Introduction

Software vulnerabilities have widely been recognized as a serious threat to public and private organizations (Kaur & Singh, 2021; Khan et al., 2022). To mitigate this risk, governments have recognized the need to engage with the public to collect information about potential vulnerabilities (Faily & Flechais, 2020).

The National Cyber Security Centre (NCSC) website currently provides an online form (NCSC, n.d.) that members of the public can use to make Coordinated Vulnerability Disclosures (CVDs) which are captured, assessed, and actioned. The current process for reporting vulnerabilities requires for users to manually encrypt information, and communication with reporting users is not streamlined. The proposed system will address these flaws and make the process efficient and streamlined.

Apart from information about the vulnerability, the form only requires a valid email address, but the user is free to provide additional personal information such as name, surname and contact details.

System Proposal.

The proposed system will allow users to securely submit vulnerability reports. A user is required to create an account and log in before submitting a report but only an email is required in line with the NCSC's current workflow. Once logged in, users will also be able to view their previous reports or delete their personal details. A detailed list of CRUD capabilities can be found in the Appendix 1.

Commented [SF1]: This sentence doesn't make sense - is it me??

Commented [MX2R1]: Request.....which is consistent

Commented [SF3]: Appendix 1?

Secure Software Development Models.

Using an agile method, we will follow selected stages of the Microsoft secure SDLC (Lipner, 2004) involving sprints to code parts of the solution. Every sprint's objective will allow us to mitigate against the most serious threats as identified by the Open Web Application Security Project (OWASP) (OWASP, 2021) and the General Data Protection Regulation (2016).

A data flow diagram of the proposed system is shown in Figure 1.

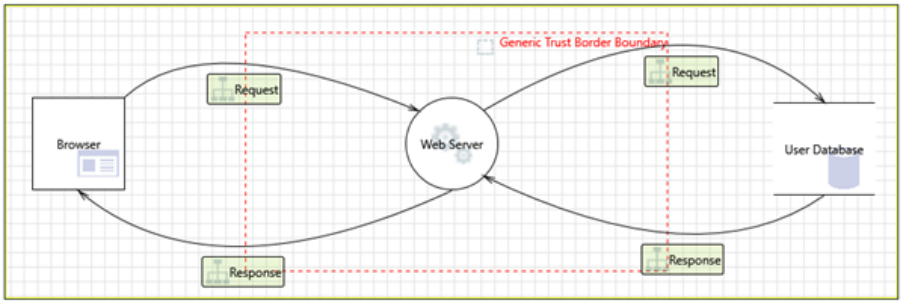

Figure 1: Data Flow Diagram

Libraries

1. Encryption: All data will be encrypted for confidentiality and data integrity which is crucial to keeping information about vulnerabilities secure until they are addressed.

AES is a widely-used symmetric encryption algorithm that provides strong encryption (Papadimitriou & Giannetsos, 2016) and is supported in Python (Liu, 2020).

2. Data Storage: The data captured by the system should be stored securely and in compliance with the GDPR. An SQLite database will be used as a proof-of-concept, as it is lightweight and easy to deploy (Arockiam & Chandrasekaran, 2017) but should be replaced by a secure RDBMS such as Oracle or MySQL in the final production.
3. Frameworks: Flask will be used which is a lightweight and flexible framework suitable for small to medium-sized applications (Grinberg, 2018). Flask 2.2.3 (with included libraries), Apache (web-server) (W3Techs, 2023), HTML5 and CSS3 (front-end).
4. Libraries: The Fernet library will be used to allow for the encryption and decryption of data. Also, the bcrypt library will be used to securely generate password hashes.
5. Backend: Python 3.11.

Assumptions and Limitations

1. Access: The system should be accessible both locally and remotely, and therefore it is necessary to deploy it on a cloud server.
2. Storage and CPU capacity: Given that vulnerability reports are not frequently received, and the data is in any case text-based, a medium-sized storage capacity along with a typically powerful CPU with sufficient RAM will suffice for this application.
3. The system will have two types of actors: Users can submit vulnerability reports and view/send secure messages from/to an admin user; Admin users can do all of the above as well as manage users and manage vulnerability reports.

Security challenges and mitigations

Table 2 shows a summary of the threats identified by STRIDE and mapped against OWASP Top10 2021 (OWASP, 2021) is shown below.

