Cobertura de Código y Pruebas Unitarias - DataGenie

Casos de Prueba por Componente

Número del Caso de Prueba	Componente	Descripción de lo que se Probará	Prerrequisitos
<<CA001>>	Connection Manager	Validar conexión exitosa a base de datos MySQL	- MySQL server activo - Credenciales válidas - Red accesible
<<CA002>>	Query Processor	Validar procesamiento de consultas en lenguaje natural	- Conexión a BD establecida - Esquema de BD disponible
<<CA003>>	LLM Service	Validar integración con Groq API y generación SQL	- API Key de Groq válida - Conexión a internet - Modelo Llama3 disponible
<<CA004>>	Result Formatter	Validar formateo y visualización de resultados	- Datos de prueba disponibles - Consulta SQL ejecutada
<<CA005>>	Streamlit Frontend	Validar interfaz de usuario y session state	- Aplicación Streamlit iniciada - Componentes UI cargados

---

<<CA001>> - Connection Manager

Nº	Descripción	Método	Datos Entrada	Salida Esperada	¿OK?	Obser
1	Conexión exitosa con credenciales válidas	`test_valid_connection()`	host='localhost', user='root', password='123', database='test_db'	Conexión establecida, status=True		
2	Fallo de conexión con credenciales inválidas	`test_invalid_credentials()`	host='localhost', user='wrong', password='wrong', database='test_db'	Error de autenticación, status=False		
3	Validación de pool de conexiones	`test_connection_pool()`	max_connections=5, concurrent_requests=10	Pool gestiona conexiones correctamente		
4	Timeout de conexión	`test_connection_timeout()`	host='192.168.1.999', timeout=5	Timeout exception después de 5s		
5	Cierre seguro de conexiones	`test_connection_cleanup()`	active_connections=3	Todas las conexiones cerradas		
6	Validación de esquema de BD	`test_schema_validation()`	database='test_db'	Schema extraído correctamente		

<<CA002>> - Query Processor

Nº	Descripción	Método	Datos Entrada	Salida Esperada	¿OK?	Observaciones
1	Procesamiento de consulta simple	`test_simple_query()`	"¿Cuántos usuarios hay?"	Consulta procesada, tipo='SELECT'		
2	Validación de consulta compleja	`test_complex_query()`	"Muestra usuarios con más de 25 años ordenados por edad"	Consulta con JOIN/WHERE válida		
3	Rechazo de consultas maliciosas	`test_sql_injection()`	"'; DROP TABLE users; --"	Consulta rechazada, error de seguridad		
4	Manejo de consultas ambiguas	`test_ambiguous_query()`	"Dame información"	Solicitud de clarificación		
5	Validación de sintaxis SQL generada	`test_sql_syntax()`	"Usuarios activos del último mes"	SQL sintácticamente correcta		
6	Procesamiento de múltiples tablas	`test_multi_table_query()`	"Pedidos con información de clientes"	JOIN correctamente formado		

<<CA003>> - LLM Service

Nº	Descripción	Método	Datos Entrada	Salida Esperada	¿OK?	Obse
1	Integración exitosa con Groq API	`test_groq_api_connection()`	api_key='valid_key', model='llama3-8b-8192'	Response status=200, conexión OK		
2	Generación de SQL desde lenguaje natural	`test_nl_to_sql()`	"Muestra todos los productos"	"SELECT * FROM productos;"		
3	Manejo de errores de API	`test_api_error_handling()`	api_key='invalid_key'	Error capturado, mensaje informativo		
4	Contexto de conversación	`test_conversation_context()`	["¿Cuántos usuarios?", "¿Y cuántos activos?"]	Contexto mantenido entre consultas		
5	Límite de tokens	`test_token_limit()`	consulta_muy_larga (>8192 tokens)	Consulta truncada o error controlado		
6	Validación de respuesta LLM	`test_llm_response_validation()`	respuesta_groq	SQL válida extraída correctamente		

<<CA004>> - Result Formatter

Nº	Descripción	Método	Datos Entrada	Salida Esperada	¿OK?	Observaci
1	Formateo de resultados tabulares	`test_table_formatting()`	[(1,'Juan',25), (2,'Ana',30)]	DataFrame pandas bien formateado		
2	Manejo de resultados vacíos	`test_empty_results()`	[]	Mensaje "No se encontraron resultados"		
3	Formateo de tipos de datos	`test_data_type_formatting()`	datos_mixtos (int, str, date, float)	Tipos correctamente formateados		
4	Límite de filas mostradas	`test_row_limit()`	10000 filas	Paginación o límite aplicado		
5	Exportación de resultados	`test_export_functionality()`	datos_formateados	CSV/Excel generado correctamente		
6	Visualización de gráficos	`test_chart_generation()`	datos_numericos	Gráfico Streamlit generado		

<<CA005>> - Streamlit Frontend

Nº	Descripción	Método	Datos Entrada	Salida Esperada	¿OK?	C
1	Carga de componentes UI	`test_ui_components()`	página_principal	Sidebar, input, botones cargados		
2	Gestión de session state	`test_session_state()`	st.session_state['historial']	Estado persistente entre interacciones		
3	Validación de formularios	`test_form_validation()`	campos_conexion_vacios	Mensajes de error mostrados		
4	Interactividad de botones	`test_button_interactions()`	click_conectar, click_consultar	Acciones ejecutadas correctamente		
5	Responsive design	`test_responsive_layout()`	diferentes_tamaños_pantalla	Layout se adapta correctamente		
6	Manejo de errores en UI	`test_ui_error_handling()`	error_conexion_bd	Error mostrado amigablemente		

Métricas de Cobertura Objetivo

- **Cobertura de Líneas**: ≥ 85%
- **Cobertura de Funciones**: ≥ 90%
- **Cobertura de Ramas**: ≥ 80%
- **Casos de Prueba Exitosos**: ≥ 95%

Herramientas de Testing

- **Framework**: pytest
- **Cobertura**: pytest-cov
- **Mocking**: pytest-mock, unittest.mock
- **Testing de Streamlit**: streamlit-testing
- **Base de Datos**: pytest-mysql (para BD de pruebas)

Comandos de Ejecución

```bash
# Ejecutar todas las pruebas
pytest tests/ -v

# Ejecutar con cobertura
pytest tests/ --cov=src --cov-report=html

# Ejecutar pruebas específicas
pytest tests/test_connection_manager.py -v
```