Mullumbimby High School
Assessment Task

Student Name:	

Task Name:	Secure Software Architecture		
Year:	12	Faculty:	TAS
Subject:	Software Engineering	Teacher:	Steedman
Date Assigned:	10/02/2026	Date Due:	21/03/2026
Weighting:	20%	Total Mark:	___ / 100

Outcomes to be Assessed:

SE-11-01 - describes methods used to plan, develop and engineer software solutions
SE-11-02 - explains how structural elements are used to develop programming code
SE-11-04 - applies safe and secure practices to collect, use and store data
SE-11-06 - applies tools and resources to design, develop, manage and evaluate software
SE-12-07 - designs, develop and implements safe and secure programming solutions
SE-11-08 - applies language structures to refine code
SE-11-09 - manages and documents the development of a software project

Task Description:

In this assessment task, your client, 'The Unsecure PWA Company', has engaged you as a software engineering security specialist to provide expert advice on the security and privacy of their application. This progressive web app is currently in the testing and debugging phase of the software development lifecycle.

Submission Instructions:

Hand in:

1. This completed workbook
2. Your code via GitHub Classroom

Absence / Misadventure Instructions:	Academic Integrity:	Acknowledgement:
All Stage 6 students have been issued with an Assessment Policy Handbook, which outlines the procedures relating to absence/illness/misadventure. This handbook can also be found on the school website. Students must submit an illness/misadventure form to the Head Teacher Senior Studies (Mrs Elliott) for foreseeable absences/misadventures before the task due date and immediately on return to school for unforeseeable absences / misadventures. The following penalties for late submission without an acceptable reason will apply: one day late - 20% deducted, two days late - 40% deducted. Three or more days late will be awarded zero.	Any instances of academic dishonesty may be interpreted as a non-attempt (failing grade). Please refer to the assessment policy booklet for more information. To ensure academic integrity, teachers may require students to submit their work using Google Docs or other platforms with version history tracking (or some other means of evidence of authorship). This enables teachers to verify that the content has been created by the student and not generated by an AI tool.	CLICK HERE OR SCAN

Overview

Your client, ‘The Unsecure PWA Company’, has engaged you as a software engineering security specialist to provide expert advice on the security and privacy of their application. This progressive web app is currently in the testing and debugging phase of the software development lifecycle and can be accessed here: **Unsecure Progressive Web App (PWA)**.

Requirements:

- run a range of security tests and scans, along with a white/grey/black box analysis of the application to identify as many security and privacy vulnerabilities as possible
- provide examples of secure code solutions, for example, the application programming interface (API) may include a web-based login component that provides a solution using a Python and Flask framework to allow for two-factor authentication (2FA) and cross-site request forgery (CSRF) protection and session management
- prepare a professionally written security report for your client that includes
    - an overview of your approach to the technical analysis
    - documentation of the scope of privacy and security issues, including:
        - security or privacy issues that cannot be mitigated by technical engineering solutions
        - security issues that must be tested in the production environment
        - security or privacy vulnerabilities you discovered and an impact assessment of each
        - recommendations relating to a security and privacy by design approach going forward for ‘The Unsecure PWA Company’
        - designing and developing implementations using HTML/CSS/JS/SQL/JSON/Python code and/or web content changes as required to patch each vulnerability you discover.

Submission Details:

- **Part A:** A complete security report.
- **Part B:** GitHub commits of implemented solutions or sample secure code.

1. Introduction

1.1 Overview of ‘The Unsecure PWA Company’.

The client, ‘The Unsecure PWA Company’, has engaged us as a software engineering security specialist to provide expert advice on the security and privacy of their application.

This progressive web app is currently in the testing and debugging phase of the software development lifecycle and can be accessed here: Unsecure PWA.

1.2 Importance of secure software architecture in today’s digital landscape.

Briefly explain to your client the importance of secure software architecture in today’s digital landscape.

1.3 Common vulnerabilities found in web applications

Describe to your client some of the common vulnerabilities found in web applications. This will help inform your cybersecurity audit of the ‘Unsecure PWA’.

