Guidelines for Application Development - DRAFT

IIUM ICT GUIDELINES

PREPARED FOR:
International Islamic University Malaysia

PREPARED BY:
Information Technology Division

Document Change Log

Release Version	Date	Pages Affected	Remarks/Change Reference
Version 1.0	25/06/2025	-	*Endorsement from ITD Management*
Version 2.0	28/10/2025	4 & 5	Amend item 2.2

Responsibility and Activity Log

Requestor	Description	Submission Date	Approval Date
Siti Zarina binti Muhamat	Endorsement from ITD Management	23/06/2025	25/06/2025
Siti Zarina binti Muhamat	Submission to ITD Management	28/10/2025	-

1. OBJECTIVE

1.1 This guideline aims to provide a clear, practical, and standardized approach for the development and management of application systems within the university.

1.2 It serves as a reference for all individuals—both technical and non-technical—who are involved in planning, designing, developing, or managing application systems.

1.3 The objectives of this guideline are as follows:

1.3.1 To provide fundamental guidance to KCDIOM and IIUM vendors or individuals involved in application system development and application project management.

1.3.2 To outline the guideline to ensure uniformity, consistency, and interoperability between newly developed systems and the existing university environment.

1.3.3 To serve as the primary reference for the development of application systems at IIUM.

1.3.4 To introduce a practical, standardized methodology as the official framework for IIUM stakeholders in managing and developing application systems effectively.

1.3.5 To ensure that systems developed meet the actual needs of users and support the academic, administrative, and service functions of the university.

1.3.6 To comply with recognized security and quality standards that safeguard data and ensure trustworthiness of systems.

1.3.7 To promote the use of consistent tools, technologies, and platforms to simplify development, support, and maintenance efforts.

1.3.8 To ensure systems are reliable, user-friendly, and able to operate smoothly and continuously over time.

1.3.9 To contribute to a more efficient, transparent, and sustainable system development environment at IIUM.

2. SCOPE

2.1 The scope of this guideline covers the management and development of new application systems, as well as enhancements or improvements to existing systems based on current and evolving university needs.

2.2 This procedure shall be implemented in accordance with IIUM Information Security

Management System (IIUM ISMS) Procedure and in compliance with **ISO/IEC 27001:2022** requirements to ensure confidentiality, integrity, and availability of information. All processes, decisions, and activities under this policy/guideline must uphold these principles to protect the University's information assets and data

2.3 It applies to application systems that are:

2.3.1 Developed in-house by IIUM's internal teams,

2.3.2 Outsourced to external vendors or service providers,

2.3.3 Co-sourced, involving collaboration between internal teams and external parties.

2.4 This guideline is applicable to all KCDIOM within IIUM that are involved in any part of the system lifecycle, including:

2.4.1 **System Development Lifecycle (SDLC)**: From initial planning, requirement analysis, design, development, testing, deployment, to post-deployment support and maintenance.

2.4.2 **Roles and Responsibilities**: Involving requestors, system owners, developers, technical coordinators, and support personnel.

2.4.3 **Standard Practices**: Covering documentation, coding standards, testing protocols, deployment procedures, and version control.

2.4.4 **Technology Standards**: Use of approved tools, platforms, programming languages, and frameworks to ensure consistency across systems.

2.4.5 **Security and Compliance**: Adhering to IIUM's policies on data privacy, cybersecurity, and regulatory requirements.

2.4.6 **Integration and Interoperability**: Ensuring system compatibility with existing IIUM infrastructure and applications.

2.4.7 **System Sustainability**: Supporting long-term maintainability, scalability, and continuity of services.

3. TERMS AND DEFINITIONS

Term	Definition
IIUM	The International Islamic University Malaysia, otherwise known as the "University"
ICT	Information and Communication Technology
ITD	Information Technology Division
ITD Management	Chief Information Officer, Director, Deputy Information Technology Officer and Deputy Engineer of ITD
KCDIOM	Kulliyyah / Centre / Division / Institute / Office / Mahallah
SQL	Structured Query Language
XSS	Cross-site scripting

4. GUIDELINES

This section provides detailed guidance for managing the end-to-end lifecycle of application system development at IIUM. It supports all forms of development—in-house, outsourced, and co-sourced—and ensures that systems are developed in accordance with the university's strategic direction, technical standards, and governance requirements.