Threat	STRIDE Category	Priority	OWASP Top 10 2021
An adversary can gain unauthorized access to the database due to a lack of network access protection.	Elevation of Privileges	High	A01 A05
An adversary can gain unauthorized access to the database due to loose authorization rules.	Elevation of Privileges	High	A06
An adversary can gain access to sensitive PII or HBI data in the database.	Information Disclosure	High	A03 A05
An adversary can gain access to sensitive data by performing SQL injection.	Information Disclosure	High	A02
An adversary can deny actions on the database due to a lack of auditing.	Repudiation	Medium	A09
An adversary can tamper with critical database securable and deny the action.	Tampering	High	A03 A05
An adversary may leverage the lack of monitoring systems and trigger anomalous traffic to database.	Tampering	High	A08

Table 21: OWASP/STRIDE Threats

The team intends to mitigate against as many of the OWASP Top 10 as practicable; these can be seen in Table 3 with their respective mitigation methods.

Commented [SF4]: Rename to Table 1??

Commented [SF5]: Think this should be table 2??

Commented [MX6R5]: Yes

Threat	Description	Mitigation	STRIDE Category	OWASP Top 10 2021	Priority
Cross-Site Scripting	Attacker injects client-side scripts into web pages viewed by other users.	Effective input validation and sanitization, output encoding, and the use of Content Security Policy (CSP) can mitigate this (Kirda et al., 2006)	Tampering	A7	High
SQL Injection	Attacker injects malicious SQL statements into input fields, potentially giving them access to sensitive data.	Use of parameterized queries, input validation and sanitization (Halfond et al., 2006)	Tampering	A1	Critical
Cross-Site Request Forgery	An attacker executes actions on behalf of a victim who is logged in.	The use of CSRF tokens, input validation, and limiting the number of requests from a single IP address (Barth et al., 2008)	Tampering	A8	High
Session hijacking	Attackers steal a user's session ID, potentially allowing them to take control of the user's account	Secure session management techniques, such as expiring sessions after a certain amount of time or after a user logs out (Sekar et al., 2001)	Elevation of privilege	A2	High
Insufficient authentication and authorization	The attacker is given unauthorized access to sensitive information or functionality	Strong password policies, two-factor authentication, and role-based access control (Juels et al., 2006)	Elevation of privilege	A2	High
Denial-of-service	Attackers may use (D)DOS attacks to take down the service.	Make use of an Intrusion Detection and Prevention System	Denial of service	A10	Medium
Third-party software vulnerabilities	All third-party libraries, frameworks, APIs and software, if not updated regularly, can introduce vulnerabilities into the system	Regularly update all such components of the system (Frei, Christey, & Mohan, 2002)	Information disclosure	A9	Medium
Phishing and pretexting attacks on admin users	The attacker may use various social engineering techniques to obtain elevated (admin) access to the system	Conduct regular cybersecurity awareness training (Levy et al., 2016) and instil a security-aware culture (Lai et al., 2019)	Spoofing	A3	High

Table 3: Security Challenges and Mitigations

Commented [SF7]: I think you have removed the references to the citations used in this table; they should be reinstated into the reference list.

Commented [SF8]: This is the 2nd table, not the 3rd and should be renamed and referred to correctly, as suggested above.

Diagrams

The actions available to admin and normal users are summarized here:

- Figures 2 and 3: Use Case Diagrams (Admin/User)
- Figure 4: Class Diagram
- Figure 5: Sequence Diagram