Topic	Description
Structured query language (SQL) injection	
Cross-site scripting (XSS)	

Cross-site forgery request (CSRF)	
Invalid forwarding and redirecting	
Memory management	
Session mismanagement	
Broken authentication	
Race conditions	

1.4 Latest trends impacting the internet and cybersecurity.

Describe to your client some of the latest trends impacting the internet and cybersecurity. These trends will inform your cybersecurity audit of the 'Unsecure PWA'.

Topic	Description
Data protection	
Cyber attacks	
Static application security testing (SAST)	
Dynamic application security testing (DAST)	
Vulnerability assessment	
Penetration testing	
API security	

1.5 Lessons from Software Security Case Studies.

Research and explain to your client the impact of recent cybersecurity case studies, including lessons learned. Evaluate the social, ethical and legal issues and ramifications that affect people and enterprises.

Topic	Description and Impact	Lessons Learned
Equifax (2017)		
Veeam (2018)		
Facebook (2021)		
Log4j (2021)		

2. Benefits of Developing Secure Software

2.1 Data protection

2.1.1 Explanation of the significance of protecting sensitive data.

Describe to your client the benefits of protecting sensitive data from unauthorised access, tampering, or loss In this section, list and explain at least three benefits of data protection for the client. Use examples like protecting user credentials, securing financial transactions, preventing financial loss, protecting intellectual property, business continuity, enhancing reputation or complying with regulations.

Benefit	Explanation

2.1.2 Confidentiality, Integrity, and Availability

How do confidentiality, integrity, and availability (CIA) help achieve these benefits?

2.2 Minimising Cyber Attacks and Vulnerabilities

2.2.1 Benefits of minimising common cyber threats and vulnerabilities.

Describe to your client the benefits of minimising common cyber threats and vulnerabilities. In this section, list and explain at least three benefits such as preventing data breaches, protecting customer trust, complying with regulations, business continuity, cost savings, competitive advantage.

Benefit	Explanation

2.1.2 Strategies employed for minimising common cyber threats and vulnerabilities.

How do authentication, authorisation, and accountability (AAA) help achieve these benefits?

3. Software Development Steps to Develop Secure Code

For each phase of the software development lifecycle:
1. Outline strategies to strengthen the client's development process, focusing on security and privacy.
2. Explain how these strategies contribute to a secure architecture and uphold 'privacy by design' principles (e.g. proactive planning, embedding privacy into design, respecting user privacy).

Phases	**Description**
Requirements Definition	
Determining Specifications	
Design phase	
Development phase	
Integration	
Testing and debugging	
Installation	
Maintenance	

4. Influence of End Users on Secure Design Features

Your client is concerned about how the capabilities and experience of their end users influence the secure design of their software. Explain three key considerations they should address when designing secure features for their users.

For each consideration, include:

1. *A description of the user-related challenge or limitation.*
2. *A specific recommendation for how to address it in the design of the PWA.*
3. *An explanation of why this recommendation is effective and how it balances security with usability.*

Examples to discuss might include: *Understanding the User Base, Balancing Security and Usability, Role-Based Access Control (RBAC), Training and Awareness, Secure Defaults, Multi-Factor Authentication (MFA), Password Management and Strength Indicators, Accessible and Inclusive Security Features, Simplified Account Recovery Processes, Localisation of Security Prompts and Documentation.*

Issue	Explanation

5. Security Features Incorporated Into Software

Evaluate the "Unsecure PWA," identify flaws in each area (Data Protection, Security Measures, Privacy Protection, Regulatory Compliance, User Authentication and Authorisation), and propose actionable solutions.

Security feature	**Application**
Data protection	
Security measures	
Privacy protection	
Regulatory compliance	
User authentication and authorisation	

6. Cryptography and Sandboxing

Explain the role of cryptography in data security to the client, ‘The Unsecure PWA company’.

Explain how sandboxing limits exposure to vulnerabilities.

