Development Team Project: Design Document

Introduction

The NCSC of the Netherlands aims to improve cyber security (Government of The Netherlands, n.d.). Dutch Internet Forensics proposes a secure repository application. The application will accept cyber-crime complaints from users, inform authorities, and store data securely. Police will investigate complaints from a database.

System Requirement:

Appendix AP01 lists Web server requirements. Dual systems will be installed behind load balancer in auto scaling configuration.

Local users will access the website within the network, whereas agents will connect over a VPN or local network to access the webpage.

Application Requirements

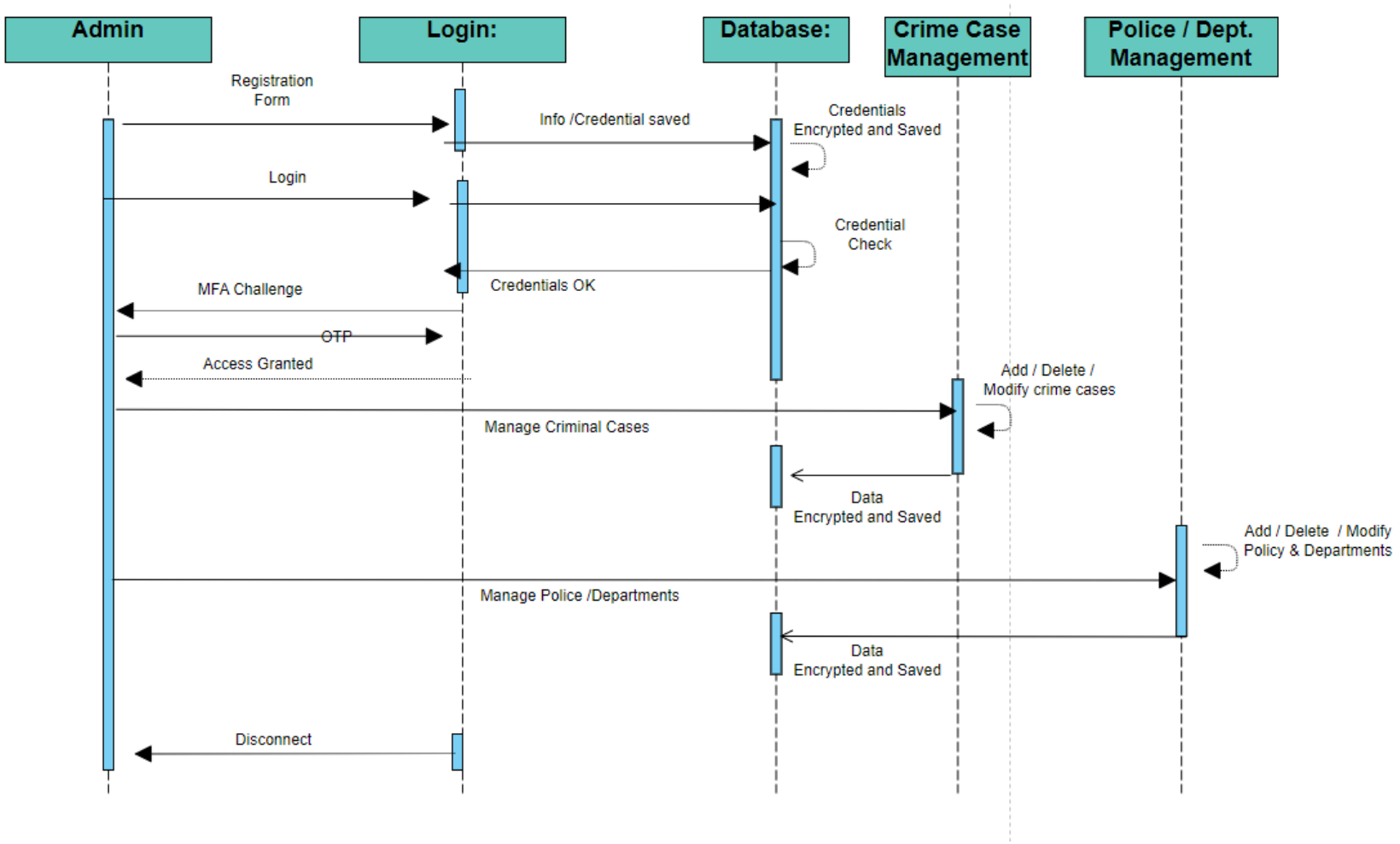
In the application,

- Cyber Crime complaint form will be available in a ticketing system for users to submit complaints.
- Existing complainants can come back to the ticket system, authenticate through MFA and check the status.
- Databases containing complaints have information that police can use to access the data.
- Data transmitted and saved in the database will be encrypted at all times and decrypted for police agency reading
- Different Python libraries will be used for encryption/decryption, MFA and database

