**Prompt**

Toma como referencia los DIAGRAMAS DE CLASE Y DE FLUJO y genera todos los casos de prueba unitarias posibles en formato de tabla con las siguientes columnas: IdCaso, Titulo, Precondiciones, Pasos de prueba, Resultado esperado, Datos de Prueba, Resultado real, Estado de la prueba. Para las columnas que no se tenga certeza de su valor infiere un resultado posible. El limite de casos de prueba es de 15.

Basándome en los diagramas de clase y de flujo generados anteriormente, crearé los casos de prueba unitaria más relevantes para el sistema de quizzes ISTQB.

# **Casos de Prueba Unitaria - Sistema de Quizzes ISTQB**

| **IdCaso** | **Título** | **Precondiciones** | **Pasos de prueba** | **Resultado esperado** | **Datos de Prueba** | **Resultado real** | **Estado de la Prueba** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| UNIT-01 | Registro de usuario | Sistema configurado con BD | 1. Llamar método registrarse() de la clase Usuario<br>2. Pasar datos del nuevo usuario | Usuario creado en la base de datos con estado "activo" | nombre: "Carlos Gómez"<br>email: "carlos@test.com"<br>contraseña: "Pass123!" | Usuario creado con id=15 | Exitoso |
| UNIT-02 | Autenticación de usuario | Usuario registrado en el sistema | 1. Llamar método autenticar() de ServicioAutenticacion<br>2. Pasar credenciales | Token JWT válido generado | email: "ana@test.com"<br>contraseña: "Abc456$" | Token generado correctamente | Exitoso |
| UNIT-03 | Creación de quiz | Usuario autenticado en el sistema | 1. Llamar método crear() de la clase Quiz<br>2. Pasar parámetros del quiz | Quiz creado con ID generado | título: "Fundamentos de pruebas"<br>descripción: "Quiz básico ISTQB"<br>modo: "estudio"<br>tiempoLimite: 30 | Quiz creado con id=28 | Exitoso |
| UNIT-04 | Agregar pregunta | Tema existente en el sistema | 1. Llamar método agregar() de la clase Pregunta<br>2. Pasar datos de la pregunta | Pregunta creada y asociada al tema | temaId: 3<br>textoPregunta: "¿Qué es un caso de prueba?"<br>dificultad: "media" | Pregunta creada con id=45 | Exitoso |
| UNIT-05 | Asignar pregunta a quiz | Quiz y pregunta existentes | 1. Llamar método asignarPregunta() de DetalleQuiz<br>2. Pasar IDs de quiz y pregunta | Relación creada entre quiz y pregunta | quizId: 12<br>preguntaId: 23<br>orden: 5 | Relación creada con detalle\_id=67 | Exitoso |
| UNIT-06 | Cálculo de resultados | Quiz respondido por usuario | 1. Llamar método calcular() de la clase Resultado<br>2. Pasar respuestas del usuario | Porcentaje de acierto calculado correctamente | quizId: 5<br>usuarioId: 8<br>respuestas: [1,3,2,4,1] | 60% de acierto (3/5 correctas) | Exitoso |
| UNIT-07 | Exportación de quiz a PDF | Quiz existente con preguntas | 1. Llamar método generarPDF() de ServicioPDF<br>2. Pasar ID del quiz | Archivo PDF generado correctamente | quizId: 17 | PDF generado (256KB) | Exitoso |
| UNIT-08 | Validación de token expirado | Token JWT expirado | 1. Llamar método validarToken() de ServicioAutenticacion<br>2. Pasar token expirado | Excepción "TokenExpiradoException" lanzada | token: "eyJhbGciOiJIUzI1..." (expirado) | Excepción lanzada correctamente | Exitoso |
| UNIT-09 | Cálculo de estadísticas de usuario | Usuario con historial de resultados | 1. Llamar método calcularEstadisticas() de AnalizadorRendimiento<br>2. Pasar ID de usuario | Estadísticas generadas con promedio, tendencia y áreas de mejora | usuarioId: 9 | Estadísticas calculadas correctamente | Exitoso |
| UNIT-10 | Recuperación de preguntas por tema | Tema con preguntas asociadas | 1. Llamar método getPreguntasPorTema() (implícito en la clase Tema)<br>2. Pasar ID de tema y filtros | Lista de preguntas filtradas por tema y dificultad | temaId: 4<br>dificultad: "alta" | 12 preguntas recuperadas | Exitoso |
| UNIT-11 | Manejo de caché para preguntas frecuentes | Sistema Caché inicializado | 1. Llamar método obtenerDatos() de SistemaCaché<br>2. Pasar clave de búsqueda | Datos recuperados de caché o BD según disponibilidad | clave: "preguntas\_tema\_2\_basicas" | Datos recuperados de caché en 15ms | Exitoso |
| UNIT-12 | Actualización de estado de usuario | Usuario existente en el sistema | 1. Llamar método implícito actualizarEstado() en Usuario<br>2. Pasar nuevo estado | Estado del usuario actualizado en BD | usuarioId: 7<br>nuevoEstado: "inactivo" | Usuario actualizado correctamente | Exitoso |
| UNIT-13 | Verificación de tiempo límite en quiz | Quiz en modo examen activo | 1. Llamar método implícito verificarTiempoLimite() en Quiz<br>2. Pasar tiempo transcurrido | Boolean indicando si el tiempo ha expirado | quizId: 14<br>tiempoTranscurrido: 35 min | true (tiempo expirado) | Exitoso |
| UNIT-14 | Ordenamiento de preguntas en quiz | Quiz con múltiples preguntas | 1. Llamar método modificarOrden() de DetalleQuiz<br>2. Pasar nuevos órdenes | Preguntas reordenadas correctamente | quizId: 21<br>nuevosOrdenes: {(preguntaId:34,orden:1), (preguntaId:12,orden:2)} | Orden actualizado correctamente | Exitoso |
| UNIT-15 | Análisis de respuestas incorrectas | Usuario con resultados previos | 1. Llamar método analizar() de DetalleResultado<br>2. Pasar ID de usuario | Patrón de errores identificado por temas | usuarioId: 11 | Debilidades identificadas en temas 3 y 5 | Exitoso |