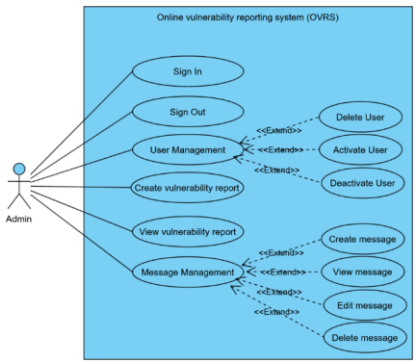

Figure 2: Admin Use Case Diagram

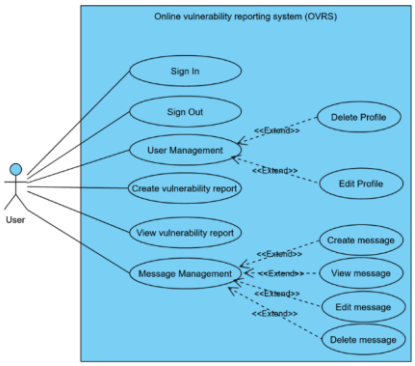

Figure 3: User Use Case Diagram

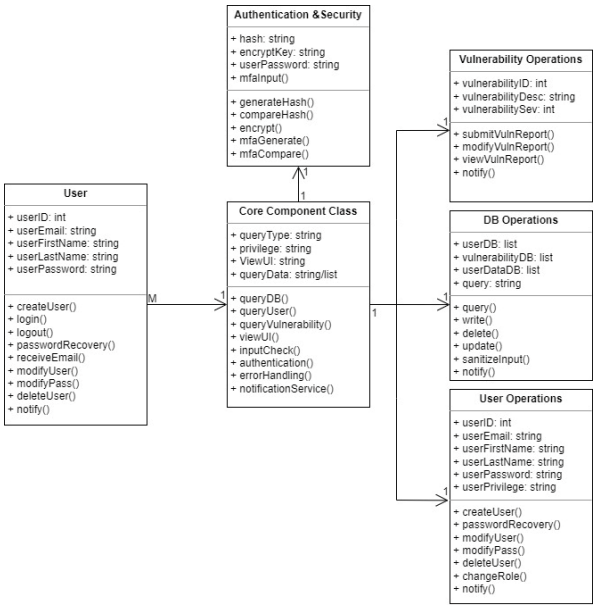

Figure 4: Class Diagram

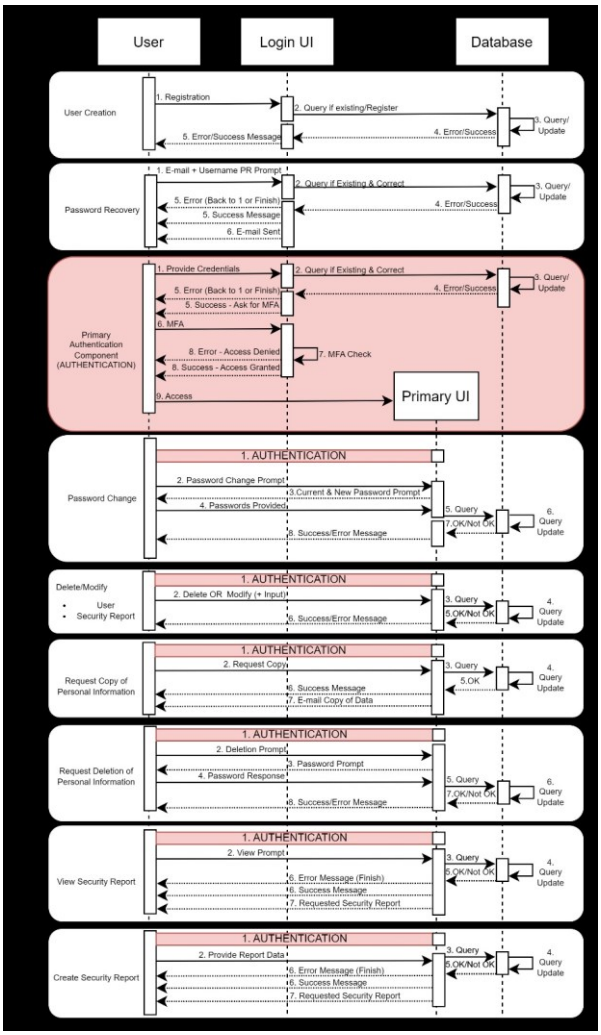

Figure 5: Sequence Diagram

References

Alqahtani, F., Alqahtani, M., & Alabdulatif, A. (2021). A review of software vulnerability reporting and disclosure processes. Journal of Information Security and Applications, 63, 102848. https://doi.org/10.1016/j.jisa.2021.102848

Arockiam, L., & Chandrasekaran, R. M. (2017). Comparative study of SQLite, MySQL, and Oracle. International Journal of Applied Engineering Research, 12(9), 2297-2301.

Barth, A., Jackson, C., & Mitchell, J. C. (2008). Robust defenses for cross-site request forgery. In Proceedings of the 2008 IEEE Symposium on Security and Privacy (pp. 75-89).

European Parliament and Council. (2016). Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). Official Journal of the European Union, L119, 1-88. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

Faily, S., & Flechais, I. (2020). Hacktivism in the United Kingdom: A study of prosecution files. Computers & Security, 96, 101933. https://doi.org/10.1016/j.cose.2020.101933

Frei, R., Christey, S., & Mohan, N. (2002). Analysis of third-party component vulnerabilities in commercial software. Proceedings of the 2002 ACM workshop on Computer security. New York, NY, USA: ACM.

