

# Technical Design Document (TDD) – Proyecto Learning Journey BCP

**Proyecto:** BCP Learning Journey Automation — Fase 1 “envía email ↔ recibe Excel”

**Versión:** 1.0 — 25-abr-2025

**Autor:** Equipo IA & Data Perú – Indra / Minsait

## 1. Resumen Técnico

### 1.1 Objetivo del sistema

Diseñar e implementar una solución serverless en Azure que permita **automatizar la generación de Learning Journeys personalizados** a partir de Mapas de Carrera, mediante procesamiento semántico y búsqueda de cursos en catálogos educativos estructurados.

El sistema debe funcionar con **mínima fricción operacional**, operar de forma **segura, trazable y escalable**, y ser completamente ejecutado **frente a la infraestructura de BCP** durante la Fase 1.

### 1.2 Alcance de esta versión (Fase 1)

- **Entrada:** archivo Excel enviado por correo con el mapa de carrera de un rol o especialidad.
- **Proceso:** validación, embeddings, búsqueda híbrida en base vectorial y catálogo estructurado, selección óptima de recursos.
- **Salida:** archivo Excel con Learning Journey sugerido, devuelto por correo al remitente.
- **Infraestructura:** basada en componentes nativos de Azure, contenedores, y arquitectura desacoplada.
- **Catálogo:** actualizado mensualmente desde Udemy y Microsoft Learn.

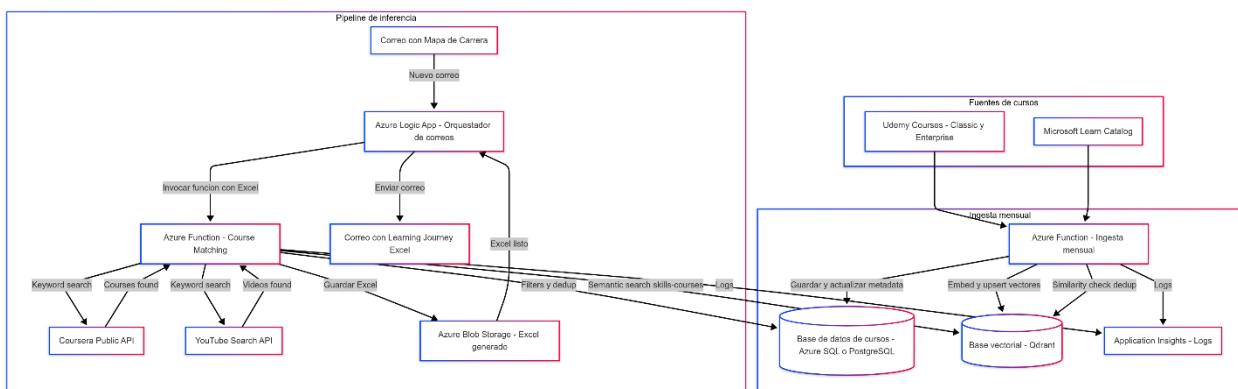
### 1.3 Arquitectura general:

Infraestructura event-driven serverless en Azure, que combina IA semántica (OpenAI), almacenamiento estructurado (PostgreSQL), búsqueda vectorial (Qdrant), y orquestación vía correo electrónico (Logic App), para automatizar rutas de aprendizaje desde mapas de carrera.

## 2. Arquitectura técnica

Esta sección describe los **componentes clave de la solución técnica**, su función específica dentro del flujo, y cómo se comunican entre sí para implementar la Fase 1 completa de automatización del Learning Journey.

## 2.1 Diagrama de arquitectura



## 2.2 Componentes principales

Componente	Tipo	Función principal
Azure Logic App (Email Listener)	Orquestador	Escucha correo con adjunto, lanza procesamiento IA, y envía respuesta al usuario
Azure Function – Ingesta mensual	Serverless	Extrae, normaliza y actualiza catálogos desde Udemy y MS Learn; genera embeddings
Azure Function – Course Matching	Serverless	Procesa mapas de carrera, consulta base vectorial y estructurada, y arma recomendación
Qdrant	Base vectorial (Container App)	Almacena embeddings de cursos y permite búsqueda ANN semántica
PostgreSQL	Base estructurada	Almacena metadata estructurada de los cursos (categorías, idioma, dificultad, rating)
Azure Blob Storage	Almacenamiento	Guarda los archivos Excel de entrada y salida, versionados y trazables
Application Insights	Observabilidad	Registra logs, métricas y errores de todas las ejecuciones
Azure Key Vault	Seguridad	Guarda credenciales, claves API y configuraciones sensibles
YouTube APIs	Servicios externos	Permiten complementar sugerencias si el catálogo interno no tiene suficientes recursos
Azure OpenAI	Modelo IA	Genera embeddings por skill-nivel usando text-embedding-ada-002