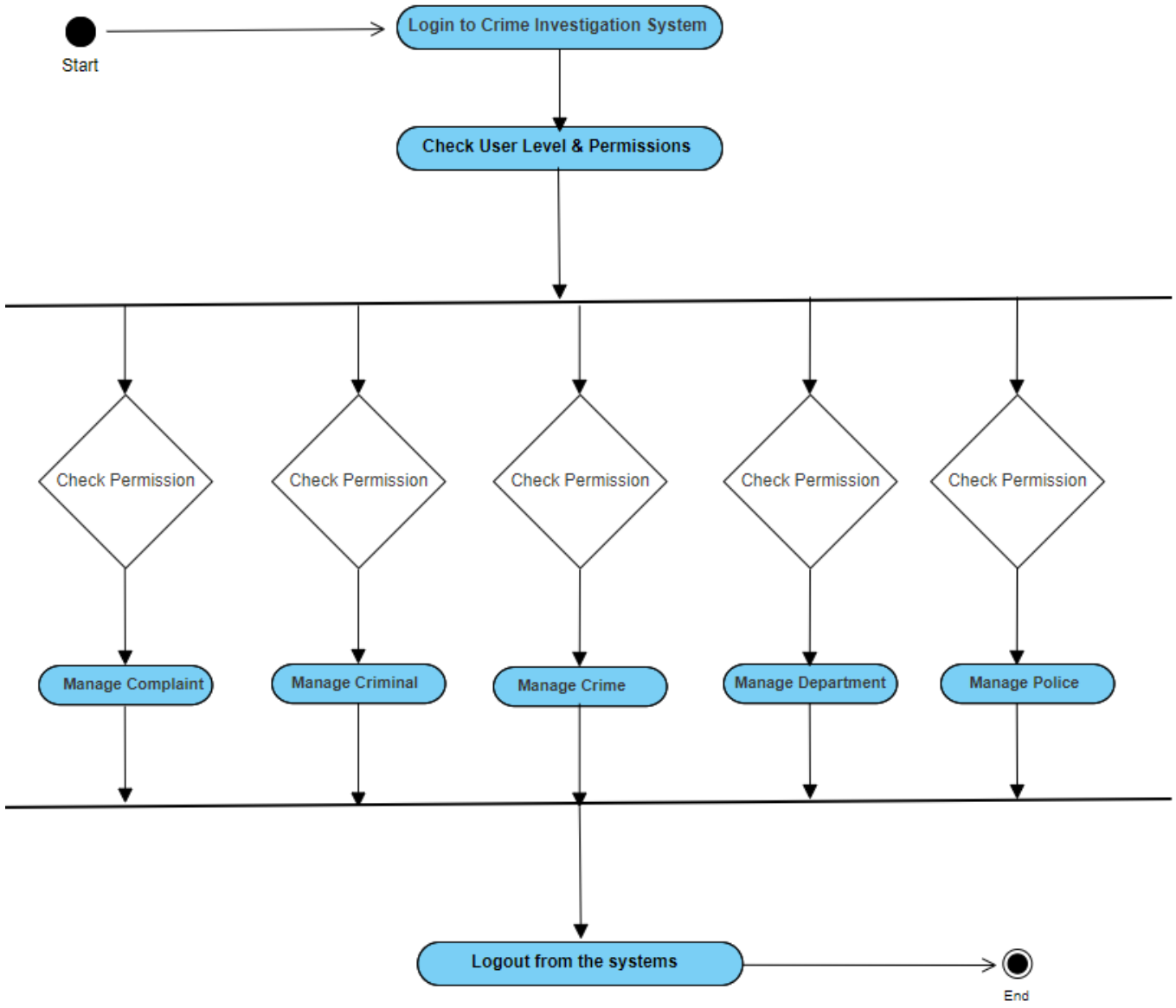
As Pohl & Hof (2015) stated secure scrum enhances application security in an appropriate manner. Thus, secure scrum focuses on the security while resources available are being used efficiently.

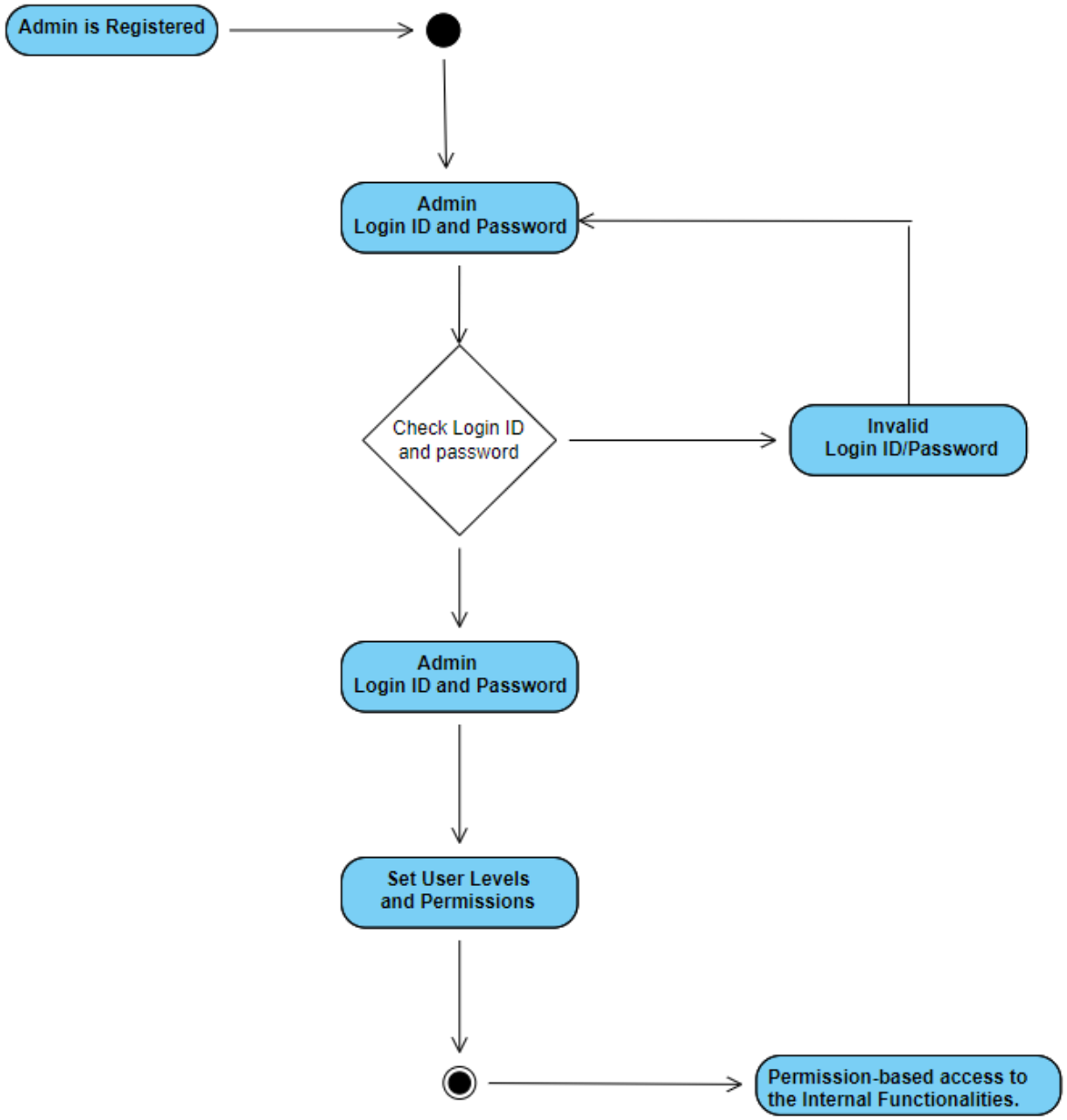
High level functionality of the application is presented in the diagrams below using **UML Diagrams** (Sequence, Activity and Class):

Sequence Diagram:

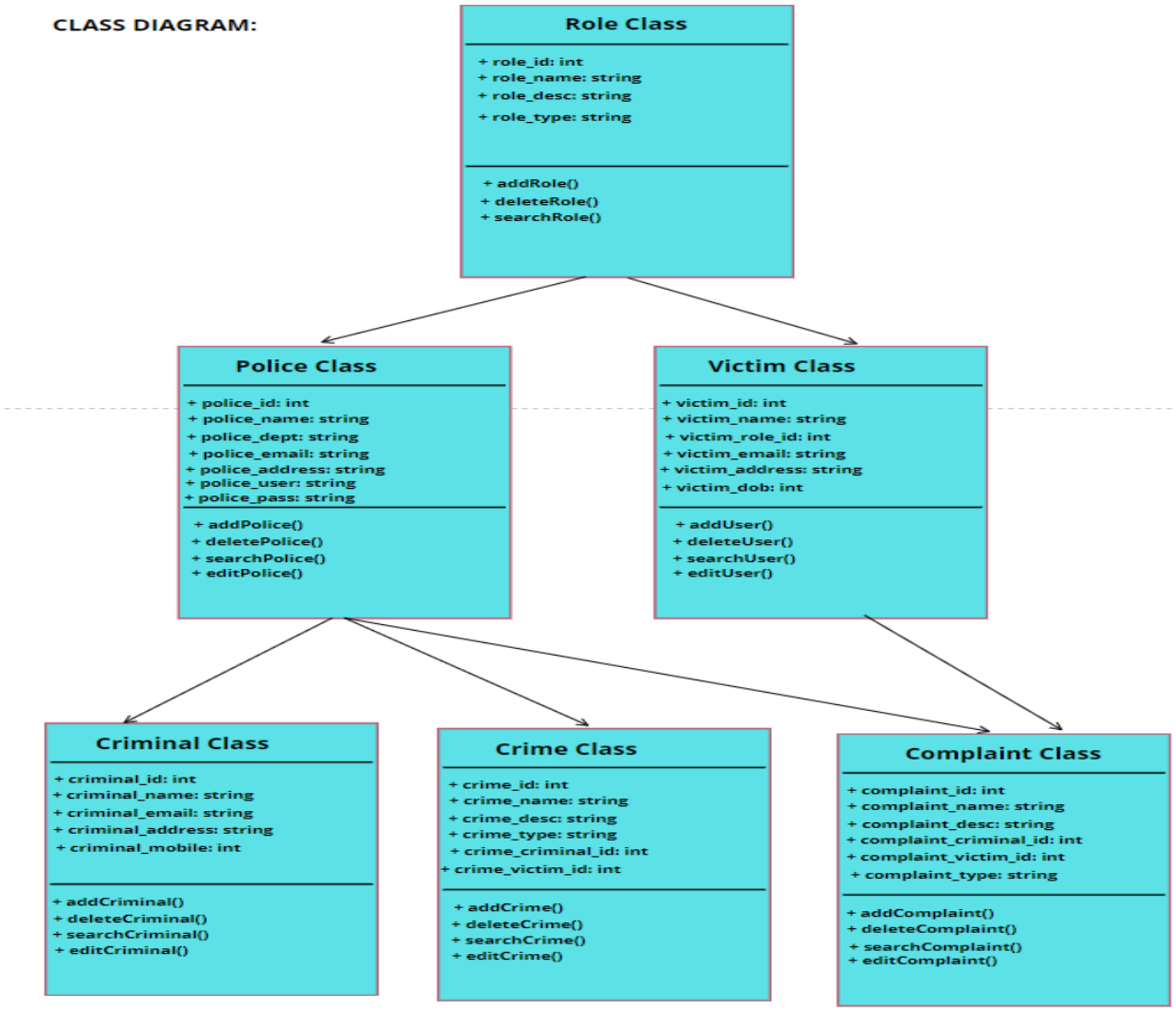


Activity Diagram


Start
Login to Crime Investigation System
Check User Level & Permissions
Check Permission
Check Permission
Check Permission
Check Permission
Check Permission
Manage Complaint
Manage Criminal
Manage Crime
Manage Department
Manage Police
Logout from the systems
End


Admin is Registered
Admin
Login ID and Password
Check Login ID
and password
Invalid
Login ID/Password
Admin
Login ID and Password
Set User Levels
and Permissions
Permission-based access to
the Internal Functionalities.

CLASS DIAGRAM:



Development Framework:

Here is the comprehensive list of requirements and the corresponding design decisions. According to Sane (2020), although the ranking of weaknesses mentioned by OWASP does not match the National Vulnerability Database; OWASP still includes all the vulnerabilities found in the NVD. Consequently, OWASP Top Ten was used to identify the security risks associated with the application and to propose countermeasures to mitigate those risks. The below table shows the risk and the mitigation to those risks (OWASP, n.d.)(Atiewi et. al, 2020)(Ciriani et. al, 2010)(Long, 2019).