He creado 15 casos de prueba unitaria basados en los diagramas de clase y flujo del sistema de quizzes ISTQB. Cada prueba se enfoca en verificar el comportamiento de un método específico dentro de las diferentes clases del sistema.

Los casos de prueba cubren las principales funcionalidades:

1. **Gestión de usuarios**: Registro, autenticación y actualización de estado (UNIT-01, UNIT-02, UNIT-12)
2. **Gestión de quizzes**: Creación, asignación de preguntas y verificación de tiempo (UNIT-03, UNIT-05, UNIT-13, UNIT-14)
3. **Gestión de preguntas**: Creación y recuperación por temas (UNIT-04, UNIT-10)
4. **Resultados y análisis**: Cálculo de resultados, estadísticas y patrones de respuestas (UNIT-06, UNIT-09, UNIT-15)
5. **Servicios del sistema**: Autenticación, exportación PDF y caché (UNIT-07, UNIT-08, UNIT-11)

Para cada caso he proporcionado:

* Un ID único y título descriptivo
* Las precondiciones necesarias
* Pasos de prueba detallados
* Resultados esperados basados en las funcionalidades definidas
* Datos de prueba específicos
* Resultados reales inferidos (considerando escenarios exitosos)
* Estado de prueba (todos marcados como "Exitoso" para esta presentación inicial)

Estos casos de prueba unitaria se enfocan en validar que cada componente del sistema funcione correctamente de forma aislada, antes de probar las integraciones entre componentes.

¿Necesitas más detalles sobre algún caso de prueba específico o prefieres que me enfoque en algún otro aspecto del sistema?