City of Madison, Wisconsin
Information Technology

Application and Database Policies, Standards, and Guidelines

Table of Contents

Introduction

Cyberattacks against government computer networks are unfortunately now common place and have to be prepared for. These attacks result in financial losses, lost productivity, system damage, and lost and/or compromised sensitive data. A single, poorly secured application on the City network can put the entire network at risk. The City seeks to curtail any vulnerabilities during the development and ongoing maintenance of applications and databases. These applications and databases must be constructed and managed with sufficient controls to meet industry best practices, withstand audits, provide maximum data security, and maintain the integrity of the City of Madison's data and sensitive information.

Purpose

The purpose of this document is to establish policies, standards, and guidelines for the development of applications and databases for the City of Madison. The goal of policies is to protect the City's data resources from accidental or intentional unauthorized access or damage, while preserving open information requirements. These policies provide an enforceable governance model for ensuring information security principles of confidentiality, integrity, and availability are upheld. Standards provide City employees with tools for creating and maintaining the City's applications and databases. These standards outline what is essential to maintain a consistent, professional, and compliant environment through industry best practices. Guidelines aid web developers for City applications and databases. Despite being guidelines, it is highly encouraged that users follow them to enhance the visitor experience.

All policies, standards, and guidelines are periodically reviewed and updated as necessary by Information Technology (IT).

Scope

These policies, standards, and guidelines cover all City of Madison applications and databases, regardless of where they are hosted, and anyone developing them.

Exceptions

Any exceptions to these policies or standards require written approval of IT Director or their designated employee. Requests shall be in writing, will state the specific policy or standard that is being challenged, and the business reason for the exception. The decision of the IT Director shall be final.

Consequences for Noncompliance

Any agency, organization, group, application, or database that represents the City of Madison, or that is hosted by the City of Madison, that fails to comply with these policies and standards may be required to take their application or database down until it is in compliance.

Staff Responsibilities

Application Administrator

The Application Administrator is charged with the daily operational aspects of an application. In addition, the Application Administrator works with the Application Owner to develop and enforce appropriate policies and procedures related to the application. The responsibilities of the Application Administrator include, but are not limited to:

- Assists the Application Security Administrator in generating the audit report according to a predetermined schedule.
- Implements procedures authorized by the Application Owner (or their designees).
- Assists the Application Owner with reviewing access rights as frequently as required by applicable policies or laws, or at minimum one (1) time a year, and upon termination or reassignment of any employee with access to the relevant application.
- Coordinates upgrades, configuration changes, and application of patches.
- Create and maintain the configuration files on client servers and development server. Do this for both normal operations and contingency mode operations.
- Primary point-of-contact with customer support, both internally and externally.
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

Applications Development Manager

The responsibilities of the Applications Development Manager include, but are not limited to:

- Ensures the confidentiality, integrity and availability of City information.
- Maintains the information security awareness, training and education program.
- Investigates suspected violations of the Information Security Policy.
- Assists in creating and maintaining standards and procedures related to this policy.
- Oversee all e-government and e-commerce applications.
- Approval of all in-house developed applications and databases.
- Maintaining the Policies, Standards, and Guidelines as necessary.
- Leading the IT Development staff.
- Approval of the use of any outside resources for application and/or database development.
- Approval of the use of any outside developed application and/or database software.
- Approval of the use of any tools purchased to be used in the development or maintenance of any applications and/or databases.
- Provide ongoing security training to development staff.

- Participate in auditing applications and databases for compliance with applicable policies and standards.
- Implements and enforces the Application and Database Policies, Standards, and Guidelines.
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

Application Owner

Application Owners are ultimately responsible for their agency or service's use of an application. The Application Owner should develop and enforce policies and procedures for reviewing and approving data related to their agency or service. The Application Owner is responsible for designating who will be their Application Security Administrators and ensure these individuals receive sufficient training and are sufficiently competent to do the work. The responsibilities of the Application Owner include, but are not limited to:

- Appoints the Application Security Administrator, if applicable.
- Reviews access rights as frequently as required by applicable policies or laws, or at minimum one (1) time a year, and upon termination or reassignment of any employee with access to the relevant application.
- Authorizes access to applications and databases, including direct database and privilege accounts.
- Authorizes data sharing agreements with other applications and entities.
- Sets standards for data access (what roles can access which data).
- Establishes procedures and controls for the application processes.
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

Application Security Administrator

The Application Security Administrator manages the security and privileges to application objects through access rights. The responsibilities of the Application Security Administrator include, but are not limited to:

- Follow established processes for receiving application access requests, and ensure that the appropriate Application Owner or designee has authorized access.
- Ensure the maintenance of any system used for access to the current standard applicable to the most sensitive data in the database.
- Manages access rights at the behest of the Application Owner and Authorized Contact.
- Assists the Application Owner with reviewing access rights as frequently as required by applicable policies or laws, or at minimum one (1) time a year, and upon termination or reassignment of any employee with access to the relevant application.
- Maintains an audit trail of requests and authorizations.

- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

Database Administrators (DBA)

The Database Administrators – not to be confused with Application Administrators – manage the backend database. The responsibilities of DBA's include, but are not limited to:

- Creates role types based on required level of access. This is performed in consultation with the Application Administrator and Application Security Administrator.
- Enables access based on authorization.
- Implements procedures authorized by the Application Owner or their designees.
- Creates audit (exception) reports for privileged accounts and direct database logins (as needed).
- Review audit (exception) report of privileged accounts and direct database logins (as needed).
- In consultation with the Application Security Administrator or designee, authorizes privileged accounts.
- Creates privileged accounts as required.
- Ensure database services are appropriately patched.
- Create new (empty) databases on test database servers.
- Erase old (obsolete) databases on database servers.
- Define and implement database backup procedures.
- Order appropriate database licenses.
- Maintain database license inventory.
- Monitor database license usage.
- Install new licenses.
- Create and maintain scripts to start and shutdown the databases on the database server.
- Create, maintain, and run scripts to manage the databases and perform other routine maintenance functions to provide sufficient database performance and provide for contingency mode operations.
- Create and maintain the configuration files on database server. Do this for both normal operations and contingency mode operations.
- Create and maintain the configuration files on client servers and development server. Do this for both normal operations and contingency mode operations.
- Review database log files on a monthly basis, looking for anomalies.
- Keep monthly statistics, broken down daily, on such things as: database size; database reads/writes; buffer hits; license counts; and user counts

- Coordinate changes to configuration files with developers to improve system performance.
- When necessary – and at least annually – schedule and assist developers with index rebuilds of databases, and other performance enhancing procedures.
- Assist developers with deployments and re-deployments.
- Assist developers with problem resolution.
- Participate with initial installation and upgrades of software.
- Maintain data about systems that would be used to make decisions on operations of databases in a contingency mode.
- Maintain inventory of database accounts (not application accounts), including additions, modifications, deletions with date and time of each.
- Maintain inventory of databases and all data elements across all databases.
- Maintain log files for security, continuity of operations, data integrity, and usage.
- Develop standards and procedures for creating database accounts and DSN's.
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

Developers

The City's Application and Database Developers operate under the direction of the Applications Development Manager or designated team leader. The responsibilities of Developers include, but are not limited to:

- Design, creation, documentation, deployment, modification and maintenance of City applications and databases.
- Adherence to secure coding methods.
- Assist customers with problem resolution.
- Maintain documentation concerning connections to other applications and databases.
- Review application logs on a periodic basis looking for improvement opportunities and issues of concern.
- Participate in the review of applications and databases.
- Participate in the assessment of third-party software, if applicable.
- Participate with initial installation and upgrades of software, if applicable.
- Prepare appropriate technical documentation.
- Provide support for the standard development tools.
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

General Users

The responsibilities of General Users include, but are not limited to:

- Protect the privacy and security of City applications, computer systems, data, and networks under their control.
- Comply with applicable information handling standards (e.g., HIPAA, PCI, etc.).
- Report suspected violations of these policies and standards to the IT Director or to the appropriate Application Owner.
- Never share access or access tokens (e.g., passwords) with any other person.
- Report any change in job responsibilities that changes the need for access to an application or database.
- Appropriately use the application under the direction of the Application Owner.
- Attend appropriate trainings prior to accessing the application.
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

IT Director

The responsibilities of the IT Director include, but are not limited to:

- Acts as data steward for all City information not otherwise assigned.
- Classifies all City information.
- Approves information handling standards.
- Recommends business process or control changes necessary for compliance with this policy.
- Arbitrates disputes related to this policy or related standards and has the final say.
- Ensures that the City conducts periodic risk assessments.
- Conducts periodic review of any standards related to this policy.
- Conducts annual review of the Application and Database Policies, Standards, and Guidelines (or designates employee).
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

Server Administrator

The responsibilities of Server Administrator include, but are not limited to:

- Perform server backups and recoveries.
- Maintain security (user rights) for local accounts on the application servers, the development server, and database servers.

- Appropriate security officers must first approve additions and deletions to local and network accounts:
    - For users outside of development, this comes from the Application Owner and Authorized Contact.
    - For users within IT, this comes from the Applications Development Manager.
- Review server resource usage and performance. Where appropriate, suggest configuration changes to the appropriate development staff.
- Maintain and update the server operating system and communicate changes in coordination with development staff.
- Assist developers with problem resolution.
- Follow the Application and Database Policies, Standards, and Guidelines, and all City policies, APM's and City Ordinances.

Policies, Standards, and Guidelines

Accessibility & ADA Requirements

As a government organization, all application interfaces will be designed to meet requirements of the Americans with Disabilities Act (ADA) and Section 508 standards.

Application & Database Development Tools

All application and database development tool standards are inventoried in the City of Madison Technology Standards Directory.

Application Architecture

All application architecture standards are inventoried in the City of Madison Technology Standards Directory.

Application Authentication

When practical, applications should integrate authentication with Active Directory services.

Application Database Access

As a best practice, the presentation layer (i.e., outermost tier) should not connect to the database directly. Reports and queries should be rendered on a different layer than the presentation layer. Stored procedures and web-services should be used whenever practical.

Application Development Tools

All application development will use the tools outlined in the Technology Standards Directory. Any web browser-based interface shall follow all applicable City of Madison Web Policies, Standards, and Guidelines.

Application Integrations

Where practical, communications between applications shall use a Service Oriented Architecture, and make use of Web Services and JSON or XML tagged data.

Application Stored Procedures

Use stored procedures unless it is not practical to do so (e.g., complex reporting). For routine queries, stored procedures should be utilized. Avoid building queries on the fly within scripts or code procedures. Never accept input parameters into queries without first validating the input to guard against various injection and buffer-overflow attacks.

Application Support

IT will support applications that it develops or approves a request to support. Requests must receive written approval from the IT Director. If an agency contracts out for a system or purchases a system, support must be provided by the contractor or vendor, unless otherwise agreed to by IT.

Database Architecture

Audit control fields

For auditing purposes when applicable, tables should include the following fields:

- Date/time the record was created.
- Date/time last updated.
- User ID of who made the last update (application user ID, not necessarily the database account).
- Record status to indicate whether a record is active or inactive (i.e., “soft delete”).

Database management system (DBMS)

The standard DBMS is Microsoft SQL Server.

Database compartmentalization

Do not create one monolithic database to host the functions of disparate applications. Databases should be aligned directly with the application functions they are serving.

Having separate databases reduces the impact to the organization if an individual database is compromised.

Database tiers

City databases are categorized into the following tiers: (1) General purpose; (2) Supervision; (3) Separation; (4) Isolation; and (5) Deprecation

Full-text indexes

When defining indexes, consider the use of full-text indexes on certain character fields where appropriate, such as names, notes or comments.

Table indexes

Indexes should be defined to aid in data retrieval for enhanced performance for queries and reports. Care should be taken to order the fields properly for multi-field indexes. Do not create multi-field indexes that duplicate the functions of other indexes. For example:

- **Index 1:** Field D, Field B, and Field E
- **Index 2:** Field D and Field B

In this case, Index 2 is redundant and adds unnecessary overhead to the database.

