Section: PUBP

Web Application Penetration Testing Course For Developers

Zach Steele

Problem Statement: In this section, new text is blue for the professors convenience

There is a large difference in priorities and skillset between penetration testers and code developers, leading to a troubling amount of insecure web applications and difficulty communicating vulnerabilities to the application teams. While developers are constantly told to follow secure coding practices, oftentimes even utilizing security focused development frameworks, they rarely prioritize secure coding practices. (Secure Code Warrior, 2022) (Perry et al., 2023) prove that insecure code is going to become even more common as AI code assistants become more popular because developers trust its ability to write secure code. This problem will continue to grow if developers do not start taking cyber security and secure coding practices more seriously. In one study, 68% of participants stated that they only did secure code training because of a compliance need or recent exploit. (Security Journey, 2024)

Personally, I believe that developers do not take more cyber security training programs largely because of the gap in cyber security training programs. These courses and programs fall into two categories based on my experiences as a penetration tester responsible for teaching application teams how to test and remediate vulnerabilities. First, developer focused resources are too high level to be helpful. These courses may teach the basics of specific misconfigurations or exploits, but they do not provide many opportunities for developers to walk away with knowledge that can be applied to their current applications. In one study, developers state that training is not hands on, and/or it is taught in a vacuum. (Secure Code Warrior, 2022) Second, resources targeted towards penetration testers, meaning the course is extremely in depth, requires specialized tools, and a massive time investment from the developer. As an example, PortSwigger's Cross-Site Scripting section includes 30 labs with each solution video ranging from 50 seconds to over 20 minutes. If a developer wanted to learn the risks, remediation, and how to exploit XSS at a basic level, they would have to parse through multiple pages of documentation and choose between 50 second videos with zero context/explanations or 20+ minute videos explaining testing methodologies and edge cases throughout. Lastly, these walkthrough videos do not use tools that are available to developers in most corporate environments. (PortSwigger, 2025) From my own experience researching this topic, I often find these courses do not cover a wide range of topics. Generally, these courses are aimed at teaching exploitation, but do not cover policy based findings or misconfigurations, such as response header infractions. Please see (**Appendix Figure 1**) showing a summary of my own opinions and analysis after completing other courses that are publicly available. Please note that this is a high level view of these courses and is not intended to be an in-depth review of each course. The purpose is not to disparage other courses, but to instead point out gaps or audience/tool differences that I hope to fill with my solution.

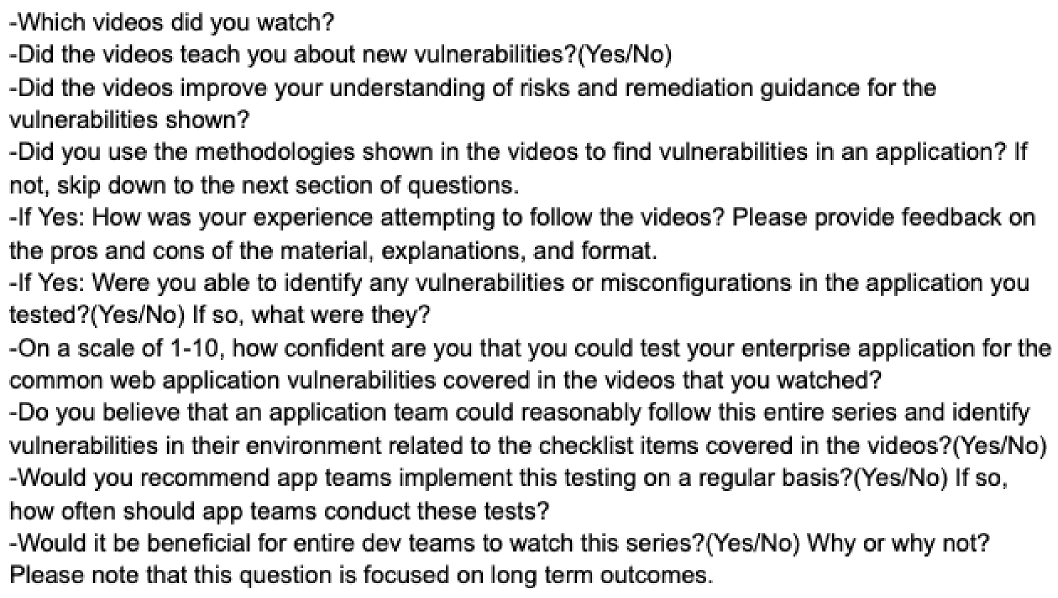
I believe that this pushes away most developers that were willing to commit a small amount of time to upskilling their cyber security knowledge. According to one study, 44% of respondents stated that the reason for not taking secure coding courses is that they were not aware of a good course and another 44% stated that they did not have the time. (Robinson & Russo, 2024) However, it is important to note that the solution described below may not solve the issues covered in this 88% of respondents because time constraints and course quality may be subjective. Ultimately, I believe that this gap in training material is the reason we continue to see an increase in vulnerabilities every year, even though most developers receive annual penetration tests on their applications and secure coding training. (Tribbey & Winterfeld, 2023) If application teams were capable of testing their environments and implemented security checks on a regular basis, we would most likely see a decrease in the 31% of critical vulnerabilities and 52% of lower priority vulnerabilities that go unremediated. (Lamar, 2025)