4.1 **System Development Lifecycle (SDLC) : All application development must follow IIUM IT Project Management Lifecycle (Refer to: https://division.iium.edu.my/itd/it-project-management/)**

4.1.1 **Initiation:** This phase begins with the recognition of a need for a new application system or enhancement of an existing one.

Step 1: Submit IT Request Form

- All new system requests must be submitted using the official IIUM IT Request Form by KCDIOM.
- The request should include a clear problem statement, background, and expected outcomes.

Step 2: Project Determination

- KCDIOM will be assessed whether the request qualifies as a project, based on factors such as (IT Classification Form):
  - Level of change in business process

- Multi-departmental impact
- Project team member size
- Timeline and resources

If declared as a project, the following steps must be completed:

- **Feasibility Study Report**: Analyze cost-benefit, technical feasibility, and resource requirements.
- **UTICTEC and JPICT Endorsement**: Present project justification and feasibility report to governance committees for formal approval.
- Issuance of Appointment Letter: Formally appoint a project manager and development team. Define project sponsor, system owner, and roles of key stakeholders.

Purpose of the Initiation Phase:

- Clearly define the business need.
- Determine whether full project governance is required.
- Obtain initial approvals for further planning.

4.1.2 **Planning:** The Planning phase defines the strategic, functional, and technical blueprint for the application system and ensures all necessary preparations are in place before development begins. It includes formalizing the project scope, timelines, responsibilities, and requirements.

4.1.3 **Execution:** The Execution phase involves the actual development of the application system, following the System Development Life Cycle (SDLC) using agile or hybrid methodologies. During this phase, the project team designs, builds, tests, and prepares the system for deployment based on the approved requirements. Activities include detailed system design, coding, regular updates to the Project Management Plan (PMP) and Change Register, and continuous progress monitoring. System Integration Testing (SIT) is conducted to ensure technical compatibility, followed by User Acceptance Testing (UAT) to validate the system against user expectations. User training is carried out to ensure smooth adoption, and essential documentation such as the User Manual

and Operations Manual is prepared. Agile practices are embedded through iterative cycles of planning, designing, building, testing, and deploying. The primary goal of this phase is to develop a fully functional, tested, and user-ready system that meets business objectives and is prepared for go-live.

4.1.4 **Closing:** The Closing phase marks the formal conclusion of the application development project. It involves finalizing all activities to ensure the system is fully transitioned to operational support and that all deliverables have been completed and accepted. Key tasks include preparing the Project Handover Report, compiling the Project Closing Report, and conducting a Project Closing Meeting with relevant stakeholders. A Post-Implementation Review is carried out to assess project outcomes, identify lessons learned, and document recommendations for future projects. The system owner assumes full responsibility for ongoing operations, and all project documentation—including source code, manuals, and training materials is archived in the designated repository. This phase ensures that the project is officially closed, all responsibilities are transferred, and the university retains institutional knowledge for continuous improvement.

4.2 **Documentation Requirements: All application system development initiatives at IIUM must be supported by comprehensive documentation throughout the project lifecycle. Proper documentation ensures clarity, accountability, effective communication, and alignment with institutional standards.**

The required documents are aligned with the respective phases of the IIUM IT Project Management Lifecycle and must follow the listings as outlined in the official Project Management Review Report. These documents serve as mandatory references and checkpoints to support project governance, system quality, and successful delivery.

4.3 **Technology Standard: To ensure quality, maintainability, security, and interoperability, all application systems developed at IIUM must adhere to the following technology standards. These standards apply to all in-house, outsourced, and co-sourced development efforts and must be enforced throughout the system development lifecycle. All deviations or exceptions from these standards require formal approval from ITD Management.**

4.3.1 **Approved Technology Stack:**

- All platforms, programming languages, databases, and frameworks must be approved by ITD Management based on recommendations from KCDIOM.
- The preferred stack includes **open-source technologies** (e.g., PHP, Python, JavaScript, PostgreSQL, MySQL) to ensure cost efficiency, transparency, and community support.
- Any proposal to use alternative or proprietary technologies must be justified and formally approved by **ITD Management**.