7. Testing and Evaluating Security and Resilience

Your client needs a comprehensive analysis of the vulnerabilities present in the Unsecure PWA.
In this section of the report:

Conduct Security Testing:

Test the Unsecure PWA for a range of vulnerabilities and issues, including but not limited to:

- Input handling (e.g., SQL injection, XSS)
- Privacy (e.g. lack of privacy policy, account management, user data encryption)
- Authentication and session management (e.g., weak passwords, session timeouts)
- API security and unnecessary endpoints
- File handling vulnerabilities (e.g., unsafe file uploads)
- Misconfigured headers and lack of security policies (e.g., CSP, CORS, HTTPS)
- Redirects, forwards, and race conditions

Document Vulnerabilities:

For each issue identified, provide:

- A brief description of the vulnerability and how you found it.
- The potential impact on the application and its users.
- An assessment of the severity (low, medium, high).

Vulnerability	Description	Impact	Severity

Vulnerability	Description	Impact	Severity

8. Design, Develop and Implement Secure Code

Now that you have identified vulnerabilities in the Unsecure PWA, your next task is to recommend and demonstrate solutions to address them. Your response will be assessed based on how effectively and practically you address the issues.

Addressing the Vulnerabilities:

Choose at least three vulnerabilities from Section 7.

Demonstrate how you would improve the PWA to mitigate each vulnerability, using one of the following approaches:

- **Exceptional Response:** Provide actual code fixes implemented in the Unsecure PWA.
- **Strong Response:** Provide detailed code snippets or samples demonstrating a fix for the issue.
- **Adequate Response:** Provide a clear and detailed explanation of what would need to be done to resolve the issue.

Key Areas to Focus On:

- Input validation, sanitisation, and error handling to protect against injection attacks.
- Secure authentication and session management practices.
- API security, including proper endpoint configurations.
- File and user data handling practices to prevent unauthorised access or tampering.
- Protection against XSS, CSRF, and other user-action vulnerabilities.
- Implementation of security headers (e.g., CSP, HTTPS, CORS).
- Mitigating redirects, forwards, and race conditions.

Vulnerability	Solution	Implementation (Code or Explanation)

Vulnerability	Solution	Implementation (Code or Explanation)

9. Conclusion

Summarise key findings and recommendations for enhancing security in 'The Unsecure PWA Company', including the enterprise benefits and how secure practices improve products and influence future developments.

10. References

List of resources and references used in the report.

Marking Rubric

Introduction (20 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Importance of Secure Software Architecture	Minimal explanation of the importance of secure software architecture.	Limited explanation with some relevance to today's digital landscape.	Clear and detailed explanation of its importance, including practical examples.	Comprehensive explanation with strong examples and discussion of real-world implications.	/3
Common Vulnerabilities	Minimal discussion of common vulnerabilities (e.g., SQL injection, XSS).	Limited explanation of vulnerabilities with few examples.	Clear identification of common vulnerabilities with relevant examples and descriptions.	Comprehensive identification and analysis of vulnerabilities, including impacts and mitigation approaches.	/6
Latest Trends in Cybersecurity	Minimal mention of cybersecurity trends (e.g., data protection, SAST).	Limited discussion of trends with minor relevance to the PWA.	Clear explanation of trends with connections to the project (e.g., penetration testing, API security).	Thorough explanation of key trends, highlighting their impact on the PWA and relevance to client objectives.	/6
Lessons from Case Studies	Minimal mention of case studies, with vague or no explanation of impacts, lessons, or issues.	Limited discussion of some case studies (e.g., Equifax, Veeam) with basic lessons or evaluations.	Clear discussion of multiple case studies, detailing their impacts, lessons learned, and related issues.	Comprehensive analysis of all specified case studies (Equifax, Veeam, Facebook, Log4j), with detailed impacts, lessons, and evaluation of social, ethical, and legal ramifications.	/5

Benefits of Developing Secure Software (10 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Data Protection	Minimal discussion of the benefits of protecting sensitive data (e.g., avoiding breaches).	Limited explanation of benefits with some examples (e.g., compliance, financial security).	Clear explanation of data protection benefits, including examples like user credentials and business continuity.	Comprehensive explanation of data protection benefits with strong, specific examples and their relevance.	/5
Minimising Cyber Threats	Minimal mention of benefits of reducing threats (e.g., protecting user trust).	Limited discussion with basic examples (e.g., cost savings).	Clear explanation of benefits, including compliance, trust, and business continuity.	Comprehensive explanation of benefits with detailed discussion of competitive advantages and specific examples.	/5