### 3. Flujos de datos

Esta sección describe los principales flujos que componen la solución:

- **Flujo de ingesta mensual** del catálogo de cursos.
- **Flujo de inferencia** (de entrada por correo hasta entrega del Learning Journey).
- **Flujo de observabilidad** y trazabilidad.

#### 3.1 Flujo de ingesta mensual

Proceso automatizado que actualiza el catálogo de recursos educativos desde fuentes externas (Udemy y Microsoft Learn), normaliza los datos y genera embeddings para búsqueda semántica.

**Secuencia:**

1. Azure Function se ejecuta por TimerTrigger mensual.
2. Se llaman las APIs de Udemy (Classic & Enterprise) y Microsoft Learn para obtener nuevos cursos.
3. Se normalizan campos: título, descripción, duración, dificultad, idioma, rating.
4. Se almacenan o actualizan los registros en PostgreSQL (upsert por course\_id).
5. Se genera un embedding (title + description) para cada curso usando Azure OpenAI.
6. Se realiza deduplicación por similitud ( $> 0.95$  cosine) en Qdrant, marcando cursos duplicados.
7. Se registran métricas de ingesta y se loguea la corrida en Application Insights y tabla ingest\_audit.

#### 3.2 Flujo de inferencia (correo → recomendación → correo)

Este flujo se activa de forma dinámica cada vez que un especialista envía un mapa de carrera por correo.

**Secuencia:**

1. Logic App escucha una nueva entrada en el buzón designado.
2. El adjunto (Excel) es validado por estructura y nombre.
3. Se invoca Azure Function con el archivo como input.
4. Por cada skill-nivel:
  - Se genera el embedding (text-embedding-ada-002).
  - Se hace búsqueda en Qdrant (top K = 40).
  - Se hace keyword search en PostgreSQL (por título, descripción).
  - Se aplican filtros (idioma, dificultad, estado activo).
  - Se computa un ranking híbrido (semántico, RRF, rating, bonus multi-skill).

- Se aplica algoritmo *greedy set-cover* (máx. 4/2/1 recursos por capacidad).
  - Se consulta YouTube para complementar los cursos.
5. Se arma un archivo Excel estructurado por Role, con hoja oculta Audit\_Log.
  6. El archivo generado se guarda en Azure Blob Storage con convención: learning\_journeys/YYYY-MM-DD/role\_timestamp.xlsx
  7. Logic App responde al remitente con el archivo adjunto y estado del procesamiento.

### 3.3 Flujo de observabilidad

**Logging y monitoreo embebido en cada etapa:**

- Logs estructurados enviados a **Azure Application Insights** desde todas las funciones.
- Métricas registradas:
  - processing\_time\_ms
  - skills\_count
  - avg\_similarity
  - resources\_suggested
  - failures\_detected
- Dashboards (opcionales) pueden montarse sobre Log Analytics o Grafana.
- Cada archivo de entrada/salida se almacena con metadatos en Blob Storage.

## 4. Interfaces y contratos

Esta sección detalla los contratos de datos, APIs y estructuras utilizadas en los componentes clave del sistema, asegurando interoperabilidad entre módulos y claridad para mantenimiento o escalamiento.

### 4.1 Contrato de entrada – Excel (Mapa de Carrera)

**Formato requerido:**

Archivo Excel .xlsx con hoja "CareerMap"

**Columnas obligatorias:**

Campo	Tipo	Descripción
Role	Texto	Nombre del rol (ej. "Project Manager Senior")
Role_Desc	Texto	Descripción del rol (ej. "El rol de Project Manager se encarga de ...")
Group	Texto	Unidad o especialidad (ej. "Project Management")

Group_Desc	Texto	Descripción del grupo (ej. "El grupo Project Management agrupa a todas las capacidades que desarrollen...")
Capability	Texto	Capacidad a desarrollar (ej. "Gestión de Proyectos")
Capability_Desc	Texto	Descripción de la capacidad (ej. "La Gestión de proyectos es la capacidad que desarrolla las habilidades ...")
Capability_Type	Texto	Tipo de capacidad ("Core", "Relevante" o "Normal")
Skill	Texto	Habilidad específica (ej. "Planificación de proyectos")
Skill_Desc	Texto	Descripción de la habilidad específica (ej. "La planificación de proyectos es la habilidad capaz de ...")
Level	Texto	Nivel esperado (Novato, PA, Competente, Proficiente, Master)
Level_Desc	Texto	Descripción del comportamiento esperado en ese nivel

#### Validaciones:

- Todas las columnas deben estar presentes.
- Valores no pueden estar vacíos.
- Solo se aceptan niveles del set predefinido de BCP.
- Se permite múltiples Roles en el mismo archivo.

#### Consideraciones:

- Un rol puede contener más de un grupo.
- Un grupo puede contener más de una capacidad.
- Una capacidad puede contener más de una skill.
- Una capacidad siempre deberá contener los 5 niveles con su descripción correspondiente.

## 4.2 Contrato de salida – Excel (Learning Journey)

#### Formato generado:

Archivo Excel .xlsx por rol, con estructura tabular + hoja oculta de auditoría.

#### Columnas visibles:

Campo	Descripción
Capability	Capacidad original
Skill	Skill del mapa
Level	Nivel esperado
Course_Title	Nombre del curso sugerido
Platform	Plataforma origen (Udemy, MS Learn, YouTube)

URL	Link directo al recurso
Language	Idioma del curso
Difficulty	Dificultad (Beginner, Intermediate, Advanced)
Rating	Score de usuario (si aplica)
Similarity	Similitud semántica (0–1)

### Hoja oculta Audit\_Log:

- Course\_ID, Source\_Type, Embedding\_Score, Ranking\_Position, Duplicate\_Of, Selected\_by

## 4.3 API externa – Azure OpenAI

Recurso	Valor
Endpoint	<a href="https://&lt;region&gt;.openai.azure.com/openai/deployments/&lt;deployment_id&gt;/embeddings">https://&lt;region&gt;.openai.azure.com/openai/deployments/&lt;deployment_id&gt;/embeddings</a>
Modelo	text-embedding-ada-002
Input	Skill + nivel (concatenado)
Output	Embedding vector (1536 dimensiones aprox.)
Seguridad	API Key almacenada en Key Vault

## 4.4 APIs complementarias (YouTube)

### YouTube API (v3):

- Endpoint: <https://www.googleapis.com/youtube/v3/search?part=snippet&q={keyword}>
- Filtros: duración, fecha, canal

## 5. Lógica de negocio

Esta sección describe el conjunto de algoritmos, reglas y transformaciones que el sistema aplica para convertir un mapa de carrera en un Learning Journey optimizado y relevante.

### 5.1 Generación de embeddings (por skill-nivel)

- Para cada fila del Excel (Skill + Level\_Desc), se concatena el texto y se genera un vector semántico utilizando **Azure OpenAI** (text-embedding-ada-002).

- Estos vectores representan el significado del skill requerido y permiten compararlos con cursos existentes.
- Los embeddings se mantienen **transient** (no se almacenan); se usan inmediatamente para hacer consulta a Qdrant.

## 5.2 Búsqueda híbrida

El motor de recomendación hace **una búsqueda híbrida en dos fuentes**:

Fuente	Técnica	Detalle
<b>Qdrant</b>	Búsqueda vectorial (ANN)	Devuelve $k = 40$ cursos más cercanos al embedding
<b>PostgreSQL</b>	Keyword search	Búsqueda textual por title, description, tags con relevancia ponderada (RRF)

Los resultados de ambas fuentes son combinados y deduplicados antes de aplicar ranking.

## 5.3 Fórmula de ranking

Cada curso es puntuado con una fórmula ponderada:

score =

$$\begin{aligned} & (0.60 * \text{cosine\_similarity}) + \\ & (0.15 * \text{RRF\_keyword\_boost}) + \\ & (0.15 * \text{normalized\_rating}) + \\ & (0.10 * \text{multi\_skill\_bonus}) \end{aligned}$$

Donde:

- **cosine\_similarity**: cercanía entre el embedding del skill y el del curso.
- **RRF\_keyword\_boost**: factor que premia si el título o descripción coincide con keywords clave.
- **normalized\_rating**: si el curso tiene calificación (1–5), se normaliza a [0–1].
- **multi\_skill\_bonus**: si un curso cubre más de un skill del mapa, recibe un extra.

Los cursos se ordenan de mayor a menor score final.

## 5.4 Deduplicación

- Durante la ingestión, los cursos que tienen alta similitud entre sí ( $\text{cosine} > 0.95$ ) se marcan como duplicados (`duplicate_of`).

- Durante la inferencia, **solo uno de cada grupo de duplicados** es elegible como resultado.

## 5.5 Selección por capacidad (greedy set-cover)

Cada Capability del mapa tiene múltiples skills. El objetivo es recomendar una **malla balanceada y no redundante**.

Se aplica un algoritmo tipo *greedy set-cover*:

- Por cada Capability, se buscan cursos que cubran múltiples skills.
- Se seleccionan **hasta**:
  - 4 cursos para capacidades “core”
  - 2 cursos para capacidades “relevantes”
  - 1 curso para capacidades “normales”
- Si no se alcanza el mínimo deseado, se ajustarán los parámetros de los algoritmos para mejor eficiencia.

## 5.6 Fallback y búsqueda externa

Si luego de aplicar filtros y ranking, **no se encuentra al menos un curso aceptable** (score > 0.7 o link roto), el sistema:

- Lanza una búsqueda en vivo en **YouTube API** usando el Skill como keyword.
- Se limitan los resultados a los top 2 por fuente, priorizando duración, canal reconocido y disponibilidad pública.
- Estos cursos externos se marcan como Source\_Type = external en la hoja de auditoría.

# 6. Seguridad y control de acceso

Esta sección documenta los mecanismos de autenticación, autorización, protección de datos y cumplimiento utilizados para mantener la solución segura tanto en tránsito como en reposo, así como las prácticas de minimización de riesgo operativo.

## 6.1 Identidades administradas (MSI)

- Todas las Azure Functions y Logic Apps se ejecutan con **Managed Identity (MSI)**.
- Los accesos a servicios como PostgreSQL, Key Vault y Blob Storage se restringen mediante **RBAC**.
- Se evita el uso de credenciales explícitas o hardcoded en el código fuente.

## 6.2 Key Vault y gestión de secretos

- **Azure Key Vault** almacena de forma segura:
  - API Keys de OpenAI y YouTube
  - Connection strings a bases de datos

- Configuraciones sensibles de parámetros (e.g. top K, pesos de ranking)
- El acceso al Vault se controla con políticas específicas por identidad.
- Todos los secretos tienen ciclo de vida y rotación definida.

### 6.3 Encriptación y protección de datos

Tipo de dato	Protección aplicada
Archivos Excel (input/output)	Almacenados en Azure Blob Storage con <b>encriptación en reposo</b> (AES-256)
Comunicación entre componentes	HTTPS/TLS 1.2+ en todas las conexiones
Datos de usuarios (nombres de roles, skills)	No contienen PII; se consideran datos organizacionales y se manejan con estándar interno de confidencialidad

### 6.4 Restricción de red y aislamiento

- El sistema opera completamente en un entorno **externo a la red BCP** durante la Fase 1.
- Los servicios están desplegados sobre recursos Azure de Indra con firewalls, NSGs y limitación de IPs públicas.
- En fases futuras, la integración con Apps Killling considerará canales seguros (ej. VNET Integration o Private Endpoints).

### 6.5 Trazabilidad y auditoría

- Cada procesamiento (tanto de ingesta como de inferencia) registra:
  - run\_id, timestamps, usuario origen (correo), input hash
  - ID de modelo OpenAI usado, versión de parámetros
  - Hash del archivo Excel generado
- Todos los logs son enviados a **Application Insights** y pueden ser auditados por fecha, usuario o recurso generado.

### 6.6 Principio de mínimo privilegio

- Cada componente tiene acceso **solo a los recursos que necesita**.
- Las APIs externas se acceden con scopes limitados.
- Los archivos y datos tienen **tiempo de vida** limitado:
  - Outputs en Blob exirian a los 90 días (configurable).
  - Logs de procesamiento con retención de 6 meses.

## 7. Observabilidad y métricas

Esta sección describe cómo se recolectan, estructuran y exponen los eventos, errores, tiempos de ejecución y resultados del sistema. También establece las métricas clave (KPIs) que serán monitoreadas para evaluar salud operativa y efectividad del modelo IA.

### 7.1 Logging estructurado

Todas las Azure Functions y Logic Apps están instrumentadas con **logs estructurados** enviados a **Azure Application Insights**.

Cada ejecución registra:

Campo	Descripción
run_id	UUID único por ejecución (ingesta o inferencia)
timestamp	Fecha y hora de inicio y fin
input_hash	Hash SHA del archivo Excel recibido
skills_processed	Número de skill-niveles procesados
processing_time_ms	Tiempo total de ejecución
avg_similarity_score	Promedio de similitud de los cursos seleccionados
courses_returned	Total de cursos propuestos en el output
fallback_triggered	Si se activó búsqueda externa (sí/no)
status	Éxito / fallo / validación fallida

### 7.2 KPIs técnicos clave

Indicador	Objetivo	Fuente
Tiempo promedio de procesamiento por archivo	≤ 5 minutos	App Insights (processing_time_ms)
Tasa de errores	< 2 % por volumen total	App Insights + Alertas
% de links activos en cursos sugeridos	≥ 90 %	Checker interno (puede ejecutarse como validación posterior)
# de ejecuciones exitosas	≥ 30 archivos procesados sin error	App Insights + Blob audit

% de sugerencias con score > 0.75	Indicador de calidad semántica	Hoja Audit_Log + logs agregados
-----------------------------------	--------------------------------	---------------------------------

### 7.3 Alertas y monitoreo

Se configuran **alertas automáticas** sobre Application Insights / Log Analytics para:

- Fallos en ejecución de Functions o Logic App (status != success)
- Tiempo de procesamiento > 5 minutos
- Respuesta nula de Qdrant u OpenAI
- Fallbacks frecuentes en +30 % de skills
- Uso de tokens OpenAI > umbral mensual

Estas alertas pueden enviarse por correo o Teams según configuración deseada.

### 7.4 Auditoría de archivos y trazabilidad

Cada archivo Excel procesado se registra:

- En **Azure Blob Storage** con metadata: role, fecha, versión, run\_id, usuario origen.
- Naming convention:  
learning\_journeys/2025-04-25/PMO\_Junior\_14h33\_runABC123.xlsx

Opcionalmente, se puede habilitar un **panel de trazabilidad** sobre Log Analytics o Power BI en fases futuras.

## 8. CI/CD y despliegue

Esta sección define el enfoque de infraestructura como código, versionado y automatización de despliegues para garantizar que el sistema sea reproducible, auditável y fácil de evolucionar.

### 8.1 Infraestructura como código (IaC)

- Toda la infraestructura se define mediante **plantillas Bicep** (o ARM templates si es requerido).
- Repositorios separados para:
  - infra/ → Logic Apps, Functions, Blob, SQL, VDB, Key Vault
  - code/ → Código fuente de Functions, scripts, configuración
- Parámetros expuestos:
  - Nombre de recursos
  - Ubicación (ej. East US o South Central US)
  - SKU / plan de consumo

- Permisos RBAC por componente

### Ejemplo de recursos aprovisionados:

Recurso	Tipo	Observaciones
Logic App	Event-driven desde O365 inbox	Permisos de lectura sobre mailbox específico
Azure Functions	Python 3.11, Consumption Plan	Slots habilitados para blue/green deploy
PostgreSQL Flexible Server	Plan Basic	Conexión vía private link opcional
Azure Blob Storage	Standard Hot	Versionado y logs habilitados
Qdrant Container	Azure Container Apps	Imagen pública o custom con endpoint REST
Azure Key Vault	Standard	Referenciado por MSI

## 8.2 Pipelines de despliegue

- **GitHub Actions** (o Azure DevOps) como sistema de CI/CD principal.
- Triggers:
  - push a rama main → despliegue a entorno de staging
  - tag con vX.Y.Z → despliegue a producción
- Pasos del pipeline:
  - Validación sintáctica de Bicep / código Python
  - Linting + tests unitarios
  - Deploy infra (si no existe)
  - Deploy de código fuente a Azure Function
  - Slot swap en staging (blue/green)
  - Validación post-deploy (ping healthcheck, test dummy input)
  - Notificación en canal Teams / Email

## 8.3 Versionado

Elemento	Método de versionado
Código fuente	Git semántico (v1.0.0)
Modelos IA / prompts	Referencia explícita en cada ejecución (deployment_id, version)
Configuración	Archivo config.yml con pesos de ranking, top K, filtros
Archivos generados	Metadata en Blob: fecha, rol, run_id, versión de lógica aplicada

## 8.4 Buenas prácticas y seguridad en CI/CD

- Todos los secretos se acceden vía **Key Vault references** (no variables en pipelines).
- Revisión de permisos mínima requerida en tokens de despliegue.
- Restricción de ambientes (staging, prod) con políticas de aprobación manual.
- Deploy automatizado, rollback manual con botón desde el portal si necesario.

## 9. Riesgos técnicos y mitigaciones

Esta sección documenta los riesgos más relevantes de la Fase 1 desde el punto de vista técnico y operativo, así como las medidas preventivas y correctivas consideradas en el diseño de la solución.

### 9.1 Riesgos de procesamiento

Riesgo	Descripción	Mitigación
<b>Archivo malformado</b>	Archivos Excel que no cumplen el formato esperado (columnas faltantes, celdas vacías)	Validación estricta del esquema. Si falla, se envía correo de error explicativo al remitente y se registra el evento.
<b>Falla en generación de embeddings</b>	Timeout o error en la API de Azure OpenAI	Retries automáticos con backoff exponencial. Si persiste, fallback a keyword search sin semántica. Logs marcados como “degradado”.
<b>Archivo demasiado grande</b>	Mapas con más de 500 skills podrían exceder límites de tiempo o memoria	Límite explícito por archivo. Se rechaza y se sugiere dividir en múltiples archivos. Se parametriza máximo de filas.

## 9.2 Riesgos de integraciones externas

Riesgo	Descripción	Mitigación
<b>APIs de YouTube no disponible</b>	Falla en búsqueda complementaria externa	Si no se logra obtener cursos externos, se notifica en la hoja de auditoría. El sistema no falla, solo marca que no se pudo completar sugerencias externas.
<b>Token de API vencido o mal configurado</b>	Fallo en autenticación a OpenAlo Key Vault	Alertas configuradas si llamadas fallan. Uso de Key Vault para rotación segura y logs específicos para errores de autenticación.
<b>Fallo en Qdrant</b>	Contenedor no responde o está saturado	Healthchecks periódicos + reinicio automático del container. Se puede fallback a PostgreSQL-only como emergencia temporal.

## 9.3 Riesgos de calidad y precisión

Riesgo	Descripción	Mitigación
<b>Recomendaciones irrelevantes</b>	Cursos sugeridos no alineados con el nivel esperado o mal ranqueados	Umbral de score mínimo ( $> 0.75$ ) para aceptar curso. Promedio de similitud monitoreado. Validación manual (Human-in-the-loop) durante Fase 1.
<b>Cursos duplicados</b>	Recursos redundantes dentro de una misma capacidad o entre capacidades	Deduplicación por embedding en la ingesta + filtrado durante inferencia. Registro de <code>duplicate_of</code> en base de datos.
<b>Links caídos o recursos no disponibles</b>	Cursos ya no accesibles en la plataforma origen	Checker HTTP opcional + hoja de auditoría que marca recursos no válidos. Prioridad de catálogo interno con refresco mensual.

## 9.4 Riesgos operativos

Riesgo	Descripción	Mitigación
<b>Fallos en Logic App (correo)</b>	No se dispara procesamiento por errores en el flujo o problemas de conectividad con Outlook	Retry automático + alerta. Fallback manual para reenviar correo si es necesario. Se puede habilitar endpoint alternativo para testeo.

<b>Uso excesivo de recursos o costos</b>	Embeddings masivos o tráfico elevado generan costos mayores a lo estimado	Embeddings en batch, top-K limitado, conteo de tokens. Monitoreo de uso mensual de OpenAI y storage. Límites configurables.
<b>Accesos indebidos a archivos o secretos</b>	Exposición de datos o configuraciones sensibles	MSI + Key Vault + RBAC. Logs de acceso auditables. Retención de archivos limitada en Blob (90 días).

## 10. Pruebas técnicas

Esta sección detalla los tipos de pruebas que se realizarán para asegurar que el sistema cumpla con los requisitos funcionales y no funcionales de la Fase 1, y que se comporte de forma robusta ante diferentes escenarios de uso y error.

### 10.1 Pruebas unitarias

- Aplican principalmente sobre la lógica dentro de Azure Functions:
  - Validación del esquema del Excel de entrada
  - Generación de embeddings
  - Ranking híbrido con fórmula completa
  - Detección de duplicados
  - Set-cover para selección óptima de cursos

**Cobertura esperada:** ≥ 80 % de funciones críticas del backend

### 10.2 Pruebas de integración

- Validación end-to-end del flujo completo:
  - Envío de correo con Excel → generación del Learning Journey → respuesta por correo
- Verificación de:
  - Interacción con Logic App y Function
  - Acceso correcto a PostgreSQL y Qdrant
  - Comunicación exitosa con APIs externas (OpenAI y YouTube)
  - Escritura correcta del archivo Excel y carga en Blob Storage
  - Logging estructurado en Application Insights

#### Casos usados:

✓ Mapa de carrera con 5 capacidades y 15 skills

- ✓ Mapa con errores intencionales (nivel inválido, columnas faltantes)
- ✓ Mapa grande con 300+ skills

### 10.3 Validación funcional (UAT)

- Especialistas de Talento y líderes técnicos del BCP validan la **relevancia de los resultados generados**
- Criterios de aceptación:
  - ≥ 85 % de cursos sugeridos considerados adecuados sin intervención
  - Cobertura completa de todas las capacidades solicitadas
  - Links funcionales y formatos compatibles con carga en Appskilling

### 10.4 Pruebas de carga (Load testing)

- Simulación de procesamiento de múltiples archivos en paralelo con alto volumen de skills
- Herramienta recomendada: k6 o Locust para medir:
  - Latencia por skill
  - Escalabilidad de Qdrant y PostgreSQL
  - Uso de CPU y memoria en Functions
- Objetivo: mantener tiempo promedio de procesamiento por archivo ≤ 5 min

### 10.5 Pruebas de fallos y resiliencia

- Casos extremos y fallas intencionadas:
  - API OpenAI caído o sin respuesta
  - Archivo corrupto o en formato incorrecto
  - Token inválido en Key Vault
  - Curso duplicado en catálogo
- Expectativa: el sistema responde con error controlado, loguea, notifica al usuario si aplica, y no cae en cascada

## 11. Checklist de producción

Esta sección actúa como lista de verificación operativa para validar que todos los aspectos críticos del sistema están correctamente implementados, configurados y probados antes del paso a producción.

### 11.1 Validación funcional completa

- Mapa de carrera correctamente validado desde Excel real
- Learning Journey generado y estructurado con formato aprobado

- Cursos sugeridos cubren todas las capacidades con lógica set-cover
- Links funcionales y activos
- Especialistas han validado  $\geq 85\%$  de aceptación sin edición

## 11.2 Validación técnica

- Todas las Functions desplegadas y funcionando en entorno productivo
- Logic App escucha correctamente el correo corporativo configurado
- Ingesta mensual de cursos completada y visible en base de datos
- Qdrant responde a consultas vectoriales en < 300 ms promedio
- PostgreSQL contiene los cursos estructurados con campos normalizados

## 11.3 Seguridad y control de acceso

- MSI habilitada en todos los recursos
- Accesos RBAC auditados y mínimos para cada componente
- Key Vault funcionando y sin secretos hardcoded en código
- HTTPS obligatorio en todos los endpoints
- Logs de acceso y errores visibles en Application Insights

## 11.4 Observabilidad y monitoreo

- App Insights recibe logs desde todas las Functions
- Dashboards de telemetría muestran:
  - Procesamientos recientes
  - Tiempo medio por archivo
  - Porcentaje de errores
- Alertas configuradas para:
  - Fallas en procesamiento
  - Uso elevado de recursos
  - Fallbacks frecuentes

## 11.5 CI/CD y trazabilidad

- Pipeline de despliegue funcional para infraestructura y código
- Deploy a entorno staging probado y validado
- Deploy a production con slot swap verificado

- Versionado del código, modelo, configuración y archivos generado correctamente
- Archivos almacenados en Blob con metadata y convención de nombres

## 11.6 Revisión final

- Documentación técnica entregada (TDD, PRD, configuración)
- Soporte técnico asignado para monitoreo post-lanzamiento
- Revisión de cumplimiento por equipo de Seguridad / Arquitectura BCP
- Plan de contingencia definido para rollback o intervención manual