4.3.2 **Web-Based Application Requirement**

- All systems must be developed as web-based applications unless otherwise approved.
- Applications must be mobile-responsive and accessible via all modern web browsers.
- Non-web systems (e.g., native mobile apps or desktop applications) require **justification and written approval from ITD Management**.

4.3.3 **Coding Standards and Best Practices**

- Developers must use consistent naming conventions for files, functions, variables and classes.

- Follow structured code formatting and indentation to improve readability.
- Implement error handling and logging mechanisms to support system reliability.
- Provide inline comments and documentation to explain logic and improve collaboration.
- Apply modular programming principles, where reusable components and services are created to avoid code duplication.
- Hardcoding of credentials, configuration values, URLs, file paths, or environment-specific settings is strictly prohibited. All such data must be stored in secure and configurable files, such as environment variables or external configuration files.
- Use appropriate input validation and sanitation to prevent security vulnerabilities (e.g., SQL injection, XSS).
- Conduct peer code reviews for quality assurance and knowledge sharing.
- Ensure all code changes are tested before merging into the main branch.

4.3.4 **Version Control: All application development at IIUM must utilize GitLab (https://gitlab.iium.edu.my) as the official version control platform to ensure traceability, collaboration, and change management throughout the system development lifecycle.**

Developers must follow these practices:

- All source code must be committed and maintained in the central GitLab repository managed by Application Development Sections.
- Each developer must create and work on their own branch (e.g., feature/[feature-name], bugfix/[issue-name]) instead of committing directly to the main or production branch.
- Merging into the main branch must only be performed

through Merge Requests (MRs) in GitLab and must be reviewed and approved by the assigned reviewer or project lead before merging.

- Use clear and descriptive commit messages that summarize the purpose and impact of the changes.
- Apply a standardized branching strategy, such as:
  - main – stable production-ready code
  - develop – staging for upcoming releases
  - feature/* – for new features
  - bugfix/* – for fixing issues
  - hotfix/* – for urgent fixes in production
  - release/* – for pre-release preparation
- Use GitLab's tagging feature to mark release versions (e.g., v1.0.0, v2.1.3-rc).
- Regularly sync branches with the latest codebase to avoid major conflicts.
- Sensitive configuration files (e.g., .env) must never be committed to the repository.

All internal teams and external vendors are required to use GitLab in accordance with this policy for all IIUM application development initiatives.

4.3.5 **System Architecture and Design: All application systems developed at IIUM must be designed using robust, scalable, and modern architecture principles to ensure long-term sustainability, performance, and maintainability. Proper architectural planning is essential and must be documented, reviewed, and approved prior to development.**

Design Requirements and Principles: Systems must adopt the following architectural principles.

- Modular or layered design – to enforce separation of concerns and logical maintainability.
- Microservices architecture – if suitable, to allow independent deployment and scaling of components.
- API-first and service-oriented architecture (SOA) – to facilitate system integration, reuse, and interoperability with other IIUM systems.

Design Documentation Submission: The following documentation must be prepared and submitted during the planning phase for approval.

- System architecture diagram
- Data models and Entity Relationship Diagrams (ERDs)
- Business process workflows
- Integration design (APIs, authentication mechanisms, data formats)

Database Design Approval: To ensure data integrity, consistency, and efficient data modeling

- A complete initial database design must be submitted to the Database Administrator (DBA) for review and approval before development starts.
- If there are any changes during development, a final and updated design must be re-submitted to the DBA for approval before deployment to production.
- All designs must comply with IIUM's database naming conventions, normalization rules, and data governance policies.

Environment Separation and Hosting Requirements: All application systems must implement a three-phase environment to support proper development, testing, and release processes.