Table keys

Each table shall have a primary, unique key defined, even if it requires the key to use multiple fields. When possible the key shall not be comprised data from the table. A unique key shall be developed along with an algorithm to assign the value. This value shall then be stored as a data element of the table. The use of keys defined by the DBMS – such as row-index – are not permitted. These cannot be counted on to remain stable when database maintenance activities occur. By not using data elements as part of the primary key, those data elements can then be modified if necessary without disrupting the primary key. Additional indexes, unique or not, can be defined.

Table normalization

Database tables shall follow at least the third normal form of data normalization. Exceptions may be made for databases specifically purposed for reporting and data warehousing. In such cases these databases must not be the primary source of data.

Database Controls

Data concurrency controls

Data Concurrency Controls protect against deadlocks when two query or update processes access the same data item at the same time. The database tool must allow for multiple levels of data locking for concurrent users. The default locking status is set

by the DBA as a shared lock. Appropriate locking levels are controlled by application code. When running procedures that only need to read a table but not update it, the code should explicitly read tables without locking them, thus allowing others to also read, or modify tables as needed.

Data validation controls

Ensure the accuracy, completeness and consistency of data that are input and maintained in the database. Table attributes can be assigned properties such as data type and size, data required (y/n), default values, input masks, and validation rules (e.g., value ranges).

Development tool controls

Development tool controls should ensure that only authorized users have access to development tools, to separate developers and general users. This should be done through user security groups.

Encryption controls

The application and database should allow for appropriate levels of encryption for sensitive data elements. If sensitive identification or financial data are stored it should be encrypted in accordance with Payment Card Industry (PCI) standards. Health data shall comply with HIPAA standards.

Existence controls

Protect the existence of the database by establishing backup and recovery procedures. To ensure this the databases at minimum must be on a network folder to be included in the general IT backup process. The database management tools should allow for database integrity controls such as transaction rollback and recovery.

For mobile applications that must be able to function without constant network connectivity, appropriate store-and-forward capabilities should exist within the software. Strong operational procedures should be developed to ensure that data is transferred to and from the mobile device on a consistent and timely basis.

User access controls

These are necessary to prevent unauthorized access to data tables and queries. Proper user access controls have the following characteristics:

- There is not a single password for any and all users to open the database, but instead there is a separate password to view and/or change the application code in general modules or class modules.
- There is a user-level specific login and password with granular permissions down to the database and application object levels (tables, forms, etc.).

Setup requires the usual database security tasks of creating users/passwords/groups, assigning granular permissions to groups and finally assigning users to groups. The security groups and their permissions must be carefully designed to be effective. (Note: These are not network file permissions that are managed by IT. Nor are they internal application permissions which need to be managed by the application security administrator. These are database permissions).

Common practice is to setup an Active Directory Security group and give that group database permissions. This way new users can be add/removed from the Active Directory Security group without a Database Administrator. These security groups should be named to identify the permissions associated in the Database. For example, an Agency Read-Only group might be named ITDatabaseReadOnly to indicate the Agency + Database Name + Permissions.

Direct Database Privileges

Database accounts are at the database level that access the database directly. These are not to be confused with application level accounts. Access to objects will be by roles, except for privileged accounts in special cases. The Database Administrator will create the role types, based on the level of access required.

- The Database Administrator enables authorized access to individuals by roles.
- Individuals with direct database access must use a system that is maintained to the current standard applicable to the most sensitive data contained in the target databases.

The Application Owner or designee will conduct a periodic review of direct database access, as frequently as required by applicable laws.

Application developers and application administrators

- Will have only query privileges in all environments, except development.
- May have update privileges in development environment upon request.
- Must not connect using privileged accounts in any environment except development.
- May have direct access (not via roles) in special cases with the approval of management.
- Must perform updates only by using an intermediate application, and in the case of financially significant or sensitive data, changes must be logged and reviewed by the appropriate data steward or designee.
- Will not be able to alter database schema, except development.

