comes time to write your report.

Milestone 3: *Proceed to the next step when you have a good sound understanding of your chosen vulnerability type and how you plan to fix the vulnerability within Guardian Store.*

4. **Implement a patch**
   Next, implement a patch to resolve your vulnerability. At a minimum, implement a single fix to the vulnerability that you exploited, however if you found additional instances of the vulnerability in Step 2, implement patches for those as well.

   Perform testing against your patched code to ensure that the vulnerability has been resolved, and that the application still functions as expected for the "happy path" (i.e., make sure you have not broken the application while implementing your patch).

   Milestone 4: *Proceed to the next step when you can demonstrate that the vulnerability has been patched and that the application still works when on the happy path.*

5. **Implement automated security testing**
   Next, review options for automated security testing that specifically targets your chosen vulnerability type. Based on the CI / CD pipeline provided in the Week 12 practical, implement your selected automated testing to automatically scan the codebase each time a pull-request is raised.

   Note: If you reach this stage of the assignment before Week 12, I recommend you skip this step and get started on your report. You can come back to this step after you have completed the Week 12 practical.

6. **Write your report**
   Finally, write a report that covers everything you have learned throughout the assignment. The report should be targeted fellow software developers within Guardian to educate them about your chosen vulnerability.

At a minimum, your report should include:

- **Executive summary**
  Summarises purpose and key outcomes of the project, aimed at an executive level.
- **Introduction**
  Introduce the report, including the purpose of the report and a summary of what it includes.
- **Vulnerability Type:** A summary of your chosen vulnerability type, based on the research performed in Step 3. This should clearly explain the vulnerability type to somebody who is not familiar with secure coding, with the intent to educate other developers at Guardian.
- **Vulnerability Instance:** A detailed explanation of the specific vulnerability instance within Guardian Store, based on what you learned in Step 1 and Step 2. Your explanation should include how you found and exploited the vulnerability, and should refer to the source code that caused the vulnerability.
- **Remediation:** An explanation of how you fixed the vulnerability, including an overview of the software development patterns and techniques used to fix the vulnerability. This should contain sufficient detail that another developer could apply a similar fix, should they find a similar vulnerability in another application.
- **Automated security testing:** Provide a summary of the automated security testing that you have implemented, including why you selected this tool, how it works, and the output it provides.
- **Recommendations:** Provide any further recommendations to Guardian that may assist them in identifying and preventing similar vulnerabilities in future. Examples may include defence-in-depth measures and recommended improvements to the secure development lifecycle.
- **Conclusion:** Conclude the report, summarising the key outcomes and recommendations provided within the report.
- **References:** You *must* provide references to reputable resources throughout the assignment. Referencing should be in the Harvard referencing style.

  Refer to the following link for help with referencing:
  https://lo.unisa.edu.au/course/view.php?id=3839

University of South Australia

Assessment

Component	Weighting
Executive Summary	10%
Vulnerability Type	15%
Vulnerability Instance	15%
Remediation - Implementation in code (7.5%) - Reporting (7.5%)	15%
Automated Security Testing - Implementation in code (7.5%) - Reporting (7.5%)	15%
Recommendations	20%
Language, structure, formatting, referencing	10%

Dates

Component	Week	Date
Assessment commences	Week 7	4th September 2023
Assignment work in practical • Opportunity to ask questions and resolve issues • Aim to complete milestone 4 and have a draft of the report complete by end of Week 11.	Week 11	16th October 2023
Security testing practical • Required to complete milestone 5	Week 12	23rd October 2023
Assignment due	Week 13	30th October 2023

Tips

- **Do not wait until after the mid-semester break to start the assignment.** Get the research and coding phases completed early, so you leave yourself enough time to write a solid report.
- **How should I structure my report?**
  You may structure your report any way you choose; however, starting with a heading per bullet point listed in the assignment task is a good place to start.

