Exam MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer – Skills Measured

---

The English language version of this exam will be updated on April 22, 2022.

Following the current exam guide, we have included a version of the exam guide with Track Changes set to "On," showing the changes that will be made to the exam on that date.

NOTE: Passing score: 700. Learn more about exam scores.

Audience Profile

Candidates for this exam are developers who work with Finance and Operations apps in Microsoft Dynamics 365 to implement and extend applications that meet the requirements of a business. Candidates provide fully realized solutions by using standardized application coding patterns, extensible features, and external integrations.

Candidates develop business logic by using X++, create and modify Finance and Operations app reports and workspaces, customize user interfaces, provide endpoints and APIs to support Power Platform apps and external systems, perform testing, monitor performance, analyze and manipulate data, create technical designs and implementation details, and implement permission policies and security requirements.

Candidates should have a deep knowledge and experience using the underlying framework, data structures, and objects associated with the Finance and Operations solutions.

Candidates should have experience with products that include Visual Studio, Azure DevOps, LCS tools, and SQL Server Management Studio.

Skills Measured

NOTE: The bullets that follow each of the skills measured are intended to illustrate how we are assessing that skill. Related topics may be covered in the exam.

NOTE: Most questions cover features that are general availability (GA). The exam may contain questions on Preview features if those features are commonly used.

Plan the architecture and solution design (10-15%)

Identify the major components of Dynamics 365 Finance and Operations apps

- identify Finance and Operations app modules required for a solution based on business requirements
- identify architectural differences between the cloud and on-premises versions of Finance and Operations apps
- identify components of the application stack
- differentiate the purposes and interrelationships between packages, projects, models, and elements

Design and implement a user interface

- describe the Finance and Operations user interface layouts and components
- design workspaces
- design and personalize user interface elements including grids, forms, and pages
- configure filtering options

Implement Application Lifecycle Management (ALM)

- describe the capabilities of the Environment Monitoring Tool within Lifecycle Services (LCS)
- select the purpose and appropriate uses of LCS tools and components
- research and resolve issues by using Issue Search
- identify activities that require asset libraries
- prepare deployment packages and deploy packages

Apply developer tools (10-15%)

Customize Finance and Operations apps by using Visual Studio

- create extension models
- design and build projects
- manage metadata using Application Explorer
- synchronize data dictionary changes with the application database
- create elements by using the element designer

Manage source code and artifacts by using version control

- create, check out, and check in code and artifacts
- branch and merge code
- compare code and resolve version conflicts

Implement Finance and Operations app framework functionality

- implement the SysOperation framework
- implement the asynchronous framework

- implement the workflow framework
- implement the unit test framework

Design and develop AOT elements (20-25%)

Create forms

- add a new form to a project and apply a pattern (template)
- configure a data source for the form
- add a grid and grid fields, groups, and other controls to a form
- create and populate menu items
- test form functionality and data connections
- add a form extension to a project for selected standard forms

Create and extend tables

- add tables to a project
- add table fields and field properties to a table
- add field groups, relations, delete actions, and indices
- add a table extension to a project for a table

Create Extended Data Types (EDT) and enumerations

- add an EDT to a project and populate EDT properties
- add an enumeration to a project
- add or update enumeration elements and enumeration element properties
- add an extension of EDT and enumerations

Create classes and extend AOT elements

- add a new class to a project
- create a new class extension and add new methods
- add event handler methods to a class
- add attributes to a class

Develop and test code (10-15%)

Develop X++ code

- identify and implement base types and operators
- implement common structured programming constructs of X++
- create, read, update, and delete (CRUD) data
- identify and implement global functions in X++
- implement table and form methods

Develop object-oriented code

- implement X++ variable scoping
- implement inheritance and abstraction concept
- implement query objects and the QueryBuilder class
- implement attribute classes
- implement chain of command

Implement reporting (10-15%)

Describe the capabilities and limitations of reporting tools in Finance and Operations apps

- create and modify report data sources and supporting classes
- implement reporting security requirements
- describe the report publishing process
- describe the differences between using Entity store and Bring your own database (BYOD) as reporting data stores