Database developer

- Will have privileged accounts, will be able to alter database schema.

Database service accounts – DSNs and connection strings

There will be a separate account created for each application that accesses a particular database. This enhances auditing and security controls. Assign access to the specified database and Master Database. User ID is in the format COMDatabasenameAccess where access is Administrator (ADM), Read Write (RW), Read Only (RO), Write Only (WO), or Execute (EX). For example ComCouncilRO is a read only user to the council database. A random password generator will be used to generate a random password with alphanumeric, mixed case, and special characters with a minimum length of 14 characters.

Any local configuration settings should not use admin privileges or contain sensitive connection parameters.

Any sensitive connection parameters that must exist in code, such as user ID and password, shall be secured to ensure that they are not accessible to end-users.

End-users

- Will have only query privileges, except by special request with the approval of IT management.
- Must not connect using privileged accounts.
- Direct updates using tools Excel, SQL Management Console, etc. are not permitted.
- Must perform updates only by using an intermediate application, and in the case of financially significant or sensitive data, changes must be logged and reviewed by the appropriate data steward or designee.

Least privileges

All accounts shall follow the principle of least privileges. Only the minimum amount of access rights needed for a role to operate shall be granted. The practice of granting rights to everything shall not be used. Although it is less convenient, it will be more secure to slowly add rights as they are identified.

Special purpose, vendor, third-party

- Will have privileges as needed to support the function.
- May have direct access (not via roles) in special cases with the approval of IT management.
- Privileged accounts that own objects must not log in unless the vendor or third-party application requires it.

Workstation accounts and permissions

Workstation account permissions on the SQL Server database will not be used in favor of using explicit accounts for database activity. Accounts will be associated with Active Directory security group.

Free Software

The use of free software, shareware, open-source code and tools, and code snippets are not allowed unless it can be demonstrated to adhere to our security standards. These items may contain vulnerabilities, including un-secure coding practices, viruses, and other malware. Considerations must be given to the ongoing support and maintenance of any additional software or code. The City must ensure that we comply with any licensing or copyright requirements, and the City Attorney's Office must review and approve any legal terms and conditions.

Any new software must follow the New Software Request Process (APM 3-20). This applies to all software and online services, including free software.

This content is repeated in the Web Policies, Standards, and Guidelines

Input Controls

As a Developer, there should be an assumption that all input is malicious. Before processing any input coming from a user, data source, component, or data service, it should be validated for type, length, and/or range, etc.

Batch & Transactional Controls

Protect integrity of data input by reconciling input number of records, hash totals, and monetary amount totals input to pre-input totals. Database transactions should be scoped to allow for recovery/rollback if an error occurs during processing a batch/transaction.

If there are no available functions for batch/transaction controls; the application must provide custom programming for this to compute, record and display batch record counts, hash totals and monetary totals.

Data Integrity Controls

Provide checks on the integrity of data by ensuring only acceptable values are stored, such as a customer number or inventory number.

One method is to provide values from a drop-down list on the input form. In some cases check digits should be used and if not available as a function, they must be programmed into the application.

Data Validation

Data validation should follow guidelines outlined by OWASP.

These guidelines cover:

- Integrity checks
- Appropriate validation for the appropriate tier
- Methods of business rule validation
- Validation Strategies
  - Accept known good
  - Reject known bad
  - Sanitize – with safelist and with blocklist
- Prevent parameter tampering
- Hidden fields
- URL encoding
- HTML encoding
- Encoded strings
- Data Validation and Interpreter Injection
  - SQL Injection
  - LDAP Injection
  - XML Injection
- Data types
- Delimiter and special characters

Validation Controls

Ensure the accuracy, completeness and consistency of input data items. Input-form data fields should be assigned appropriate properties, such as input masks or range tests. Input-form data fields can be assigned a validation rule and error text. The entire input-form can be secured as read only, edit only, delete only, input only or any combination. Individual input data fields on forms can be secured by disabling and/or lockout. Setting table attribute properties for data validation control at the database level has already been discussed. Input data validation at the application level can also be implemented by creating validation rules and validation error text attached to the data fields on the data input form.