Solution Statement: In this section, new text is blue for the professors convenience

This project will teach developers to test their own applications for vulnerabilities via a training course dedicated to quickly learning penetration testing tactics for common vulnerabilities in order to increase the security of modern web applications.

Specifically, I plan on teaching developers how to test their applications from a similar set of material that penetration testers use to conduct assessments. Most web application penetration testers utilize a checklist with a set of vulnerabilities to test, payload lists, and an intercepting proxy tool to bypass client side controls. My goal is to use a similar framework that excludes complex attack types and unnecessary overhead. My target audience is developers who are looking for a way to reproduce findings or test new code for vulnerabilities before pushing to production. However, I will be providing third party links and suggestions on how to learn more as we go through the checklist for the audience members that would like to take penetration testing to the next level.

Lastly, I would like to explain my methodology for evaluating the effectiveness of this solution. It is easy to express the need for this course when reviewing the widespread security flaws across the internet. As an example, only 2% of the top 1000 visited sites have a properly implemented Content-Security-Policy (CSP). (Pacheco, 2020) In a few minutes, we could teach developers how to build a strong CSP to prevent exploitation of cross-site scripting and framing attacks. However, measuring the full impact and objective results of this course would require a set of developers who plan to complete the whole course and a much larger time window to analyze actions taken based on the material. Most firms would not allow developers to share vulnerabilities with external parties due to reputational risk, causing some limitations in our ability to test the effectiveness of this course on true corporate applications. Since we do not have enough time in this class to both complete the penetration testing course and test its effectiveness from start to finish, I will ask developers with varying ranges of security knowledge to review the material and provide feedback. If they have time, I will encourage them to test the material on their own applications or do specific, non-malicious, passive checks on public applications. This feedback will include how effective they think the course will be, and their level of confidence checking these items in their own applications. Additionally, since part of this course is based on changing the behavior of application teams, I will be asking the individuals what action items they plan on implementing into their own teams based on the course. Application owners can provide feedback explaining the value this course adds compared to current solutions in their own environments, such as company required training and code review checklists. This will give us perspective from the full range between security professionals and developers with no security background. Hopefully, this will result in a clear picture of the potential impact the complete course will have on a true audience. The figure below shows multiple survey questions in a bulleted format. Please visit my GitHub repo to view the entire document in its original format.



