INTER-OFFICE MEMORANDUM

TO	:	ALL EMPLOYEES ASSIGNED TO CORPORATE IT – SYSDEV
SUBJECT	:	RULES AND REGULATIONS
DATE	:	October 25, 2024

In alignment with the IT Sysdev Team Rules and Regulations, all members of our department must adhere to the approved guidelines to ensure optimal performance across each team. Furthermore, any employee who fails to comply with these Rules and Regulations may face disciplinary action for **INSUBORDINATION**.

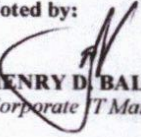The following teams are subject to these Rules and Regulations:

1. Programmers
2. System Analysts
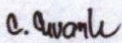3. Record Management System Team (RMS)

Reviewed by:

JENEFFER C. CUTANDA
*IT Sysdev Section Head*

MARK ANTHONY S. RABACA
IT Sysdev Section Head

Noted by:

HENRY D. BALIAR
Corporate IT Manager

MARIA CRISTINA C. EVARLE
IT Sysdev- Jr. Supervisor

Approved by:

MARIA NELIZA U. FUERTES, CPA, CIA, CSCU, CISA, REB, REA, CICA, CRFA, REC
Corporate Audit Group Managing Director

RULES AND REGULATIONS

SYSTEM ANALYST

1. Project Initiation:

- Obtain an approved Information System Request (ISR) from the relevant manager or owner before commencing any project.
- Ensure clear communication of project objectives with the business unit.

2. Business Unit/Department Coordination:

- Schedule and facilitate meetings with business unit representatives, Ma'am Nel or Ma'am Tina, and the FS Process team to discuss system requirements.
- Document all discussions and decisions made during meetings.

3. Documentation and Record-Keeping:

- Maintain an organized record of all project-related documents, including meeting minutes and attendance sheets, and submit them to the supervisor.
- Use recording devices to capture discussions, ensuring accuracy in documentation.

3.1 Document for System Implementation

    3.1.1 Maintain an organized record of all project-related documents for implementation, including the approved ISR, final Data Flow Diagram, System User Guide, final System Flow, implementation memo, and attendance sheets from the training sessions.

    3.1.2 Business Process Acceptance Form, which must be approved by the supervisor and the head users in the business unit.

4. Requirements Gathering:

- Conduct a walkthrough of the process for the business unit requesting the system with the FS Team, and perform a requirements analysis with end-users to understand their needs and expectations.
- Utilize techniques such as interviews and walkthroughs to gather comprehensive information.

5. System Design:

- Create essential design documents, including Data Flow Diagrams (DFD), system flowcharts, and Proposed System Design.

- Ensure designs align with business requirements and technical specifications

6. Project Management:

- Develop and maintain a Gantt chart to outline project timelines, tasks, and deadlines for deliverables.
- Monitor project progress and report any potential delays in the system implementation.

7. Testing and Quality Assurance:

- Collaborate with the development team to obtain and review test results from both local and live system testing processes.
- Ensure that User Acceptance Testing (UAT) is conducted effectively by the development team, with all feedback documented and addressed.

8. System Training and Presentation:

- Conduct a presentation for the Business Unit on the created system prior to implementation.
- Conduct User Acceptance Training (UAT) for end-users, ensuring they are comfortable with the new system.
- Ensure that User Acceptance Testing (UAT) is conducted effectively for end-users, with all feedback documented and addressed, along with the attendance sheet for each training session.

9. System Support and concern

- Provide ongoing support and resources to user's post-implementation.

9.1 Platform for Addressing User Concerns

9.1.1 The RFS form should be used for setting up new users, new items, new customers, and new suppliers. The system analyst must require the user to submit a request, which must be approved before addressing the user's concern.

9.1.2 The TOR form should be used for deleting, editing, or other applicable concerns. The system analyst must require the user to submit a request, which must be approved before addressing the user's concern.

10. Continuous Improvement:

- Gather feedback from users after system implementation to identify areas for improvement.
- Any concerns or feedback from users must be directed to an admin setup for resolution.