Software Development Steps to Develop Secure Code (5 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Secure SDLC Explanation	Minimal or unclear discussion of SDLC phases.	Limited explanation of benefits with some examples (e.g., compliance, financial security).	Clear explanation of secure SDLC phases with relevant examples.	Comprehensive explanation of secure SDLC phases, privacy-by-design, and security integration.	/5

Influence of End Users on Secure Design Features (5 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Challenges and Recommendations	Minimal identification of user-related challenges or vague recommendations for secure design.	Limited identification of challenges with some basic recommendations (e.g., stronger passwords).	Clear identification of challenges (e.g., usability, accessibility) with practical recommendations like MFA.	Comprehensive identification of challenges, with tailored and well-justified recommendations (e.g., inclusive features, secure defaults).	/3
Balancing Security and Usability	Minimal or unclear explanation of how recommendations balance security and usability.	Limited explanation of balancing security with usability, with minor practical examples.	Clear explanation of how recommendations (e.g., RBAC, password strength indicators) balance both priorities.	Detailed and thoughtful explanation of balancing security and usability with strong, real-world examples (e.g., localisation, training).	/2

Security Features Incorporated Into Software (5 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Evaluation of Security Features	Minimal identification of flaws in security features (e.g., data protection, authentication).	Limited identification of flaws, with basic or vague solutions for some security features.	Clear evaluation of flaws across most areas (e.g., data protection, privacy) with actionable solutions.	Comprehensive evaluation of flaws across all specified areas, with well-justified, practical, and effective solutions.	/5

Cryptography and Sandboxing (5 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Explanation of Concepts	Minimal explanation of cryptography or sandboxing.	Basic understanding of one or both concepts.	Clear explanation of both concepts with relevance to the PWA.	Comprehensive explanation of cryptography and sandboxing with practical examples.	/5

Testing and Evaluating Security and Resilience (20 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Conducting Security Testing	Minimal testing conducted, with few or no vulnerabilities identified.	Limited testing conducted with partial identification of vulnerabilities.	Comprehensive testing conducted, identifying a wide range of vulnerabilities (e.g., SQLi, XSS, API security).	Advanced and systematic testing conducted, identifying all major vulnerabilities with strong evidence (e.g., tools, manual testing).	/10
Documenting Vulnerabilities	Minimal documentation with vague descriptions of vulnerabilities.	Basic documentation of vulnerabilities, with limited severity or impact analysis.	Clear documentation of vulnerabilities, including descriptions, severity ratings, and potential impacts.	Detailed documentation of vulnerabilities with severity, impact, and detailed recommendations for mitigation.	/10

Design, Develop and Implement Secure Code (20 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Design and Explanation of Solutions	Minimal or unclear explanations for addressing vulnerabilities.	Basic explanations for solutions to some vulnerabilities, with limited justification.	Clear explanation of solutions for at least three vulnerabilities, demonstrating secure coding practices.	Comprehensive explanation of solutions for multiple vulnerabilities, showing a deep understanding of secure design principles.	/10
Code Implementation and Fixes	Minimal or no implementation of secure code.	Basic or incomplete implementation of fixes for vulnerabilities.	Clear and practical implementation of secure code fixes for vulnerabilities (e.g., input validation, session management).	Exceptional implementation of secure code, addressing vulnerabilities thoroughly with detailed examples and well-commented code.	/10

Conclusion (5 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Summary of Findings and Recommendations	Minimal or vague summary of findings and recommendations.	Limited summary with basic recommendations for enhancing security.	Clear and concise summary of findings, with actionable and relevant recommendations.	Comprehensive summary of key findings, with strategic recommendations and emphasis on benefits and future improvements.	/5

References (5 marks)					
Criteria	**Basic**	**Limited**	**Effective**	**Highly Effective**	**Mark**
Quality and Consistency of References	Minimal or inconsistent referencing, with few or no credible sources.	Limited references, with minor formatting issues or unclear citations.	Clear and consistent referencing, with a variety of credible sources following appropriate standards.	Extensive, professional, and properly formatted referencing, showcasing a wide range of high-quality and relevant sources.	/5

TOTAL	/100

Feedback: