Microsoft

R CAT
Rajiv Gandhi Centre of Advanced Technology

Exam AI-102: Designing and Implementing a Microsoft Azure AI Solution – Skills Measured

Audience Profile

Candidates for Exam AI-102: Designing and Implementing a Microsoft Azure AI Solution build, manage, and deploy AI solutions that leverage Azure Cognitive Services, Azure Cognitive Search, and Microsoft Bot Framework.

Their responsibilities include participating in all phases of AI solutions development—from requirements definition and design to development, deployment, maintenance, performance tuning, and monitoring.

They work with solution architects to translate their vision and with data scientists, data engineers, IoT specialists, and AI developers to build complete end-to-end AI solutions.

Candidates for this exam should be proficient in C#, Python, or JavaScript and should be able to use REST-based APIs and SDKs to build computer vision, natural language processing, knowledge mining, and conversational AI solutions on Azure.

They should also understand the components that make up the Azure AI portfolio and the available data storage options. Plus, candidates need to understand and be able to apply responsible AI principles.

Skills Measured

NOTE: The bullets that appear below each of the skills measured are intended to illustrate how we are assessing that skill. This list is not definitive or exhaustive.

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 and Manage an Azure Cognitive Services Solution (15-20%)

Select the appropriate Cognitive Services resource

- select the appropriate cognitive service for a vision solution
- select the appropriate cognitive service for a language analysis solution
- select the appropriate cognitive Service for a decision support solution
- select the appropriate cognitive service for a speech solution

Plan and configure security for a Cognitive Services solution

- manage Cognitive Services account keys
- manage authentication for a resource
- secure Cognitive Services by using Azure Virtual Network
- plan for a solution that meets responsible AI principles

Create a Cognitive Services resource

- create a Cognitive Services resource
- configure diagnostic logging for a Cognitive Services resource
- manage Cognitive Services costs
- monitor a cognitive service
- implement a privacy policy in Cognitive Services

Plan and implement Cognitive Services containers

- identify when to deploy to a container
- containerize Cognitive Services (including Computer Vision API, Face API, Text Analytics, Speech, Form Recognizer)

Implement Computer Vision Solutions (20-25%)

Analyze images by using the Computer Vision API

- retrieve image descriptions and tags by using the Computer Vision API
- identify landmarks and celebrities by using the Computer Vision API
- detect brands in images by using the Computer Vision API
- moderate content in images by using the Computer Vision API
- generate thumbnails by using the Computer Vision API

Extract text from images

- extract text from images by using the OCR API
- extract text from images or PDFs by using the Read API
- convert handwritten text by using Ink Recognizer
- extract information from forms or receipts by using the pre-built receipt model in Form Recognizer
- build and optimize a custom model for Form Recognizer

Extract facial information from images

- detect faces in an image by using the Face API

- recognize faces in an image by using the Face API
    - configure persons and person groups
- analyze facial attributes by using the Face API
- match similar faces by using the Face API

Implement image classification by using the Custom Vision service

- label images by using the Computer Vision Portal
- train a custom image classification model in the Custom Vision Portal
- train a custom image classification model by using the SDK
- manage model iterations
- evaluate classification model metrics
- publish a trained iteration of a model
- export a model in an appropriate format for a specific target
- consume a classification model from a client application
- deploy image classification custom models to containers

Implement an object detection solution by using the Custom Vision service

- label images with bounding boxes by using the Computer Vision Portal
- train a custom object detection model by using the Custom Vision Portal
- train a custom object detection model by using the SDK
- manage model iterations
- evaluate object detection model metrics
- publish a trained iteration of a model
- consume an object detection model from a client application
- deploy custom object detection models to containers

Analyze video by using Video Indexer

- process a video
- extract insights from a video
- moderate content in a video
- customize the Brands model used by Video Indexer
- customize the Language model used by Video Indexer by using the Custom Speech service
- customize the Person model used by Video Indexer
- extract insights from a live stream of video data

Implement Natural Language Processing Solutions (20-25%)

Analyze text by using the Text Analytics service

- retrieve and process key phrases
- retrieve and process entity information (people, places, urls, etc.)
- retrieve and process sentiment
- detect the language used in text

Manage speech by using the Speech service

- implement text-to-speech
- customize text-to-speech
- implement speech-to-text
- improve speech-to-text accuracy

Translate language

- translate text by using the Translator service
- translate speech-to-speech by using the Speech service
- translate speech-to-text by using the Speech service

Build an initial language model by using Language Understanding Service (LUIS)

- create intents and entities based on a schema, and then add utterances
- create complex hierarchical entities
    - use this instead of roles
- train and deploy a model

Iterate on and optimize a language model by using LUIS

- implement phrase lists
- implement a model as a feature (i.e. prebuilt entities)
- manage punctuation and diacritics
- implement active learning
- monitor and correct data imbalances
- implement patterns

Manage a LUIS model

- manage collaborators
- manage versioning
- publish a model through the portal or in a container
- export a LUIS package
- deploy a LUIS package to a container
- integrate Bot Framework (LUDown) to run outside of the LUIS portal

Implement Knowledge Mining Solutions (15-20%)

Implement a Cognitive Search solution

- create data sources
- define an index
- create and run an indexer
- query an index
- configure an index to support autocomplete and autosuggest
- boost results based on relevance
- implement synonyms

Implement an enrichment pipeline

- attach a Cognitive Services account to a skillset
- select and include built-in skills for documents
- implement custom skills and include them in a skillset

Implement a knowledge store

- define file projections
- define object projections
- define table projections
- query projections

Manage a Cognitive Search solution

- provision Cognitive Search
- configure security for Cognitive Search
- configure scalability for Cognitive Search

Manage indexing

- manage re-indexing
- rebuild indexes
- schedule indexing
- monitor indexing
- implement incremental indexing
- manage concurrency
- push data to an index
- troubleshoot indexing for a pipeline

Implement Conversational AI Solutions (15-20%)

Create a knowledge base by using QnA Maker

- create a QnA Maker service
- create a knowledge base
- import a knowledge base
- train and test a knowledge base
- publish a knowledge base
- create a multi-turn conversation
- add alternate phrasing
- add chit-chat to a knowledge base
- export a knowledge base
- add active learning to a knowledge base
- manage collaborators

Design and implement conversation flow

- design conversation logic for a bot
- create and evaluate *.chat file conversations by using the Bot Framework Emulator
- add language generation for a response
- design and implement adaptive cards

Create a bot by using the Bot Framework SDK

- implement dialogs
- maintain state
- implement logging for a bot conversation
- implement a prompt for user input
- add and review bot telemetry
- implement a bot-to-human handoff
- troubleshoot a conversational bot
- add a custom middleware for processing user messages
- manage identity and authentication
- implement channel-specific logic
- publish a bot

Create a bot by using the Bot Framework Composer

- implement dialogs
- maintain state
- implement logging for a bot conversation
- implement prompts for user input
- troubleshoot a conversational bot
- test a bot by using the Bot Framework Emulator

- publish a bot

Integrate Cognitive Services into a bot

- integrate a QnA Maker service
- integrate a LUIS service
- integrate a Speech service
- integrate Dispatch for multiple language models
- manage keys in app settings file