Grinberg, M. (2018). Flask web development: Developing web applications with Python. O'Reilly Media, Inc.

Halfond, W. G., Orso, A., & Manolios, P. (2006). AMNESIA: analysis and monitoring for NEutralizing SQL-injection attacks. In Proceedings of the 2006 ACM SIGPLAN conference on Programming language design and implementation (pp. 120-129).

Juels, A., Rivest, R. L., & Szydlo, M. (2006). The blocker tag: Selective blocking of RFID tags for consumer privacy. In Proceedings of the 8th international conference on Ubiquitous computing (pp. 103-116).

Kaur, A., & Singh, R. (2021). Security vulnerabilities in software systems: A review. Journal of Network and Computer Applications, 175, 102886. https://doi.org/10.1016/j.jnca.2020.102886

Kirda, E., Kruegel, C., Vigna, G., & Jovanovic, N. (2006). Noxes: A client-side solution for mitigating cross-site scripting attacks. In Proceedings of the 15th international conference on World Wide Web (pp. 931-940).

Lai, C. H., Chen, C. W., & Chang, C. L. (2019). The impact of information security culture on information security awareness and behavior. Journal of Information Privacy and Security, 15(1), 1-25.

Levy, N., Ramim, M. M., & Schurr, A. (2016). The effect of awareness and training programs on employees' knowledge of information security policy and compliance intentions. Journal of Information Privacy and Security, 12(3-4), 161-176.

Li, W., & Liu, L. (2021). A Web-Based Vulnerability Management System for Small and Medium-Sized Enterprises. Journal of Information Processing Systems, 17(3), 542-554. https://doi.org/10.3745/JIPS.03.0144

Lipner, S., (2004). The trustworthy computing security development lifecycle. In 20th Annual Computer Security Applications Conference (pp. 2-13). IEEE.

Liu, W., Python Cryptography: A Review (2020). In 2020 IEEE International Conference on Computational Science and Engineering (CSE), Delft, Netherlands, 2020, pp. 322-327, doi: 10.1109/CSE49834.2020.00057.

National Cyber Security Centre (NCSC) (n.d.). About NCSC. National Cyber Security Centre (NCSC). Government of the Netherlands. Retrieved March 17, 2023, from https://www.ncsc.nl/english/about-ncsc

OWASP (2021). OWASP Top Ten. [online] Owasp.org. Available at: https://owasp.org/www-project-top-ten/.

Papadimitriou, P., & Giannetsos, T. (2016). Cryptography techniques and security assessment mechanisms for cloud computing environments. Journal of Cloud Computing, 5(1), 1-19.

Sekar, R., Bendre, M., Dhurjati, D., Bollineni, P., & Shi, J. (2001). A fast automaton-based method for detecting anomalous program behaviors. In Proceedings of the 2001 IEEE Symposium on Security and Privacy (pp. 144-155).

W3Techs. (2023). Usage statistics and market share of Apache. https://w3techs.com/technologies/details/ws-apache . Accessed: 20th March 2023.

Appendix A

Functional Requirement*	External User	Internal User (employee)	
		Office Admin	System Admin
Create internal User accounts	×	×	✓
Register external user	✓	×	×
Register internal user	×	×	✓
Confirm registration via email	✓	N/A	N/A
Login	✓	✓	✓
Password recovery	✓	✓	×
Change password	✓	✓	✓
Delete external user accounts	✓	×	✓
Amend external user profile	✓	×	×
Amend internal user profile	×	✓	✓
Delete external user account	×	×	✓
Delete internal user account	✓	×	✓
Request copy of personal data**	✓	✓	✓
Request deletion of personal data	✓	✓	✓
View submitted security reports	✓	✓	✓
Report security weakness	✓	×	×
Confirm submission of report by email	✓	N/A	N/A
Amend submitted security reports	✓	✓	✓
Delete a submitted security report	✓	×	✓

Table A2: Functional Requirements by User Type.

*It may be necessary to break down some functional requirements further to enable more fine-tuning of access rights.
**The rights of external users and employees are different under GDPR.

Commented [SF9]: Should say Appendix 1

Commented [SF10]: This table should have a title.