11. Verification of System Issues:

- Before responding to user inquiries or concerns, system analysts must verify and conclude whether the issue is indeed a system error. This involves thorough investigation and documentation of findings.
- A system analyst must be the first point of contact for all system concerns and should resolve issues before forwarding them to the programmer.
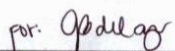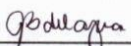
12. Change Management Procedures:

- Any proposed changes to the system must follow established change management procedures. Analysts must consult with supervisors for any changes that deviate from standard processes, regardless of their scale.

13. Compliance and Ethics:

- Adhere to all organizational policies and procedures, including data security and confidentiality standards.
- Collaborate with the programmer to create backups of their code and database.

SYSTEM ANALYST(S):

_____
LUISA MAY RENACIA

_____
BERLY DELA PEÑA

_____
ROSEBELLETTE WAHING

NOTE: SIGNATURE & DATE

Corporate IT

SysDev't

Head Office – Information Technology
System and Development
Rules and Regulations for System Programmers

Table of Contents

---

1. General Responsibilities

- **Accountability**: System Programmers are responsible for the development and maintenance of all project code.

- **Collaboration**: Work closely with project managers, system analyst, FS/Process auditor and other team members to ensure smooth project progress.

- **Deadlines**: Strict adherence to project timelines is required unless unavoidable circumstances arise.

- **Confidentiality**: Protect all proprietary information and code from unauthorized access or disclosure.

2. New Project Guidelines

- **Requirement Analysis**: Thoroughly analyze the project requirements before beginning any coding.

- **Technology Stack Selection**: Use approved programming languages, frameworks, and tools.

  o **Languages**: PHP, JavaScript, FoxPro

  o **Back-End Frameworks**: Laravel, CodeIgniter 3, CodeIgniter 4

- o **Core Web Technologies:** HTML, CSS

- o **Front-End Libraries/Frameworks:** Vue.js, React.js, Bootstrap, Flutter, Angular

- o **Mobile Dev:** Dart, Java (for Android development)

3. Existing Project Guidelines

- **Code Review:** Conduct regular code reviews to ensure that existing code adheres to quality standards and is maintainable.

- **Refactoring:** Improve the internal structure of the code without changing its external behavior to enhance performance, readability, and maintainability.

- **Bug Fixes:** Address bugs or errors in a timely manner and document the resolution process.

- **Feature Updates:** When adding new features to an existing project, ensure backward compatibility to prevent disrupting current functionality. Avoid introducing unnecessary breaking changes. **Avoid making changes to live code unless necessary or on a case-by-case basis. Always prioritize editing in the local or staging environment first, thoroughly testing the updates, and only deploying to the live environment after proper validation.**

4. Project Implementation Guidelines

- **Deployment Environment:** Adhere to the standardized environment for deploying applications. All deployments should be tested in a staging environment before going live.

- **Change Management:** Follow established procedures for introducing changes, including proper documentation and approval workflows.

- **Post-Deployment Monitoring:** Set up monitoring systems to track the performance of deployed applications. Be available to handle post-deployment issues, including downtime or performance degradation.

5. Code Quality Standards

- **Code Readability:** Write clean, understandable, and maintainable code. Code should be well-commented and follow the right naming conventions.

- **Modularization:** Code should be organized into small, reusable modules to avoid redundancy.

- **Efficiency:** Strive for code that is optimized for performance and scalability.

- **Error Handling:** Implement robust error handling to ensure the system continues to function smoothly during unexpected conditions.

6. Documentation

- **Code Documentation**: All code must be accompanied by clear, inline comments explaining complex logic.

- **Project Documentation**: Maintain comprehensive documentation including system architecture, APIs, database schema, and any custom libraries.

- **Update Logs**: Maintain a changelog documenting any updates, bug fixes, or feature additions to the system.

7. Version Control

- **FoxPro & Mobile Projects:** Implement version control to track changes and ensure data integrity. Maintain detailed change logs and establish backup procedures for rollbacks.

- **PHP Projects Upgrade:** Existing Native PHP projects must transition to frameworks like CodeIgniter or Laravel. Code refactoring is required, and database migration should be performed if necessary for compatibility. Thorough testing must be conducted to ensure functionality remains intact. Compliance with this rule is essential for maintaining code quality.

8. Testing and Debugging

- **Unit Testing**: All code must have corresponding unit tests to verify the functionality of individual components.

