NUTECH COMPUTER TRAINING INSTITUTE

1682 E. GUDE DRIVE #102, ROCKVILLE, MD 20850

WEB: www.NUTECHTRAINING.COM TEL: 301-610-9300

MCSD Web Applications – Course Outlines

70-487 Developing Microsoft Azure and Web Services

Toptic 1. Accessing data:

- Choose data access technologies
  - Choose a technology (ADO.NET, Entity Framework, WCF Data Services, Azure storage) based on application requirements
- Implement caching
  - Cache static data, apply cache policy (including expirations); use CacheDependency to refresh cache data; query notifications
- Implement transactions
  - Manage transactions by using the API from System.Transactions namespace; implement distributed transactions; specify transaction isolation level
- Implement data storage in Azure
  - Access data storage in Azure; choose data storage mechanism in Azure (blobs, tables, queues, SQL Database); distribute data by using the Content delivery network (CDN); handle exceptions by using retries (SQL Database); manage Azure Caching
- Create and implement a WCF Data Services service
  - Address resources; implement filtering; create a query expression; access payload formats (including JSON); use data service interceptors and service operators
- Manipulate XML data structures

- Read filter, create, modify XML data structures; Manipulate XML data by using XMLReader, XMLWriter, XMLDocument, XPath, LINQ to XML; transform XML by using XSLT transformations

Toptic 2. Query and manipulate data by using the Entity Framework

-
    - Query, update, and delete data by using DbContext; build a query that uses deferred execution; implement lazy loading and eager loading; create and run compiled queries; query data by using Entity SQL; perform asynchronous operations using Entity Framework; map a stored procedure
- Query and manipulate data by using Data Provider for Entity Framework
    - Query and manipulate data by using Connection, DataReader, and Command from the System.Data.EntityClient namespace; perform synchronous and asynchronous operations; manage transactions (API); programmatically configure a Data Provider
- Query data by using LINQ to Entities
    - Query data by using LINQ operators (for example, project, skip, aggregate, filter, and join); log queries and database commands; implement query boundaries (IQueryable vs. IEnumerable); implement async query
- Query and manipulate data by using ADO.NET
    - Query and manipulate data by using Connection, DataReader, Command, DataAdapter, DataSet; perform synchronous and asynchronous operations; manage transactions (API)
- Create an Entity Framework data model
    - Structure the data model using table per type, table per class, table per hierarchy; choose and implement an approach to manage a data model (code first vs. model first vs. database first); implement POCO objects; describe a data model by using conceptual schema definitions, storage schema definition, mapping language (CSDL, SSDL, MSL), and Custom Code First Conventions

Toptic 3. Designing and implementing WCF Services:

- Create a WCF service
    - Create contracts (service, data, message, callback, and fault); implement message inspectors; implement asynchronous operations in the service
- Configure WCF services by using configuration settings

    - Configure service behaviors; configure service endpoints; configure bindings including WebSocket bindings; specify a service contract; expose service metadata (XSDs, WSDL, and metadata exchange endpoint); configure message compression and encoding
- Configure WCF services by using the API
    - Configure service behaviors; configure service endpoints; configure binding; specify a service contract; expose service metadata (XSDs, WSDL, and metadata exchange); WCF routing and discovery features
- Secure a WCF service
    - Implement message level security, implement transport level security; implement certificates; design and implement multiple authentication modes
- Consume WCF services
    - Generate proxies by using SvcUtil; generate proxies by creating a service reference; create and implement channel factories
- Version a WCF service
    - Version different types of contracts (message, service, data); configure address, binding, and routing service versioning
- Create and configure a WCF service on Azure
    - Create and configure bindings for WCF services (Azure SDK—extensions to WCF); relay bindings to Azure using service bus endpoints; integrate with the Azure service bus relay
- Implement messaging patterns
    - Implement one way, request/reply, streaming, and duplex communication; implement Azure Service Bus and Azure Queues
- Host and manage services
    - Manage services concurrency (single, multiple, reentrant); create service hosts; choose a hosting mechanism; choose an instancing mode (per call, per session, singleton); activate and manage a service by using AppFabric; implement transactional services; host services in an Azure worker role

Toptic 4. Creating and consuming Web API-based services:

- Design a Web API
    - Define HTTP resources with HTTP actions; plan appropriate URI space, and map URI space using routing; choose appropriate HTTP method (get, put, post, delete) to meet requirements;

choose appropriate format (Web API formats) for responses to meet requirements; plan when to make HTTP actions asynchronous; design and implement routes

- Implement a Web API
    - Accept data in JSON format (in JavaScript, in an AJAX callback); use content negotiation to deliver different data formats to clients; define actions and parameters to handle data binding; use HttpMessageHandler to process client requests and server responses; implement dependency injection, along with the dependency resolver, to create more flexible applications; implement action filters and exception filters to manage controller execution; implement asynchronous and synchronous actions; implement streaming actions; implement SignalR; test Web API web services
- Secure a Web API
    - Implement HTTPBasic authentication over SSL; implement Windows Auth; prevent cross-site request forgery (XSRF); design, implement, and extend authorization and authentication filters to control access to the application; implement Cross Origin Request Sharing (CORS); implement SSO by using OAuth 2.0; configure multiple authentication modes on a single endpoint
- Host and manage Web API
    - Host Web API in an ASP.NET app; self-host a Web API in your own process (a Windows service) including Open Web Interface for .NET (OWIN); host services in an Azure worker role; restrict message size; configure the host server for streaming
- Consume Web API web services
    - Consume Web API services by using HttpClient synchronously and asynchronously; send and receive requests in different formats (JSON/HTML/etc.); request batching

Toptic 5. Deploying web applications and services:

- Design a deployment strategy
    - Create an IIS install package; deploy to web farms; deploy a web application by using XCopy; automate a deployment from TFS or Build Server
- Choose a deployment strategy for an Azure web application
    - Perform an in-place upgrade and VIP swap; configure an upgrade domain; create and configure input and internal endpoints; specify operating system configuration; deploy applications using Azure Web Site
- Configure a web application for deployment

    - Switch from production/release mode to debug mode; use SetParameters to set up an IIS app pool; set permissions and passwords; enable and monitor ASP.NET App Suspend; configure WCF endpoints (including HTTPS protocol mapping), bindings, and behaviors; transform web.config by using XSLT (for example, across development, test, and production/release environments); configure Azure configuration settings
- Manage packages by using NuGet
    - Create and configure a NuGet package; install and update an existing NuGet package; connect to a local repository cache for NuGet, set up your own package repository
- Create, configure, and publish a web package
    - Create an IIS InstallPackage; configure the build process to output a web package; apply pre- and post- condition actions to ensure that transformations are correctly applied; include appropriate assets (web content, certificates)
- Share assemblies between multiple applications and servers
    - Prepare the environment for use of assemblies across multiple servers (interning); sign assemblies by using a strong name; deploy assemblies to the global assembly cache; implement assembly versioning; create an assembly manifest; configure assembly binding redirects (for example, from MVC4 to MVC5)