Although the same validation rules in the tables and input form may be redundant, it is preferable to have them in both places. Input validation constraints should match the database validation controls. Some of the data validation controls that can be created for data input fields on the input screen include: range validation, non-null data requirements, input format masks, and default value validation.

Language

The default language of City of Madison applications and databases shall be English. This includes all user-interfaces, reports, data stores, screens, forms, pages, end-user documentation, technical documentation, and information. Providing any of this in other languages, while it may be desirable, is not required unless there is a specific APM, Ordinance, State or Federal Law, or City policy that mandates it.

The City of Madison Language Access Plan defines the goals and expectations for providing information in multiple languages. Most commonly supported languages include English, Spanish, Hmong, and Mandarin (Traditional Chinese).

This content is repeated in the Web Policies, Standards, and Guidelines

Licensing

IT will serve as the central administrator of all licenses for all application and database software used within the City of Madison. Purchasing, upgrading, and renewal of all software licenses must be coordinated with, and receive approval of, IT management. IT may delegate licensing administration responsibilities to another agency.

Naming Conventions

Having standard naming conventions reduces the opportunity for error by eliminating inconsistencies in file names and possible errors when creating reports. To avoid compatibility problems when naming databases, fields, files, folders, and tables, follow these standards:

- Follow industry best practices for case sensitivity (e.g., camel case, all lower case, etc.).
- Use only alphanumeric characters (i.e., a-z and 0-9 in file names).
- Avoid starting a file or folder name with a number.
- Avoid using spaces or special characters (except the underscore or hyphen) in files. Do not use underscores in folder names.
- Use descriptive names; avoid creating unreadable file names.
- **For databases specifically:** do not use reserved words.

Some specifics related to database object types:

- **Tables:** Do not use spaces; underscores can be used in place of spaces.
- **Views:** Do not use spaces; underscores can be used in place of spaces.
- **Stored Procedures:** These are action oriented, so the name should describe how they work.
- **User defined functions:** Use the same convention as for stored procedures.
- **Triggers:** A special kind of stored procedure that require a separate naming convention.
    - Context + Person + Action + Object

- **Indexes:** Since indexes are dependent on the underlying base tables, include the “name of the table” and “column on which it’s built” in the index name. Further, indexes can be of two types – clustered and non-clustered. These two types of indexes could either be unique or non-unique.
    - Table name + column name(s) + unique/non-unique + clustered/non-clustered
- **Columns:** Be descriptive, but do not use spaces; underscores can be used in place of spaces. Do not use table names as part of the column name.
- **User defined data types:** User defined data types are just a wrapper around the base types provided by the DBMS. They are used to maintain consistency of data types across different tables for the same attribute. For example, if the CustomerID column appears in half a dozen tables and you must use the same data type for all occurrences of the CustomerID column. This is where user defined data types come in handy. Just create a user defined data type for CustomerID and use it as the data type for all the occurrences of the CustomerID column.
- **Variables:** Follow the same naming convention as for columns.

Output Controls

Display output controls

Provide proper controls over the displaying and distribution of sensitive data. Permissions can be assigned to individual qualified users through user-level permissions (e.g. most users should not be able to see a social security number on a listing of employees).

Report output controls

Provide proper controls over the production and distribution of reports. Report permissions should be assigned to individual qualified users through user-level permissions.

Processing Controls

Audit trail controls

Provide a log (journal) of database activity of users and application events and data. In the case of financially significant or sensitive data, updates must be performed only by using an intermediate application and changes must be logged showing date, time, and who made the change. In some applications it will even be necessary to log what the change was.

Data concurrency controls

Data Concurrency Controls protect against deadlocks when two query or update processes access the same data item at the same time. The database tool must allow for multiple levels of data locking for concurrent users. The default locking status is set by the Database Administrator (DBA) as a shared lock. Appropriate locking levels are

controlled by application code. When running procedures that only need to read a table but not update it, the code should explicitly read tables without locking them, thus allowing others to also read, or modify tables as needed.