REQ. ID	**RISK / REQUIREMENT**	**Design Decision ID**	**Design Decision**
RQ-01	Attackers may attempt to spoof real users/ complainants, resulting in data being sent to the attacker's devices	DD-01	Multi Factor Authentication (MFA) will be configured. All passwords / credentials must be encrypted before being transferred, and they should never be transmitted in clear text Traffic encryption must be implemented with the help of secure protocols like SSH / HTTPS / VPN etc.
RQ-02	User providing the Personal	DD-02	The system must provide an encrypted web application

	Identifiable Information (PII) needs to be secured		for the protection of sensitive information at rest and in transit.
RQ-03	Attackers have the ability to manipulate the information (for the data in transit)	DD-03	For complete authenticity and integrity, data must be hashed before transmission using the algorithms like SHA/MD5 etc.
RQ-04	Users risk having their passwords and other sensitive information stolen or compromised if their communication device is lost or stolen. All data and traffic should be encrypted before transmission to prevent unauthorized access or disclosure.	DD-04	Encryption will be required for all sensitive data, both during transmission or at rest. To restrict access to private information to just those who should see it, strong authorization measures are required.
RQ-05	Information exposure to individuals who should not have the access.	DD-05	Security principle of least privilege must be followed and constantly evaluated and strong authorization must be implemented.

We propose to use different open-source Python libraries for encryption, MFA, User Input validation etc. to achieve the desired security. Secure coding practices will be followed. To achieve Scalability, applications can be put in the public cloud behind the Application Load Balancer with auto-scaling.

Compliance & regulation requirements:

Items	Secure	GDPR
Personal Identification	All PII details should be secure	Yes
Location of Data	Within Europe	Yes
Data encryption	Encrypt data in transit and data at rest	Will be updated during implementation/testing

Python Libraries (Planned to be used):

PYTHON LIBRARY	PURPOSE
pyotp	Multi-Factor Authentication
base64	Data Encryption
base64	Data Decryption
sqlite	Database
re	Regular Expressions

CONCLUSION:

The application's sensitive data requires security components under GDPR. The team has identified the major security needs, challenges, and solutions. The development phase should focus on strong authentication, authorisation, and encryption.

References:

Atiewi, S., Al-Rahayfeh, A., Almiani, M, Yussof, S, AlFandi, O, Abugabah, A & Jararweh, Y. (2020) Scalable and Secure Big Data IoT System Based on Multifactor Authentication and Lightweight Cryptography. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9118946 [Accessed 18 Nov 2022].

Ciriani, V., Vimercati, S.., Foresti, S., Jajodia, S., Paraboschi, S. & Samarati, P. (2010) *Combining Fragmentation and Encryption to Protect Privacy in Data Storage.* ACM Transactions on Information and System Security. 13(3). Available from: https://dl.acm.org/doi/pdf/10.1145/1805974.1805978 [Accessed 18 Nov 2022].

Government of The Netherlands (n.d.) Fighting Cybercrim in the Netherlands. Available from: https://www.government.nl/topics/cybercrime/fighting-cybercrime-in-the-netherlands [Accessed 14 Nov 2022].

Johnstone, M. (2010) Threat Modeling with STRIDE and UML. Available from: https://ro.ecu.edu.au/cgi/viewcontent.cgi?article=1087&context=ism [Accessed 20 Nov 2022].

Long, S. (2019) A Comparative Analysis of the Application of Hashing Encryption Algorithms for MD5, SHA-1, and SHA-512. Available from: https://iopscience.iop.org/article/10.1088/1742-6596/1314/1/012210/pdf [Accessed 18 Nov 2022].

OWASP (n.d.) OWASP Top Ten. Available from: https://owasp.org/www-project-top-ten/ [Accessed 18 Nov 2022].

Pol, C. & Hof, H. (2015) Secure Scrum: Development of Secure Software with Scrum. Available from: https://arxiv.org/pdf/1507.02992.pdf [Accessed 18 Nov 2022].

Sane, P. (2020) Is The OWASP Top 10 List Comprehensive Enough for Writing Secure Code. Available from: https://arxiv.org/pdf/2002.11269.pdf [Accessed 15 Nov 2022].

Appendix:

AP01: System requirement:

Item	Specification
Processor	Intel Xeon or comparable multi core CPU
Memory	16GB RAM (recommended 4 GB free
Storage	10GB disk space for Web server 30 GB disk space for Database
Operating system	Linux/Ubuntu
Web Server	Apache/Tomcat

AP02: Project Timeline

Project Timeline						
Activities	Week1	Week 2	Week 3	Week4	Week5	Week6
Initiate						
Plan and estimate						
Implement						
Testing						
Review and retrospect						
Release						