- **How do I get a good grade for language, structure, formatting, referencing?**

  Glad you asked. In addition to covering each of the requirements for the report listed in the assignment task, I would expect to see the following:

  - Cover page
  - Table of contents
  - Executive Summary
  - Introduction
  - Conclusion
  - Page numbers
  - Clearly structured headings and subheadings
  - Numbered headings and subheadings
  - Clear and concise language
  - No spelling or grammar errors (Please set dictionary to Australian English)
  - An appealing report template / theme

- **How strict is the word count? (2500 words)**

  I care more about how well you meet the objective and requirements set out in this document than how closely you can meet the word count. As such, treat the word count as an approximate target. You will not be penalised for falling short of or exceeding the word count; however, providing too little detail or not writing clearly and concisely *will* affect your grade.

University of South Australia

Setting Up

The Guardian Store project is configured to run in a similar way to how we've been running the Defensive Coding practical exercises, using VS Code and Docker. These instructions assume you have completed the Course Preparation instructions on the machine you intend to use to complete the assignment.

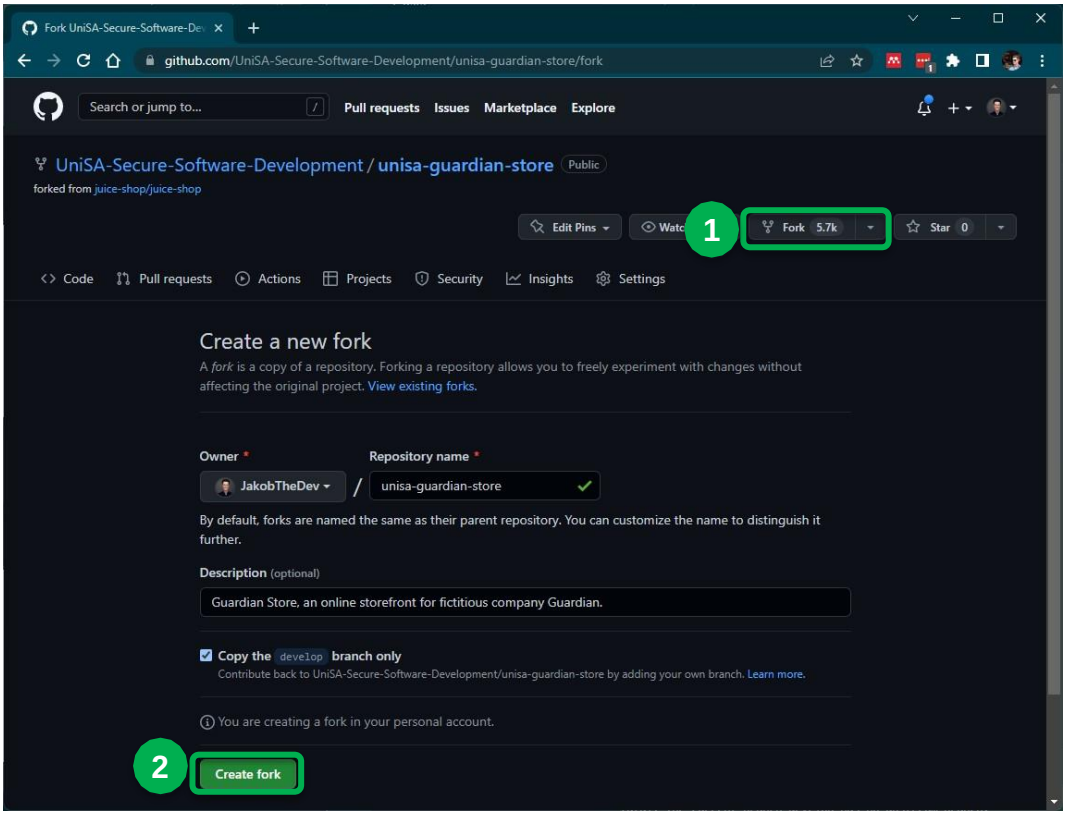
1. **Fork the Guardian Store code repository**

   First, you will need to fork a copy of the Guardian Store code repository into your personal GitHub account.