Design, create, and revise Dynamics reports

- create and modify reports in Finance and Operations apps that use SQL Server Reporting Services (SSRS)
- create and modify Finance and Operations apps reports by using Power BI
- create and modify Finance and Operations apps reports by using Microsoft Excel

Design, create, and revise Dynamics workspaces

- design KPIs
- create drill-through workspace elements
- implement built-in charts, aggregate measurements, aggregate dimensions, and other reporting components

Integrate and manage data solutions (10-15%)

Identify data integration scenarios

- select an appropriate data integration API
- identify differences between synchronous vs. asynchronous patterns

Implement data integration concepts and solutions

- develop a data entity by using Visual Studio
- develop, import, and export composite data entities
- identify and manage unmapped fields in data entities
- consume external web services by using OData and RESTful APIs

- integrate Finance and Operations apps with Microsoft Excel by using OData
- develop and integrate Power Automate and Power Apps with Finance and Operations

Implement data management

- import and export data using entities between Finance and Operations apps and other systems
- monitor the status and availability of entities
- enable Entity Change Tracking
- set up a data project and recurring data jobs
- design entity sequencing
- generate field mapping between source and target data structures
- develop data transformations

Implement security and optimize performance (10-15%)

Implement role-based security policies and requirements

- create or modify duties, privileges, permissions, and roles
- enforce permissions policies
- implement record-level security by using Extensible Data Security (XDS)

Apply fundamental performance optimization techniques

- identify and apply caching mechanisms for forms and tables
- implement the global cache
- create or modify temporary tables for optimization purposes
- determine when to use set-based queries and row-based queries
- modify queries to optimize performance
- modify variable scope to optimize performance
- analyze and optimize concurrency

Optimize user interface performance

- capture traces by using TraceParser and analyze traces
- diagnose and optimize client performance by using Microsoft Edge F12 Developer tools, Fiddler, and other common tools
- diagnose and optimize client performance by using Performance Timer

---

The following exam guide shows the changes that will be implemented on April 22, 2022 to the English language version of this exam.

Audience Profile

Candidates for this exam are developers who work with finance and operations apps in Microsoft Dynamics 365 to implement and extend applications that meet the requirements of a business. Candidates provide fully realized solutions by using standardized application coding patterns, extensible features, and external integrations.

Candidates develop business logic by using X++, create and modify finance and operations app reports and workspaces, customize user interfaces, provide endpoints and APIs to support Microsoft Power Platform apps and external systems, perform testing, monitor performance, analyze and manipulate data, create technical designs and implementation details, and implement permission policies and security requirements.

Candidates should have a deep knowledge and experience using the underlying framework, data structures, and objects associated with the finance and operations solutions.

Candidates should have experience with products that include Visual Studio, Azure DevOps Lifecycle Services (LCS) tools, Postman, GitHub, Microsoft 365, and SQL Server Management Studio.

Skills Measured

NOTE: The bullets that follow each of the skills measured are intended to illustrate how we are assessing that skill. Related topics may be covered in the exam.

NOTE: Most questions cover features that are general availability (GA). The exam may contain questions on Preview features if those features are commonly used.

Plan the architecture and solution design (~~10-15~~5-10%)

Identify the major components of Dynamics 365 finance and operations apps

- identify finance and operations app modules required for a solution based on business requirements
- identify architectural differences between the cloud and on-premises versions of finance and operations apps
- identify components of the application stack
- differentiate the purposes and interrelationships between packages, projects, models, and elements

Design and implement a user interface

- describe the finance and operations user interface layouts and components
- design workspaces
- design and personalize user interface elements including grids, forms, and pages
- define navigation elements including menus and menu items
- configure filtering options and saved views

Implement Application Lifecycle Management (ALM) and Lifecycle Services (LCS)

- describe the capabilities of the Environment Monitoring Tool within Lifecycle Services (LCS)
- select the purpose and appropriate uses of LCS tools and components
- research and resolve issues by using Issue Search
- identify activities that require asset libraries
- prepare deployment packages and deploy packages

Apply developer tools (10-15%)

Customize finance and operations apps by using Visual Studio

- create extension models
- design and build projects
- manage metadata using Application Explorer
- synchronize data dictionary changes with the application database
- create elements by using the eElement designers