- **Integration Testing**: Ensure that different modules of the system work together seamlessly. Perform integration testing whenever new features are added or when significant changes are made.

- **Performance Testing**: Test for system scalability, speed, and responsiveness to ensure optimal performance in real-world scenarios.

9. Security Practices

- **Data Security**: Adhere to security standards like data encryption, secure authentication, and protection against SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF) attacks.

- **Access Control**: Implement strict access controls for systems and databases, ensuring that only authorized personnel have access to sensitive data and administrative functions. Access levels should be granted based on role requirements, and regular audits should be performed to review and update permissions as needed.

- **Code Security:** Safeguard the source code by enforcing secure coding practices, including input validation, error handling, and regular vulnerability assessments. All code changes should undergo peer review and security checks before deployment.

10. Communication Protocols

- **Status Reporting**: Provide regular updates to the project manager and team regarding progress, blockers, and next steps

- **Issue Escalation**: Immediately escalate any issues or risks that may delay the project timeline.

- **Collaboration Tools**: Use collaboration tools for team communication, task tracking, and status reporting.

- **Meeting Attendance**: Participate in all scheduled meetings including standups, sprint reviews, and retrospectives.

11. Server Controls

11.1 Local DB Host in Live Server

- **Configuration and Maintenance:** The database hosted on the live server must be configured and maintained to ensure optimum performance and security. Proper partitioning, indexing, and resource allocation must be adhered to avoid overloading the server.

- **Access Restrictions:** Access to the local database on the live server is strictly limited to authorized personnel only. Programmers and administrators must ensure that proper credentials and security measures are in place to prevent unauthorized access.

- **Backup and Recovery:** Regular, automated backups must be performed on the live database and stored in a secure, separate location.

- **Security**: Firewalls, SSL encryption, and database hardening should be enforced to secure the local DB from external threats. Any vulnerabilities must be addressed promptly.

11.2 Remote to Server Permission

- **Controlled Remote Access**: Remote access to the live server is restricted to authorized users only. Remote connections must use SSH keys, and password-based login must be disabled to enhance security.

- **IP Whitelisting:** Remote access must be restricted to specific IP addresses using IP whitelisting. Attempts to access the server from non-whitelisted IPs should be blocked immediately.

- **User Role Assignment:** Permissions and access to the server must be granted based on user roles. Programmers should have limited access for deployment purposes, while system administrators will have higher-level permissions. Root access must only be used when necessary.

11.3   System Backup

- **Regular Backups**: Schedule regular backups for all systems, including databases and application code, to minimize data loss. High-transaction systems may require more frequent backups. High-transaction systems may require more frequent backups, with backups stored onsite (local servers or external drives for quick access).

- **Backup Retention Policy**: Define how long backups will be retained based on regulatory compliance and storage management. Regularly review and prune older backups as needed.

- **Encryption and Security**: Encrypt backups to protect sensitive data during transfer and storage. Restrict access to backup locations to authorized personnel only.

12. Database Schema Practices

1. **Consistent Naming Conventions**: All database objects (tables, columns, indexes, etc.) must follow a consistent naming convention. Use descriptive names that convey the purpose of the object, adopting a standard format (e.g., lowercase with underscores for MySQL: customer_orders, camelCase for FoxPro: CustomerOrders). This enhances readability and maintainability across different database systems.

2. **Normalization and Data Integrity**: Database schemas must be normalized to at least the third normal form (3NF) to minimize redundancy and ensure data integrity. Organize tables to reduce duplication and create appropriate foreign key relationships. For FoxPro projects, utilize referential integrity features to maintain consistency, while in mobile apps, ensure data consistency through local storage best practices.

3. **Version Control and Documentation**: Maintain version control for database schema changes, documenting each change with clear descriptions of modifications. Create migration scripts for MySQL to manage updates, use versioning systems for FoxPro tables, and maintain schema definitions for mobile applications. This practice ensures traceability and facilitates collaboration among team members.
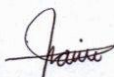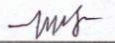
Programmer(s):

_____
GERSHOM DELA TORRE

_____
GEOVANNE DAGANATO

_____
JOMARI GALLEROS

_____
MICHAEL MALATE

_____
MARIA LEONILYN NAIVE

_____
RINO REY SECRETARIA

NOTE: SIGNATURE & DATE