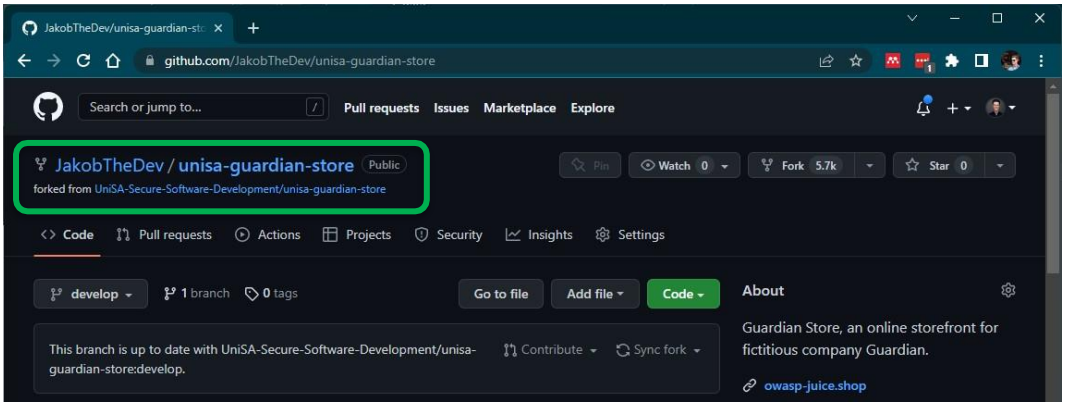
   a. Navigate to the Guardian Store code repository on GitHub. https://github.com/UniSA-Secure-Software-Development/unisa-guardian-store

   b. In the top right of the screen, click the Fork button.

   c. On the following page, leave the default values and click Create Fork.



d. You should now have your own forked copy of the unisa-guardian-store repository on GitHub. You can verify this by looking at the top right of the screen.



Note: This is the code repository you should use for your practicals. Please, **do not clone or work directly on the UniSA-Secure-Software-Development/unisa-guardian-store** code repository.

2. Clone the repository

Next, clone your fork of the Guardian Care repository to your machine.

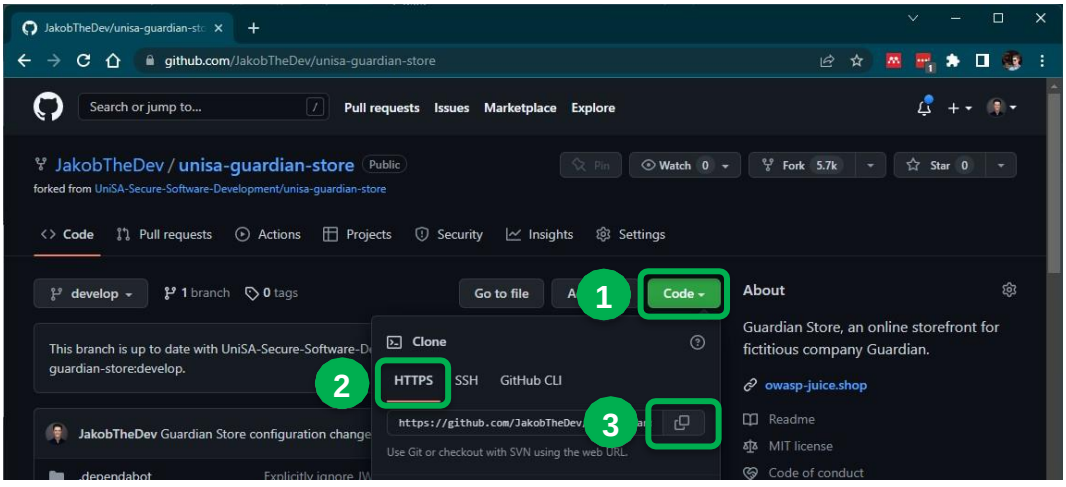
a. Open your terminal of choice.

b. Navigate to a directory where you wish to clone the repository.

```
$ cd /path/to/repository
```

c. In your browser, navigate to your forked copy of the unisa-guardian-care repository.

d. In the top right, click Code, then click HTTPS and copy the HTTPS clone URL.



e. In the terminal, run the following command.