- Development Environment - Used for initial coding and internal testing by developers. May include sample/mock data for unit testing.
- Staging (UAT) Environment - Mirrors the production setup and is used for User Acceptance Testing (UAT) and validation by system owners. Final system verification is conducted here.
- Production Environment - The live environment used by end users. Must be isolated from development and staging to ensure security and stability.

Applications must be hosted on dedicated application servers or central servers assigned for reporting and file management, as designated and managed by ITD.

4.3.6 **Integration and Interoperability: All application systems developed or implemented at IIUM must be designed to work seamlessly with the university's existing digital ecosystem. This includes ensuring secure and efficient access to shared data, avoiding duplication, and supporting communication between systems.**

Single Sign-On (SSO) Integration:

- All systems must support Single Sign-On (SSO) through IIUM's Central Authentication Service (CAS) at https://cas.iium.edu.my/cas .
- This allows users to log in once using their IIUM credentials and gain access to multiple systems without needing to re-enter their login information.

- SSO improves user experience, reduces password fatigue, and enhances account security through centralized authentication management.
- Developers must ensure SSO is implemented properly and tested in collaboration with the ITD authentication team.

Use of Standardized APIs:

- All data exchange between systems must use standardized and secure APIs (Application Programming Interfaces) provided or approved by the data owner.
- APIs enable systems to communicate reliably with each other—for example, retrieving student profiles, staff records, or financial information from core databases.
- Developers must not create separate data copies (data silos) but instead call or integrate with existing official APIs or data services maintained by the university.
- New systems that need to expose data must design their own APIs following RESTful principles and security best practices, subject to review by ITD Management.

Avoiding Data Silos and Redundancy:

- Systems must not store or replicate data that already exists in central systems such as:
  - Student Information System (SIS)
  - Human Resources Information System (HURIS)
  - Integrated Financial Information System (IFIS)
- For example, if student or staff data is required, the system must retrieve it via integration, not store a separate, outdated copy.
- This ensures data accuracy, real-time access, and easier maintenance in the long term.

Integration Planning

- During the planning phase of any system development, the integration requirements must be clearly defined and documented.
- System owners and developers must:
  - Identify required data sources and destination systems
  - Specify how the data will be accessed (API, SFTP, etc.)
  - Work with ITD to ensure compatibility and security.
  - Obtain the appropriate data owners' approval.

- Systems without integration may be rejected if they are likely to cause data fragmentation or redundancy.

4.3.7 **Security Standards and Multi-Factor Authentication (MFA): To protect university data, ensure user privacy, and reduce the risk of cyber threats, all application systems at IIUM must follow strict security standards.**

General Security Requirements

All applications, whether developed in-house, outsourced, or co-sourced, must implement the following foundational security measures.

- Secure Authentication: All systems must authenticate users using IIUM's Single Sign-On (SSO) service where applicable.
- Application systems must implement secure session management practices to prevent unauthorized access or session hijacking. This includes:
  - Enforcing session timeouts for inactive users to reduce the risk of misuse.
  - Ensuring that session identifiers or tokens are not exposed in URLs, logs, or client-side storage.
  - Storing session data securely and invalidating sessions properly upon logout or timeout.
- HTTPS Enforcement: All web applications must use HTTPS (SSL/TLS) encryption for all pages, not just login screens. Certificates must be kept up to date and monitored for expiration or misconfiguration.

Role-Based Access Control (RBAC)

- Access to features and data must be restricted based on user roles and permissions.
- Users must only be able to access functions and data they are authorized to use, based on their department, role, or responsibility.
- Developers must implement RBAC at the application logic level—not just at the user interface level—to prevent unauthorized access via backdoors or APIs.

Multi-Factor Authentication (MFA)

- Multi-Factor Authentication (MFA) adds an extra layer of security by requiring users to verify their identity using two or more factors:

- Something you know (e.g., password)
- Something you have (e.g., mobile device, OTP app, token)
- Something you are (e.g., fingerprint, facial recognition – if applicable)

- MFA is mandatory for all sensitive systems.

