Secure Development Lifecycle Policy

1. Purpose

1.1. Queen Mary University of London (Queen Mary, The University) is an institution of research and learning and is heavily dependent on using information in all forms. Protecting the institution's information assets and the IT systems on which these are processed, stored, or transmitted will help enable the University to fulfil its mission, comply with legislation, protect its reputation, and ensure that a high-quality service can continue to be offered to our students, employees, and alumni.

1.2. The over-arching Information Security and Acceptable Use Policy (IS01) describes The University's overall security objectives and associated policy objectives.

1.3. This policy (Secure Development Lifecycle Policy) outlines the minimum requirements of good practices to be adopted by analysts and software developers, making the process of designing systems more reliable, auditable, stable and protected against threats.

2. Legislative and regulatory context

2.1. Queen Mary's Information Security Management System (ISMS) is aligned to the requirements of the International Standard ISO 27001: 2022. This is recognised as best practice in information security management. This policy is aligned with the ISMS Legal Register.

3. Terminology

3.1. The following terms will be used throughout the policy:

- Must is used to state a **mandatory** requirement of this policy
- Must not is used to state a **prohibition** of this policy
- Should is used to state a **recommendation** of this policy

4. Scope

4.1. This policy applies to all employees, University affiliates, agency employees, and contractors with responsibility for software development and configuration of IT services.

4.2. This policy covers all locations including remote workers and overseas campuses.

4.3. This policy does not apply to Undergraduate and Post Graduate Taught students.

5. Principles

5.1. The overriding principle of "Secure by Design[1]" must be applied to all software development whether developed in-house, or whether outsourced to a third party.

5.2. Where stages of software development are the responsibility of a third-party, the standards outlined within this policy must be ensured via contractual or other appropriate methods for the appropriate stages of the development lifecycle.

5.3. Queen Mary utilises a development lifecycle "V-Model". Each section of this document includes a mapping to the relevant phases of the v-model.[2]

6. Roles and responsibilities

6.1. It is the responsibility of ITS staff implementing new services or modifying developed applications (whether directly, or via a third-party contractual arrangement) to ensure that the requirements set out in this policy are compiled-with. These staff must review this policy annually.

7. Separation of Environments

7.1. Development, testing, and production environments must be labelled and separated in line with IT Services environment strategy.

7.2. Identification labels must be visible in menus displaying the appropriate environment for the code.

7.3. All environments must be protected through regular patching and updates in line with the requirements for the production environment outlined in the IS19 Vulnerability and Malware Policy.

7.4. All environments must be securely configured in line with the requirements of the IS06 Configuration & Change Management Policy.

[1] For a definition of Secure by Design, refer to https://www.ncsc.gov.uk/collection/cyber-security-design-principles

[2] For a definition of V-Model, see https://en.wikipedia.org/wiki/V-model

7.5. Access controls must be in place for all environments in line with the requirements of the IS15 Access Control Policy.

7.6. All environments must be monitored for change in line with the IS20 Logging and Monitoring Policy, as well as IS06 Configuration & Change Management Policy.

7.7. A single person must not have the ability to make changes to both development and production environments without prior approval; this must be enshrined in formal software development procedure.

7.8. Independent periodic tests must be carried out to ensure the security of the different environments (in line with defined security standards).

Development V-Model phases relevant to this section	
V-Model Phase	**Description**
Architecture design	Exchange of data and communication between the internal modules and external systems are well-understood and defined

8. Secure development coding requirements

8.1. Code must be reviewed prior to release by a competent person, and someone other than the code author / developer.

8.2. The University's secure development lifecycle process must be followed during all development activities, including software components from third parties and open source.