Note: You may be prompted for your GitHub credentials.

```
$ git clone PASTE_GIT_URL_HERE
Cloning into 'unisa-guardian-store'...
remote: Enumerating objects: 113014, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (15/15), done.
Receiving objects: 100% (113014/113014), 111.96 MiB
Resolving deltas: 100% (88356/88356), done.
Updating files: 100% (999/999), done.
```

3. Set up VS Code

The next step is to set up Visual Studio Code so that we can write, run and debug code.

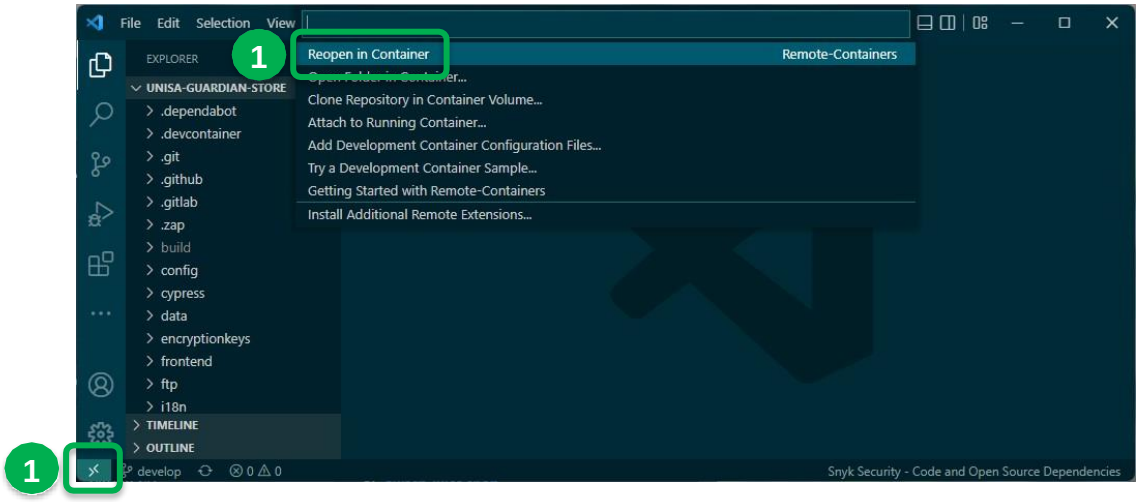
a. Open Visual Studio Code.

b. Click File > Open Folder, then select the folder containing the Guardian Care codebase.

Note: It's important that you select the root directory of the codebase. It should be called unisa-guardian-store and contain the config, data and encryptionkeys folders.

c. Next, we'll use the Remote Containers extension to open the codebase within a running Docker container that has all the necessary requirements installed.

In the bottom left, click the Remote Containers button (looks like ><), then select Reopen in Container in the dropdown.



The first time you open the folder in a container, Visual Studio Code will download and build the Docker image required to run the project and install software dependencies. This may take some time, so be patient.