-Which videos did you watch?
-Did the videos teach you about new vulnerabilities?(Yes/No)
-Did the videos improve your understanding of risks and remediation guidance for the vulnerabilities shown?
-Did you use the methodologies shown in the videos to find vulnerabilities in an application? If not, skip down to the next section of questions.
-If Yes: How was your experience attempting to follow the videos? Please provide feedback on the pros and cons of the material, explanations, and format.
-If Yes: Were you able to identify any vulnerabilities or misconfigurations in the application you tested?(Yes/No) If so, what were they?
-On a scale of 1-10, how confident are you that you could test your enterprise application for the common web application vulnerabilities covered in the videos that you watched?
-Do you believe that an application team could reasonably follow this entire series and identify vulnerabilities in their environment related to the checklist items covered in the videos?(Yes/No)
-Would you recommend app teams implement this testing on a regular basis?(Yes/No) If so, how often should app teams conduct these tests?
-Would it be beneficial for entire dev teams to watch this series?(Yes/No) Why or why not? Please note that this question is focused on long term outcomes.

Figure 2: Evaluation Survey, abridged format (Steele, 2025)

Completed Tasks (Last 2 Week):

- Completed the checklist items and GitHub documentation for the first three checklist sections(16 checks). Please visit the repo to view all work papers: https://github.gatech.edu/zsteele3/ZS6727Summer2025
- Next, I continued to add notes and outlines for the video series to prepare content throughout the videos.
- Afterwards, I created a playlist for the training course, prepared, filmed, and edited the first 5 videos. This resulted in 75 minutes of video training material covering 15+ checklist items. To see the current course video series with detailed video descriptions and timelines, visit: https://www.youtube.com/watch?v=oNDEU_uNtzI&list=PLqPCUirqsN_x_seY51DiivfV-CjgJDKUp
- Created a survey for developers, security professionals, and app team managers to express their feedback and need(or lack thereof) for this course.
- Created an excel sheet giving my personal opinions on the current web app pentesting/secure coding courses online.

Tasks for the Next Project Report:

Over the next two weeks, I will continue to build out the payloads in GitHub and complete the checklist items through OS Command injection. Additionally, I will film videos for all payload required checks, ranging from cross-site scripting(XSS) to OS Command Injection. Next, I will finish building out my review of current solutions. Lastly, I will send out initial surveys asking people to watch the course videos and test their applications if they have the capability.

Questions I have or Issues I'm running into:

During the last two weeks I have not run into any issues. Preparing, filming, and editing videos is taking longer than expected, but that will not result in any missing material or loss in quality.

Methodology Paragraph Summary:

During the first few weeks, I will be building the structure of all required documents and creating the video order to ensure I have a pathway to follow with a clear understanding of the final outcome. Afterwards, I will spend a few hours preparing and creating the documentation required for each video in order. For example, when it is time to film the SQL injection video, I will first build out the payload list, checklist, and find the exact labs and tools that will best demonstrate how to test for SQL injection. This methodology will provide a clear path forward after every action item and will make it easy to see the work completed each week along with what is coming in the following weeks. It is important to note that some videos and deliverables may need to be completed out of order due to required resources or time to complete. For example, the introductory video cannot be completed before building out the course materials because that video will display the completed checklist and GitHub to give the viewer an idea of what we will be doing for the remainder of the course. As the course gets built out, I will be sending out a survey to developers, penetration testers, and other technology professionals to see how they have utilized parts of the course. More information will be provided in the evaluation section.

Timeline:

Week #	Description of Task	Status
W1	Installed VM, required tools and test apps.	Completed
W1	Created checklist template and first section.	Completed
W1/W2	Created GitHub and payloads page outline. https://github.gatech.edu/zsteele3/ZS6727Summer2025	Completed
W2	Created the outline for the video series.	Completed
W2	Created YouTube channel and uploaded placeholder video.	Completed
W3/W4	Built out the response headers and cookies checklist and GitHub pages.	Completed
W3	Recorded video for continuous review, response headers, and cookie testing.	Completed
W4	Conducted research on secure coding practices and filmed Video 3 - Best coding practices/remediation.	Completed
W4/W5	Created course evaluation survey and excel sheet showing pros/cons of current solutions.	In Progress
W5/W6	Complete GitHub pages and checklist for all payload based exploits covered in the checklist.	Not Started
W5/W6	Film videos 7(XSS) through 11(OS Comm inj)	Not Started
W5/W6	Reach out to developers for evaluation of material up to this point.	Not Started
W5/W6	Re-evaluate checklist items based on studies of most common vulnerabilities.	Not Started
W7/W8	Complete videos 11(login pages) to Video 16(Outro).	Not Started
W7/W8	Make changes on past material based on developer feedback.	Not Started
W7/W8	Film optional videos if time permits	Not Started
W9	Review entire project for places that may need improvement to make the series more cohesive.	Not Started

