**Course Description:**

Students will use developer strategies to manipulate data, present database systems theory, and develop database applications. Students will learn to import and export data, manipulate table properties, make advanced queries, and run basic SQL forms and reports. Students will develop macros for automating database tasks and building menu-driven applications. Knowledge and skills of data modeling, diagraming, query writing, and design theory will be developed.

### Strand 2. IT Fundamentals

###### Learners apply fundamental principles of IT, including the history of IT and its impact on society, common industry terms, systems theory, information storage and retrieval, database management, and computer hardware, software, and peripheral device configuration and installation. This base of knowledge and skills may be applied across the career field.

**Outcome 2.4. Emerging Technologies**

Identify trending technologies, their fundamental architecture, and their value in the marketplace.

**Competencies**

2.4.2. Describe the differences, advantages, and limitations of cloud computing (e.g., public cloud, private cloud, hybrid cloud) and on premises computing.

2.4.3. Utilize cloud computing applications (e.g., services, applications, virtual environments).

**Outcome: 2.8. Databases**

Describe the fundamentals of databases.

**Competencies**

2.8.1. Identify emerging database technology (e.g., Not only Structured Query Language [NoSQL], New Structured Query Language [NewSQL], graph databases).

2.8.2. Describe the use and purpose of a database and a Database Management System (DBMS).

2.8.3. Compare databases (e.g., flat file, hierarchical, relational).

2.8.4. Describe the elements of a database (e.g., table, record/row, field, relationships, transactions).

2.8.5. Describe the elements of a database user interface (e.g., form, queries, filters, reports).

2.8.6. Describe Structured Query Language (SQL)

2.8.7. Describe how data can be stored in and extracted from a database.

2.8.8. Explain the importance of data integrity and security.

2.8.9. Differentiate between a front-end interface and a back-end

**Outcome: 2.9. Project Concept Proposal**

Develop a project concept proposal.

**Competencies**

2.9.1. Identify and incorporate branding strategies.

2.9.2. Determine the scope and purpose of the project.

2.9.3. Determine the target audience, client needs, expected outcomes, objectives, and budget.

2.9.4. Develop a conceptual model and design brief for the project.

2.9.5. Develop a timeline, a communication plan, a task breakdown, costs (e.g., equipment, labor), deliverables, and responsibilities for completion.

2.9.6. Develop and present a comprehensive proposal to stakeholders.

**Outcome: 2.11. Troubleshooting**

Select and apply troubleshooting methodologies for problem solving.

**Competencies**

2.11.1. Identify the problem.

2.11.2. Select troubleshooting methodology (e.g., top down, bottom up, follow the path, spot the differences).

2.11.3. Investigate symptoms based on the selected methodology.

2.11.4. Gather and analyze data about the problem.

2.11.5. Design a solution.

2.11.6. Test a solution.

2.11.7. Implement a solution.

2.11.8. Document the problem and the verified solution.

**Outcome: 2.12. Performance Tests and Acceptance Plans**

Develop performance tests and acceptance plans.

**Competencies**

2.12.1. Create a written procedure agreed by the stakeholders and project team for determining the acceptability of the project deliverables.

2.12.2. Develop a test system that accurately mimics external interfaces.

2.12.3. Develop test cases that are realistic, compare with expected performance, and include targeted platforms and device types.

2.12.4. Develop, perform, and document usability and testing integration.

2.12.5. Make corrections indicated by test results.

2.12.6. Seek stakeholder acceptance upon successful completion of the test plan.

**Outcome: 2.13. Rollout and Handoff**

Plan rollout and facilitate handoff to customer.

**Competencies**

2.13.1. Include overall project goals and timelines in the rollout plan.

2.13.2. Communicate rollout plans to key stakeholders in a timely manner.

2.12.3. Conduct final review and approvals according to company standards.

2.13.4. Identify support staff, training needs, and contingency plans in the rollout plan.

2.13.5. Test delivered application to assure that it is fully functional for the customer or user and meets all requirements.

2.13.6 Deliver support and training materials.

**Strand 5. Programming Concepts**

Learners apply principles of computer programming and software development to develop code; build, test, and debug programs; create finished products; and plan, analyze, design, develop, implement, and support software applications.

**Outcome: 5.1. Programming Concepts**

Describe programming concepts.

5.1.1. Describe how computer programs and scripts can be used to solve problems (e.g., desktop, mobile, enterprise).

5.1.2. Explain how algorithms and data structures are used in information processing.

5.1.3. Model the solution using both graphic tools (e.g., flowcharts) and pseudocode techniques.

5.1.4. Describe, compare, and contrast the basics of procedural, structured, object-oriented (OO), and event-driven programming.

5.1.5. Describe the concepts of data management through programming languages.

5.1.6. Analyze the strengths and weaknesses of different languages for solving a specific problem.

5.1.7. Compare the functions and operations of compilers and interpreters.

5.1.8. Describe version control and the relevance of documentation.

**Outcome: 5.2. Computational and String Operations**

Develop code that performs computational and string operations.

**Competencies**

5.2.1. Compare primitive types of numeric and nonnumeric data (e.g., integers, floats, Boolean, strings).

5.2.2. Identify the scope of data (e.g., global versus local, variables, constants, arrays).

5.2.3. Write code that uses arithmetic operations.

5.2.4. Write code that uses subtotals and final totals.

5.2.5. Write code that applies string operations (e.g., concatenation, pattern matching, substring).

**Outcome: 5.3. Logical Operations and Control Structures**

Develop code that uses logical operations and control structures.

**Competencies**

5.3.1. Explain Boolean logic.

5.3.2. Solve a truth table.

5.3.3. Write code that uses logical operators (e.g., and, or, not).

5.3.4. Write code that uses relational operators and compound conditions.

5.3.5. Write code that uses conditional control structures (e.g., if, if-then-else).

5.3.6. Write code that uses repetition control structures (e.g., while, for).

5.3.7. Write code that uses selection control structures (e.g., case, switch).

5.3.8. Write code that uses nested structures and recursion.

5.3.9. Write code that creates and calls functions.

5.3.10. Code error handling techniques.

5.3.11. Write code to access data repositories.

5.3.12. Write code to create classes, objects, and methods.

**Outcome: 5.4. Integrated Development Environment**

Build and test a program using an integrated development environment (IDE).

**Competencies**

5.4.1. Configure options, preferences, and tools.

5.4.2. Write and edit code in the integrated development environment (IDE).

5.4.3. Compile or interpret a working program.

5.4.4. Define test cases.

5.4.5. Test the program using defined test cases.

5.4.6. Correct syntax and runtime errors.

5.4.7. Debug logic errors.

**Outcome: 5.5. Programming Conventions**

Develop programs using applications security practices.

**Competencies**

5.5.1. Develop programs using data validation techniques.

5.5.2. Develop programs that use reuse libraries.

5.5.3. Develop programs using operating system calls.

5.5.4. Develop programs that call other programs.

5.5.5. Use appropriate naming conventions and apply comments.

5.5.6. Format output (e.g., desktop, mobile, enterprise, reports, data files).

**Outcome: 5.6. Software Development Lifecycle**

Apply the software development lifecycle (SDLC).

**Competencies**

5.6.1. Determine requirements specification documentation.

5.6.2. Identify constraints and system processing requirements.

5.6.3. Develop and adhere to timelines.

5.6.4. Identify a programming language, framework, and an integrated development environment (IDE).

5.6.5. Identify input and output (I/O) requirements.

5.6.6. Design system inputs, outputs, and processes.

5.6.7. Document a design using the appropriate tools (e.g., program flowchart, dataflow diagrams, Unified Modeling Language [UML]).

5.6.8. Create documentation (e.g., implementation plan, contingency plan, data dictionary, user help).

5.6.9. Review the design (e.g., peer walkthrough).

5.6.10. Present the system design to stakeholders.

5.6.11. Develop the application.

5.6.12. Compare software methodologies (e.g., agile, waterfall).

5.6.13. Perform code reviews (e.g., peer walkthrough, static analysis).

5.6.14. Ensure code quality by testing and debugging the application (e.g., system testing, user acceptance testing).

5.6.15. Train stakeholders.

5.6.16. Deploy the application.

5.6.17. Collect application feedback and maintain the application.

**Outcome: 5.7. Configuration/Change Management**

Describe configuration management activities.

**Competencies**

5.7.1. Explain version management and interface control.

5.7.2. Explain baseline and software lifecycle phases.

5.7.3. Analyze the impact of changes.

**Strand 8. Databases**

Learners apply principles of designing, creating, and maintaining databases, including data storage, retrieval, modeling, manipulation, and formatting; database access, management, and administration; and database hardware and software issues.

**Outcome 8.1. Data Modeling**

Develop a data model to describe an application’s data.

**Competencies**

8.1.1. Develop specifications for a database in consultation with the client.

8.1.2. Identify the real-world entities (e.g., customers, purchases) to put in a table and the attributes of the entities (e.g., customer names and addresses, purchase dates and amounts).

8.1.3. Identify the relationships between database entities.

8.1.4. Determine the data types (e.g., text, numbers) and domains (e.g., number greater than zero, text string of two letters) of attributes.

8.1.5. Determine whether attributes allow for null value.

8.1.6. Determine unique identifiers (i.e., keys) of entities.

8.1.7. Normalize data model as appropriate for the application.

8.1.8. Generate data modeling documentation (e.g., entity-relationship, workflow, Unified Modeling Language [UML]).

8.1.9. Verify that the data model matches specifications.

**Outcome 8.2. Design and Creation**

Design and create databases.

**Competencies**

8.2.1 Design and create database tables and relationships.

8.2.2 Create database columns and specify properties (e.g., name, type, domain).

8.2.3. Name tables and fields in conformance with naming conventions.

8.2.4. Define indexes as appropriate for the application.

**Outcome 8.3. Data Entry and Access**

Enter and access data in databases.

**Competencies**

8.3.1 Insert, update, and delete records.

8.3.2 Enter and bulk import data into databases and transfer data between databases.

8.3.3 Write Structured Query Language (SQL) scripts and stored procedures.

8.3.4 Retrieve, filter, sort, and parse data.

8.3.5 Commit and roll back transactions.

8.3.6 Generate and print forms, reports, and results of queries (e.g., calculated fields, functions).