8.3. Secure and structured programming techniques must be used.

8.4. Programming defects must be documented and removed.

8.5. Prior to new developed software being promoted to production, the following must be evaluated:

- The attack surface
- Application of the principle or least privilege
- Analysis of the most common programming errors and documenting that these have been checked/mitigated
- Updates to code must be securely packaged and deployed

8.6. Identified vulnerabilities must be handled within agreed timescales as per the Vulnerability & Malware policy.

8.7. Vulnerabilities or misconfigurations reported through responsible disclosure must be handled within agreed timescales as per the IS19 Vulnerability & Malware policy.

8.8. Errors and suspected incidents must be logged in order to enable review and for the necessary corrective changes to be made.

Development V-Model phases relevant to this section (see Policy Information and Document Control)
System Design, System Testing, Module Design, Unit Testing, Coding Phase

9. Access to source code

9.1. Development code must be stored in a secure code repository with access control and segregation of duty in place in line with the requirements of the IS15 Access Control Policy.

9.2. Read and write access to source code, development tools and software libraries must be appropriately managed based on employee's job roles and responsibilities in line with the IS15 Access Control Policy and the overriding principle of least privilege.

9.3. A source code management system must be used to centrally store and manage access to code, with version control capabilities.

9.4. An audit log must be maintained of all accesses and changes to source code.

Development V-Model phases relevant to this section (see Policy Information and Document Control)
Module Design, Coding Phase, Unit Testing, Integration Testing, System Testing

10. Testing

10.1. Testing must be performed in a test environment that matches the target production environment as closely as possible, following ITIL Release Management and Change Management processes.

10.2. Security testing must include the testing of:

a) Security functions e.g. user authentication, access restrictions and use of cryptography

b) Secure coding

c) Secure configurations, including operating systems, firewalls and other security components

10.3. Testing plans must be determined using set criteria which must include:

a) Detailed schedule of activities and test

b) Inputs and expected outputs under a range of conditions

c) Criteria to evaluate the results

d) Decision for further actions as necessary

10.4. Automated tools such as code analysis tools and vulnerability scanners must be utilised where necessary.

10.5. Testing must be performed against the most recent OWASP top 10 at the time of testing (OWASP Top 10:2021).

10.6. All vulnerabilities identified during testing must be corrected prior to promotion to production or managed as part of the risk management process as per IS08 Risk Management Policy.

Development V-Model phases relevant to this section (see Policy Information and Document Control)
Coding Phase, Unit Testing, Integration Testing, System Testing

11. Test data

11.1. Access control procedures must be applied to test data systems in line with the IS15 Access Control Policy.

11.2. Where sensitive information is required at test this must first receive adequate review as per https://www.qmul.ac.uk/governance-and-legal-services/governance/information-governance/data-protection/data-protection-impact-assessments/ and suitable techniques employed such as data masking, pseudonymization or anonymization.

11.3. Test data must be properly deleted from test environments immediately after testing is complete.

11.4. Test data must be stored securely and undergo independent validation.

Development V-Model phases relevant to this section (see Policy Information and Document Control)
System Design, Architecture Design, Module Design, Coding Phase, Unit Testing, Integration Testing, System Testing

12. Code in production

12.1. Code must be promoted to production by approved personnel only.

12.2. Code promoted to production must be subject to the change control process, following ITIL Release Management and Change Management processes.

12.3. Production environments must be backed-up prior to the promotion of code to production to aid roll-back in the event of a failed change.

12.4. Test data must be removed before the system is promoted to production.

12.5. Development files and test data must not be stored in the production environment.

Development V-Model phases relevant to this section (see Policy Information and Document Control)
System Design, Architecture Design, Module Design, Coding Phase, Unit Testing, Integration Testing, System Testing

13. Application security requirements

13.1. Application security requirements must be identified and specified as part of the functional and non-functional requirements which must be determined through a risk assessment.

13.2. Information security requirements must be identified, specified, and approved when developing or acquiring applications.

13.3. Applicable local, state, national, and international laws and regulations must be considered for application development and system design.