4. Run Guardian Store

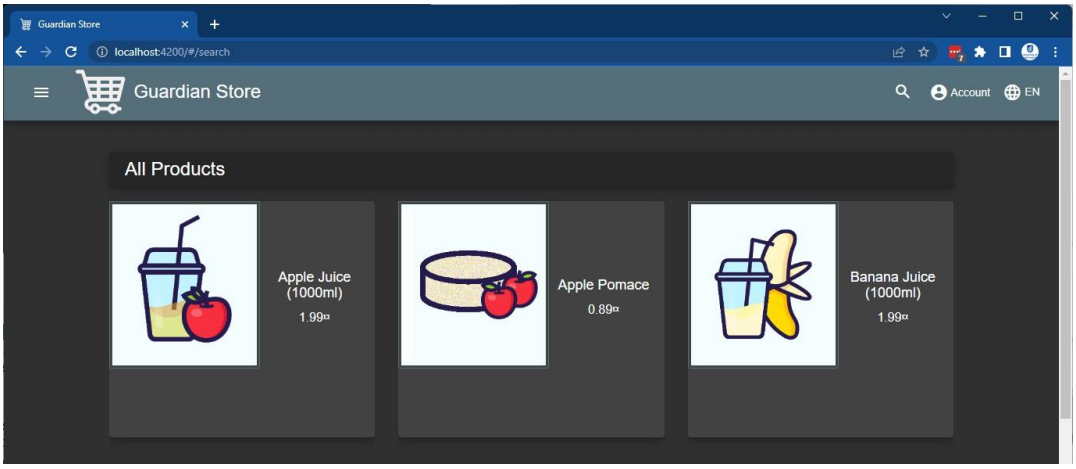
The final step is to run the project. There is a helper script set up to run the front-end and back-end of the application in watch mode, meaning any code changes you make while completing the assignment will cause the application to rebuild and the browser to refresh, showing your changed code.

a. In the terminal pane, run the following command:

```
$ npm run serve
```

When the program is running, you should see the message " Angular Live Development Server is listening on localhost:4200 ".

b. Open the browser to http://localhost:4200



You are now ready to begin the assignment. Good luck, reach out if you need any help, and have fun with it!

University of South Australia

Submission

There are two deliverables you will need to submit for the assignment:

- Your report
- Your code changes

Report submission

Your report can be submitted via the submission box in LearnOnline. The submission box can be found at Content > Assessment > Assignment.

Please note: You must submit your report as a single PDF document. No other submission format will be accepted.

Code submission

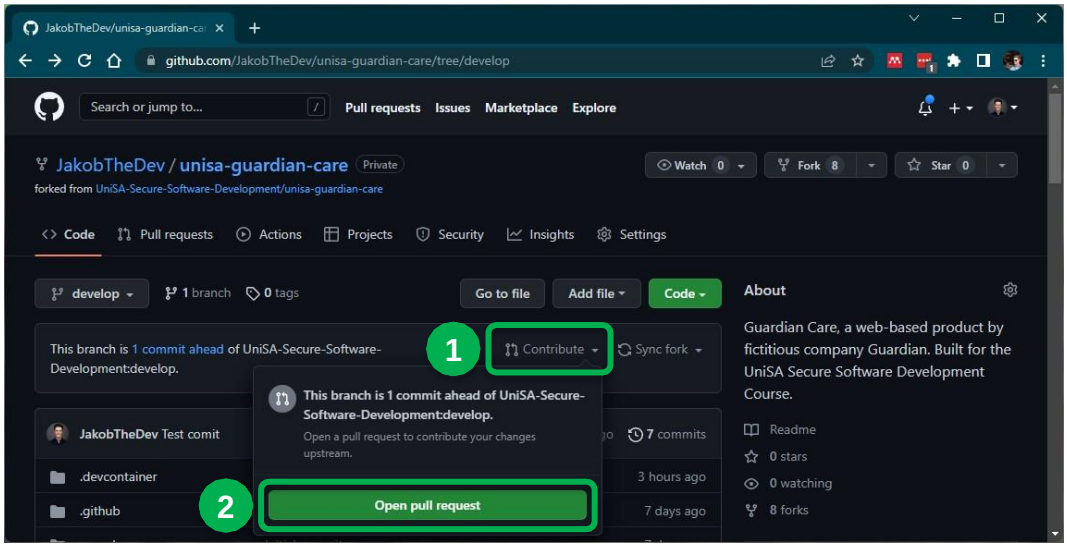
1. Commit your code changes by running the following commands in your terminal from the root directory of the git repository:

```
git add .
git commit -m "Completed assignment submission"
```

2. Push your changes to GitHub.

```
git push
```

3. Open GitHub in the browser.
4. Start a pull request by clicking "Contribute", then "Open pull request".



5. Fill in your details in the pull-request template, then click "Create pull request".