Evaluation: This section was completely re-written based on feedback from PR 1

I plan to evaluate the success and effectiveness of this course by surveying multiple developers, application owners, and penetration testers with varying levels of development and cyber security knowledge. The purpose of this survey will be to gather feedback to decide on future additional content, issues with past content, confidence in the material, action items that they plan on taking in their own teams, and hands-on practice with the course subjects to identify vulnerabilities within test environments. Most developers would need approval before testing their code for some of the vulnerabilities covered in the course, and most likely could not share the results with us. However, all developers are empowered to test the material on training applications or complete non-malicious passive checks on public facing applications. The results of this survey should display a clear improvement in the individuals' understanding of the vulnerabilities and ability to test their own applications based on the videos that they watched.

Report Outline: This entire section was empty in PR 1

Introduction

Introduce the problem and solution at a high level.

What is the problem/why is this a problem?
problem statement at the top of the progress report.

My Solution
Solution statement from the top of this document, plus an in depth explanation of the course.

Efficacy/evaluation
Explain the goals of our survey, the audience that received the survey, and summarize/visualize the results to show if/how this course worked and what items could be improved in future work.

Limitations
Describe the limitations of our course.

Future Work
Implement survey feedback, create flow charts to better step through the different types of payloads, include a wider range of tests.

Conclusion
Summarize what was covered in the entire paper, and explain the end result.
References
Appendix

References:

Specific penetration testing payloads and remediation guidance will be linked throughout the GitHub repo and checklist resources section. The references below are all utilized in this document.

Secure code warrior survey finds 86% of developers do not view application security as a top priority. Secure Code Warrior. (Apr 05, 2022). https://www.securecodewarrior.com/press-releases/secure-code-warrior-survey-finds-86-of-developers-do-not-view-application-security-as-a-top-priority

A Study on Secure Coding Training. Security Journey. (2024, January). https://www.securityjourney.com/hubfs/SJ_StudyonSecureCodingTraining24_011624.pdf

Tribbey, B., & Winterfeld, S. (2023, April 18). *Slipping through the security gaps: The rise of application and API attacks* | Akamai. Akamai. https://www.akamai.com/blog/security-research/the-rise-of-application-and-api-attacks

Pacheco, P. (2020, September 3). *Content security policy limits dangerous activity... so why isn't everyone doing it?* Bitsight. https://www.bitsight.com/blog/content-security-policy-limits-dangerous-activity-so-why-isnt-everyone-doing-it

Robinson, C., & Russo, D. (2024, June). Secure Software Development Education 2024 Survey. https://www.linuxfoundation.org/hubfs/LF%20Research/Secure_Software_Development_Education_2024_Survey.pdf?hsLang=en

Lamar, J. (2025, April 14). *Key takeaways from the State of Pentesting Report 2025*. Cobalt. https://www.cobalt.io/blog/key-takeaways-state-of-pentesting-report-2025