Protection Against Common Vulnerabilities

- Developers must proactively safeguard applications against common threats and attacks.
- Developers must:
  - Validate and sanitize all user input
  - Implement output encoding
  - Use parameterized queries for database interactions
  - Avoid exposing internal error messages to end users

Logging and Audit Trails

- All systems must have a comprehensive logging mechanism to track important events such as:
  - User logins and logouts
  - Data access or modifications
  - System errors or unusual activities
- Logs must be stored securely, protected from tampering, and retained.
- For sensitive systems, an audit trail should be implemented to record "who did what and when," enabling traceability in case of incidents or investigations.

4.3.8 Deployment and Hosting

- All systems must be hosted on IIUM's approved infrastructure, either on-premises or cloud-based, managed by KCDIOM.
- Use of containerization (e.g., Docker) and CI/CD pipelines is encouraged to streamline deployment and scalability.
- Backup, recovery, and disaster preparedness measures must be planned and documented.

4.3.9 User Interface and User Experience (UI/UX) Standards

- All IIUM application systems must comply with the official UI/UX guidelines published at https://style.iium.edu.my/ . These standards ensure consistency, usability, and accessibility across all IIUM digital platforms.

- Key requirements:
  - Use the approved color schemes, fonts, and layout styles.
  - Ensure UI is responsive across desktop, tablet, and mobile devices.
  - Apply accessibility features (e.g., contrast ratio, readable text size, keyboard navigation).
  - Avoid custom styles that deviate from institutional branding unless approved by ITD Management.
- All system interfaces must be reviewed during UAT to verify compliance with the university's UI/UX standard.

4.4 Sustainability and Scalability

All application systems developed or implemented at IIUM must be designed with long-term sustainability and future scalability in mind. This ensures that systems remain functional, relevant, and adaptable to changing needs over time, while minimizing the cost and effort required for maintenance or upgrades.

4.4.1 Modularity and Maintainability

- Systems must be built using modular architecture, where functionalities are separated into self-contained components or services.
- Modularity allows individual features or modules to be added, modified, or removed without affecting the entire system.
- This approach supports easier maintenance, testing, and upgrades in response to user feedback, policy changes, or technological advancement.

4.4.2 Clear and Transferable Documentation

- All system documentation—including technical specifications, user manuals, API references, and data models—must be written clearly and maintained regularly.
- Source code must follow established coding standards and include inline comments to explain logic and complex functions.
- This ensures that knowledge can be transferred between teams, reducing dependency on specific developers or vendors and supporting system continuity during personnel changes.

4.4.3 Disaster Recovery and Data Redundancy

- Systems must be designed with a disaster recovery plan (DRP) to ensure service continuity in the event of hardware failure, cyberattacks, or natural disasters.
- Key recovery features must include:
  - Automated and scheduled data backups

- Backup verification and testing procedures
- Redundancy for critical system components (e.g., database replication, failover servers)

- Backup storage must be secured and comply with IIUM's data protection policies, with off-site or cloud-based options considered for added resilience.

4.5 **Governance and Change Control**

4.5.1 Any system enhancement or change must follow IIUM's change management process.

4.5.2 IT Changes must be documented, tested, and approved by the system owner.

4.5.3 Maintain audit trails of all changes and system access.

5. IMPLEMENTATION AND NON-COMPLIANCE

The Director of ITD holds responsibility for the implementation of this guideline and shall take necessary actions in the event of violation of this guideline.

6. ENFORCEMENT

This guideline is applicable to the University community, and any infringement of the guideline may be subject to disciplinary actions and any other actions deemed necessary.

7. MAINTENANCE OF GUIDELINES

The Information Technology Division is responsible for the formulation and maintenance of this guideline.

8. RELATED POLICIES/STANDARDS/PROCEDURES/GUIDELINES

8.1 This guideline shall be read together with the following or any documents as below:

8.1.1 ICT Regulations

8.1.2 IIUM ICT Policy

8.1.3 IIUM ICT Security Procedure

8.1.4 Policy for Management of IT Project