13.4. Testing of applications must occur prior to production in a test or "sandbox" environment.

13.5. Applications must be tested against critical application security flaws.

Development V-Model phases relevant to this section (see Policy Information and Document Control)
Requirements Analysis, System Design, Architecture Design, Module Design, Coding Phase, Unit Testing, Integration Testing, System Testing, Acceptance Testing

14. Outsourced development

14.1. The University must direct, monitor and review activities related to outsourced system development, with the requirements and expectations of the supplier clearly communicated and agreed as part of the supplier approval and management process.

14.2. Contracts issued by the University must agree the right to audit suppliers carrying out development activities on its behalf.

14.3. The University must ensure suitable security controls are in place for suppliers carrying out development activities on their behalf, prior to engagement.

14.4. Following a risk-based approach, with reference to contract management process, escrow arrangements for software source code should be in place with suppliers carrying out development activities on behalf of the University.

Development V-Model phases relevant to this section (see Policy Information and Document Control)
Requirements Analysis, System Design, Architecture Design, Module Design, Coding Phase, Unit Testing, Integration Testing, System Testing, Acceptance Testing

15. Monitoring

15.1. Compliance with the policies and procedures laid out in this document must be monitored by the Information Security Team as directed by the Chief Information Security Officer, along with independent reviews by both internal and external auditors on a periodic basis.

15.2. The Governance, Risk and Compliance Manager, in conjunction with the Chief Information Security Officer, is responsible for the monitoring, revision and updating of this document.

15.3. The effectiveness of this document must be evaluated at Infosec Monthly Meetings as well as the annual management review with Senior Management.

16. Exceptions

16.1. Any exception to this policy must be documented with business reasoning and signed off by the University's Chief Information Officer and the Chief Information Security Officer before implementation.

17. Review

17.1. This policy must be reviewed by the policy owner at least every three years.

17.2. Minor updates to this policy that do not affect the rules, principles or intent of this policy may be approved by the Chief Information Security Officer on behalf of the Information Governance Group.

Policy Information and Document Control

Policy title	IS23 – Secure Development Lifecycle policy
Version number	V1.2
Related policies and procedures	ISMS Statement of Applicability ISMS Legal Register IS01 Information Security & Acceptable Use Policy IS04 Information Security Incident Reporting and Management Policy IS06 Configuration Management Policy IS07 Third Party Supplier Assessment and Management Policy IS08 Risk Management Policy IS13 Asset Management Policy IS15 Access Control Policy IS19 Vulnerability and Malware Policy IS22 Use of Cloud Services Policy DG12 Cryptographic Controls Policy DG09 Information Classification Policy
Superseded policies	n/a
Approval level	Information Governance Group
Approval date	16th July 2025
Last review date	21st May 2026
Next review due	May 2029
Policy owner	Rachel Bence - Chief Information Officer
Policy contact	Ambrose Neville - Chief Information Security Officer

Version Control

Version	Date	Reason for updates/Summary of key changes
v1.0	22/07/2025	Initial version
v.1.1	08/01/2026	Periodic review for the PSG 2026 review. Minor updates include the use of 'Must' instead of 'Shall' as per the University Policy Writing Guide.
V1.2	21/05/2026	Minor corrections.

Appendix

Development V-Model References

Stage	Purpose
Requirements Analysis	Defines what the system should do by gathering functional and non-functional requirements from stakeholders
System Design	Translates requirements into a high-level system architecture, specifying hardware, software, and data structures
Architecture Design	Breaks the system into modules/components and defines their interactions, focusing on interfaces and data flow
Module Design	Specifies the internal logic of each module, including algorithms, data structures, and control logic
Implementation	Converts detailed designs into executable source code using secure coding practices
Unit Testing	Tests individual modules for correctness based on module design; typically automated and performed by developers
Integration Testing	Verifies interaction between modules, ensuring data exchange and control flow function as intended per the architecture
System Testing	Validates the complete integrated system against system design specifications; includes performance, usability, and security tests
Acceptance Testing	Confirms that the system meets user needs and requirements as defined in the initial phase; conducted by end users or clients