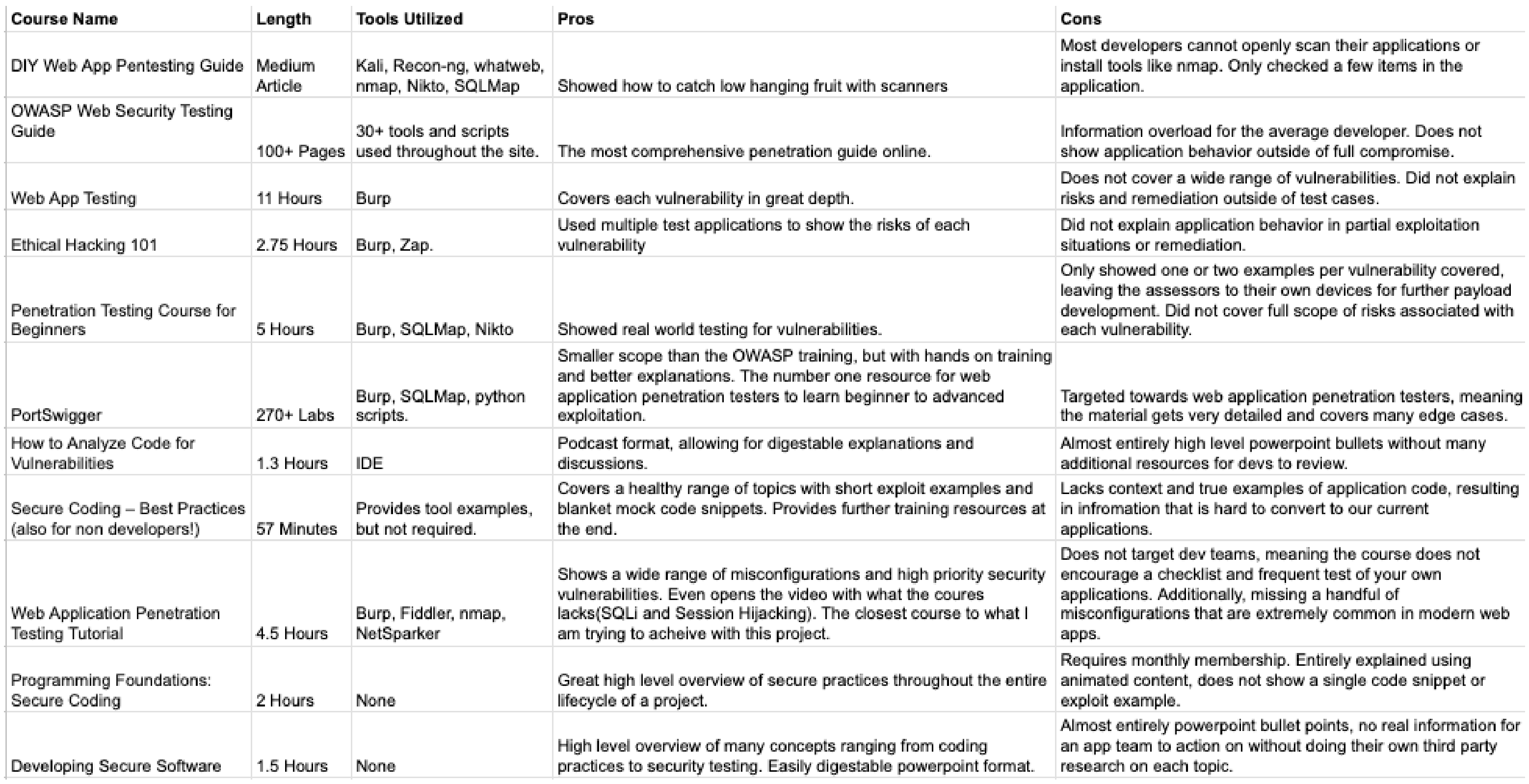
Steele, Z. (2025, June). ZS6727Summer2025. GitHub. https://github.gatech.edu/zsteele3/ZS6727Summer2025

Perry, N., Srivastava, M., Kumar, D., & Boneh, D. (2023, December 18). *Do users write more insecure code with ai assistants?*. arXiv.org. https://arxiv.org/abs/2211.03622

Why secure code training doesn't stack up (and what you can do about it) - blog. Secure Code Warrior. (2021, April 8). https://www.securecodewarrior.com/article/why-secure-code-training-doesnt-stack-up-and-what-you-can-do-about-it