Manage source code and artifacts by using version control

- create, check out, and check in code and artifacts
- branch and merge code
- compare code and resolve version conflicts

Implement finance and operations app framework functionality

- implement the SysOperation framework
- implement the SysExtension framework
- implement the asynchronous framework
- implement the workflow framework
- implement the unit test framework

Design and develop AOT elements (20-25%)

Create forms

- add a new form to a project and apply a pattern (template)

- configure a data source for the form
- add a grid and grid fields, groups, and other controls to a form
- create and populate menu items
- test form functionality and data connections
- add a form extension to a project for selected standard forms

Create and extend tables

- add tables to a project
- add table fields and field properties to a table
- add field groups, relations, delete actions, methods, and indices
- add a table extension to a project for a table

Create Extended Data Types (EDT) and enumerations

- add an EDT to a project and populate EDT properties
- add an enumeration to a project
- add or update enumeration elements and enumeration element properties
- add an extension of EDT and enumerations

Create classes and extend AOT elements

- add a new class to a project
- create a new class extension and add new methods
- add event handler methods to a class
- add attributes to a class

Develop and test code (10-15%)

Develop X++ code

- identify and implement base types and operators
- implement common structured programming constructs of X++
- create, read, update, and delete (CRUD) data
- identify and implement global functions in X++
- implement table and form methods

Develop object-oriented code

- implement X++ variable scoping
- implement inheritance and abstraction concept
- implement query objects and the QueryBuilder class
- implement attribute classes
- implement chain of command and wrapper classes

Implement reporting (10-15%)

Describe the capabilities and limitations of reporting tools in finance and operations apps

- create and modify report data sources and supporting classes
- implement reporting security requirements
- describe the report publishing process
- describe the differences between using Entity store and Bring your own database (BYOD) or Azure Data Lake as reporting data stores

Design, create, and revise Dynamics reports

- create and modify reports in finance and operations apps that use SQL Server Reporting Services (SSRS)
- create and modify finance and operations apps reports by using Power BI
- create and modify finance and operations apps reports by using Microsoft Excel
- implement Business Document Management

Design, create, and revise Dynamics workspaces

- design and implement KPIs
- Identify data integration patterns
- create drill-through workspace elements
- implement built-in charts, Power BI embedded visualizations, aggregate measurements, aggregate dimensions, and other reporting components

Integrate and manage data solutions (10-15%)

Identify data integration patterns and scenarios

- identify data integration patterns
- select an appropriate data integration API
- identify differences between synchronous vs. asynchronous patterns

Implement data integration concepts and solutions

- develop a data entity by using Visual Studio
- develop, import, and export composite data entities
- identify and manage unmapped fields in data entities
- consume external web services by using ~~OData, and~~ RESTful APIs, and SOAP
- integrate finance and operations apps with Microsoft Excel by using OData
- ~~develop and i~~Integrate Power Automate, Power BI, and Power Apps with finance and operations apps
- implement global electronic reporting

- implement custom services, Batch API, business events, and Azure Logic Apps
- integrate Microsoft Dataverse with finance and operations apps by using dual-write or virtual entities

Implement data management

- import and export data using entities between finance and operations apps and other systems
- monitor the status and availability of entities
- enable Entity Change Tracking
- set up a data project and recurring data jobs
- design entity sequencing
- generate field mapping between source and target data structures
- develop data transformations

Implement security and optimize performance (10-15%)

Implement role-based security policies and requirements

- create or modify duties, privileges, permissions, and roles
- implement segregation of duties
- enforce permissions policies
- implement record-level security by using Extensible Data Security (XDS)

Apply fundamental performance optimization techniques

- identify and apply caching mechanisms for forms and tables
- implement the global cache
- create or modify temporary tables for optimization purposes
- determine when to use set-based queries and row-based queries
- modify queries to optimize performance
- modify variable scope to optimize performance
- analyze and optimize concurrency

Optimize user interface performance

- capture traces by using TraceParser and analyze traces
- diagnose and optimize client performance by using Microsoft Edge F12 Developer tools, Fiddler, and other common tools
- diagnose and optimize client performance by using Performance Timer
- optimize performance for data entities, data source queries, batch processes, and reports