Error messages and logs

Display and record appropriate information for any errors encountered during processing procedures. If there are no available functions for error logs, the application programmer must provide for these.

Run-to-run batch control totals

Protect integrity of data processed by reconciling number of records, hash totals and monetary amount totals before, during and after processing steps, and compare to previously saved batch input totals. If there are no available functions for error logs to report on concerns with batch control totals, the application programmer must provide for these.

Validation and data code controls

Ensure the accuracy, completeness and consistency of data items processed. If there are no available functions at the processing step; the application must provide programming for this. These validate the data coming in and include those controls outlined under Validation Controls and Data Validation listed under Input Controls.

Report Tools

All Report Tools shall follow all Report Tool Policies, Standards, and Guidelines.

Security

All application and database software on any City workstation and servers – regardless of where they are hosted – shall follow secure coding practices. Standard security practices include, but are not limited to:

- Protecting database queries from SQL injections.
- Using SSL to protect secure transmissions of logins and secure data.
- Integrating any objects requiring authentication with Active Directory services.
- Documents should not contain a login ID in the headers, footers, or elsewhere. For example: a document footer with F:\users\itabc\ should not be allowed.
- Document properties should not contain initials or the login ID of the document author.
- Compliance with the City of Madison Network Security Policies and Procedures (APM 3-9 Attachment B).

- Connections to databases and processes must have encrypted connection parameters. They should not be sent via clear text, nor should they be easily visible in application code.

The following security principles should be adhered to:

Avoid security by obscurity

Security through obscurity is a weak security control, and nearly always fails when it is the only control.

The security should rely upon many other factors, including reasonable password policies, defense in depth, business transaction limits, solid network architecture, and fraud and audit controls.

Fail securely

When errors occur, ensure that the application does not disclose any information or expose vulnerabilities.

Minimize attack surface area

Every feature that is added to an application adds a certain amount of risk to the overall application. The aim for secure development is to reduce the overall risk by reducing the attack surface area.

Principle of defense in depth

The principle of defense in depth suggests that where one control would be reasonable, more controls that approach risks in different fashions are better. Controls, when used in depth, can make severe vulnerabilities extraordinarily difficult to exploit and thus unlikely to occur.

With secure coding, this may take the form of tier-based validation, centralized auditing controls, and requiring users to be logged on all pages.

Principle of least privilege

The principle of least privilege dictates that accounts have the least amount of privilege required to perform their business processes.

Separation of duties

A key aspect of organizational security and data integrity is separation of duties. Whenever practical this concept should be employed.

SQL Coding Conventions

Consistent formatting of Transact-SQL code makes it easier for others to scan and understand your code. T-SQL statements have a structure, and having that structure be visually evident makes it much easier to locate and verify various parts of the statements. Uniform formatting also makes it much easier to add sections to and remove sections from complex T-SQL statements for debugging purposes.

Transact-SQL standards:

1. If you use table name aliases, keep the aliases short, but as meaningful as possible.
2. Use appropriate data type declarations and consistent capitalization when declaring T-SQL local variables (such as @lngTableID).
3. Always specify the length of a character data type, and make sure to allow for the maximum number of characters users are likely to need.
4. Always specify the precision and scale of the decimal data type, which otherwise defaults to an unspecified precision and integer scale.
5. Avoid using "undocumented" features such as undocumented columns in system tables, undocumented functionality in T-SQL statements, or undocumented system or extended stored procedures.
6. Do not rely upon any implicit data type conversions. For example, don't assign a character value to a numeric variable assuming that T-SQL will do the necessary conversion. Instead, use the appropriate CONVERT or CAST function to make the data types match before doing a variable assignment or a value comparison.
7. Do not directly compare a variable value to NULL with comparison operators (symbols). If the variable could be null, use IS NULL or IS NOT NULL to do a comparison, or use the ISNULL function.
8. Never depend on a SELECT statement returning rows in any particular order unless the order is specified in an ORDER BY clause.

Additional T-SQL coding practices that should be considered can be found in the Guidelines portion of this document.