What is cross-site scripting (XSS) and how to prevent it?: Web security academy. What is cross-site scripting (XSS) and how to prevent it? | Web Security Academy. (n.d.). https://portswigger.net/web-security/cross-site-scripting

Appendix



Course Name	Length	Tools Utilized	Pros	Cons
DIY Web App Pentesting Guide	Medium Article	Kali, Recon-ng, whatweb, nmap, Nikto, SQLMap	Showed how to catch low hanging fruit with scanners	Most developers cannot openly scan their applications or install tools like nmap. Only checked a few items in the application.
OWASP Web Security Testing Guide	100+ Pages	30+ tools and scripts used throughout the site.	The most comprehensive penetration guide online.	Information overload for the average developer. Does not show application behavior outside of full compromise.
Web App Testing	11 Hours	Burp	Covers each vulnerability in great depth.	Does not cover a wide range of vulnerabilities. Did not explain risks and remediation outside of test cases.
Ethical Hacking 101	2.75 Hours	Burp, Zap.	Used multiple test applications to show the risks of each vulnerability	Did not explain application behavior in partial exploitation situations or remediation.
Penetration Testing Course for Beginners	5 Hours	Burp, SQLMap, Nikto	Showed real world testing for vulnerabilities.	Only showed one or two examples per vulnerability covered, leaving the assessors to their own devices for further payload development. Did not cover full scope of risks associated with each vulnerability.
PortSwigger	270+ Labs	Burp, SQLMap, python scripts.	Smaller scope than the OWASP training, but with hands on training and better explanations. The number one resource for web application penetration testers to learn beginner to advanced exploitation.	Targeted towards web application penetration testers, meaning the material gets very detailed and covers many edge cases.
How to Analyze Code for Vulnerabilities	1.3 Hours	IDE	Podcast format, allowing for digestable explanations and discussions.	Almost entirely high level powerpoint bullets without many additional resources for devs to review.
Secure Coding – Best Practices (also for non developers!)	57 Minutes	Provides tool examples, but not required.	Covers a healthy range of topics with short exploit examples and blanket mock code snippets. Provides further training resources at the end.	Lacks context and true examples of application code, resulting in infromation that is hard to convert to our current applications.
Web Application Penetration Testing Tutorial	4.5 Hours	Burp, Fiddler, nmap, NetSparker	Shows a wide range of misconfigurations and high priority security vulnerabilities. Even opens the video with what the coures lacks(SQLi and Session Hijacking). The closest course to what I am trying to acheive with this project.	Does not target dev teams, meaning the course does not encourage a checklist and frequent test of your own applications. Additionally, missing a handful of misconfigurations that are extremely common in modern web apps.
Programming Foundations: Secure Coding	2 Hours	None	Great high level overview of secure practices throughout the entire lifecycle of a project.	Requires monthly membership. Entirely explained using animated content, does not show a single code snippet or exploit example.
Developing Secure Software	1.5 Hours	None	High level overview of many concepts ranging from coding practices to security testing. Easily digestable powerpoint format.	Almost entirely powerpoint bullet points, no real information for an app team to action on without doing their own third party research on each topic.

Figure 1: Course Comparison (Steele, 2025)

-Figure 1 location: https://github.gatech.edu/zsteele3/ZS6727Summer2025/blob/main/ClassSubmissions/Similar%20Courses.xlsx

-Web Application Penetration Testing Course: https://www.youtube.com/watch?v=oNDEU_uNtzI&list=PLqPCUirqsN_x_seY51DiivfV-CjgJDKUp

-Supporting documentation and course material: https://github.gatech.edu/zsteele3/ZS6727Summer2025

-Public GitHub: https://github.com/SchoolProjZS2025/GTSchoolProj/tree/main/capstoneProj Please note this is often 1 commit behind the official Georgia Tech GitHub repo.

-Survey Questions: https://github.com/SchoolProjZS2025/GTSchoolProj/blob/main/capstoneProj/Survey%20questions.docx