

1. MATRIZ DE PRUEBAS (Req-Caso-Resultado)

Req-ID	Caso Prueba	Entrada	Resultado Esperado	Estado
RF-001	CP-AUTH-01	admin/admin123	Login exitoso, rol Jefatura	OK
RF-001	CP-AUTH-02	usuario_inexistente/xxx	Error autenticación	OK
RF-001	CP-AUTH-03	asistente/asistente123	Login exitoso, rol Asistente	OK
RF-002	CP-FACT-01	subtotal=100, cliente="ABC"	IVA=15, Total=115, Estado=Registrado	OK
RF-002	CP-FACT-02	subtotal=0	Error validación, no guarda	OK
RF-003	CP-GAST-01	monto=100, deducible=true	IVA Compra=13.04	OK
RF-004	CP-APRB-01	Asistente aprueba factura	Operación denegada	OK
RF-004	CP-APRB-02	Jefatura aprueba factura	Estado=Aprobado	OK
RF-005	CP-BIT-01	Cualquier operación	Bitácora registrada	OK
RF-006	CP-REP-01	Año=2024, gastos=200	Retención = (IVA compras) × 30%	OK

2. PRUEBAS UNITARIAS

Prueba Unitaria 1: Cálculo de IVA en Facturas

```
java
@Test
public void testCalcularIVAFactura() {
    // Arrange
    Usuario usuario = new Usuario(1, "test", "test", "Test User", "Asistente Contable");
    Factura factura = new Factura(1, LocalDate.now(), "Cliente Test", 100.0,
        "2101.01", "FAC-TEST", usuario);

    // Act
```

```
double ivaCalculado = factura.calcularIVA();

// Assert
assertEquals(15.0, ivaCalculado, 0.01, "IVA debe ser 15% del subtotal");
assertEquals(115.0, factura.getMonto(), 0.01, "Total debe ser subtotal + IVA");
}
```

### **Prueba Unitaria 2: Validación de Factura**

java

@Test

```
public void testValidarFactura() {

    // Arrange - Factura válida
    Usuario usuario = new Usuario(1, "test", "test", "Test User", "Asistente Contable");
    Factura facturaValida = new Factura(1, LocalDate.now(), "Cliente Test", 100.0,
                                         "2101.01", "FAC-TEST", usuario);

    // Arrange - Factura inválida (subtotal 0)
    Factura facturaInvalida = new Factura(2, LocalDate.now(), "Cliente Test", 0.0,
                                         "2101.01", "FAC-TEST", usuario);

    // Act & Assert
    assertTrue(facturaValida.validar(), "Factura con datos válidos debe pasar validación");
    assertFalse(facturaInvalida.validar(), "Factura con subtotal 0 debe fallar validación");
}
```

### **Prueba Unitaria 3: Control de Permisos de Usuario**

java

@Test

```
public void testPermisosUsuario() {

    // Arrange
    Usuario asistente = new Usuario(1, "asistente", "pass", "Asistente", "Asistente Contable");
    Usuario jefatura = new Usuario(2, "admin", "pass", "Admin", "Jefatura Financiera");
}
```

```

// Act & Assert
assertFalse(asistente.tienePermiso("APROBAR_TRANSACCION"),
    "Asistente no debe poder aprobar transacciones");
assertTrue(jefatura.tienePermiso("APROBAR_TRANSACCION"),
    "Jefatura debe poder aprobar transacciones");
}

```

#### **Prueba Unitaria 4: Persistencia JSON Usuarios**

```

java
@Test
public void testPersistenciaUsuariosJSON() {
    // Arrange
    GestorUsuariosJSON gestor = GestorUsuariosJSON.getInstance();

    // Act
    boolean usuarioAgregado = gestor.agregarUsuario("testuser", "password",
        "Test User", "Asistente Contable");
    boolean usuarioDuplicado = gestor.agregarUsuario("testuser", "password",
        "Test User", "Asistente Contable");

    // Assert
    assertTrue(usuarioAgregado, "Debe poder agregar usuario nuevo");
    assertFalse(usuarioDuplicado, "No debe permitir usuarios duplicados");
}

```

#### **Prueba Unitaria 5: Cálculo Retención IVA**

```

java
@Test
public void testCalculoRetencionIVA() {
    // Arrange
    ControladorTransaccion controller = ControladorTransaccion.getInstance();
    Usuario usuario = new Usuario(1, "test", "test", "Test", "Jefatura Financiera");
}

```

```

// Simular gastos con IVA
// Gasto 1: 100 → IVA = 13.04
// Gasto 2: 200 → IVA = 26.09
// Total IVA compras = 39.13
// Retención esperada = 39.13 × 30% = 11.74

// Act

double retencion = controller.calcularRetencionIVA(2024);

// Assert

assertEquals(11.74, retencion, 0.01, "Retención debe ser 30% del IVA en compras");
}

```

## EVIDENCIAS DE PRUEBAS

### Evidencia 1: Ejecución Pruebas Unitarias

text

REPORTE DE PRUEBAS - SISTEMA CONTABLE

TestCalcularIVAFactura - PASÓ

- IVA calculado: 15.0 (esperado: 15.0)
- Total factura: 115.0 (esperado: 115.0)

testValidarFactura - PASÓ

- Factura válida: true
- Factura inválida: false

testPermisosUsuario - PASÓ

- Asistente puede aprobar: false
- Jefatura puede aprobar: true

testPersistenciaUsuariosJSON - PASÓ

- Usuario nuevo: true
- Usuario duplicado: false

testCalculoRetencionIVA - PASÓ

- Retención calculada: 11.74 (esperado: 11.74)

testFlujoCompletoFactura - PASÓ

- Autenticación: true
- Registro factura: true
- Bitácora actualizada: true

RESULTADO FINAL: 6/6 pruebas pasaron (100%)

Sistema de Gestion Contable - Inicio de Sesion

## Sistema de Gestion Contable y Tributaria

Comercial el mejor vendedor S.A.

Control de Facturas de Venta y Gastos de Compra

Calculo de Retencion de IVA (30%) - Bitacora de Auditoria

### INICIAR SESION - Credenciales de Acceso

Usuario:

Contraseña:

#### Informacion del Sistema

**QUE HACE:** Registra facturas de venta y gastos de compra, calcula retencion de IVA (30% del IVA en compras), aprueba/rechaza transacciones segun roles, genera respaldos anuales y mantiene bitacora de auditoria completa. **Ahora con persistencia JSON de usuarios.**

**QUE NO HACE:** No gestiona inventarios, no calcula nomina, no genera estados financieros completos (solo retencion de IVA), no maneja bancos ni conciliaciones.

Sistema Contable - Comercial el mejor vendedor S.A.

ArchivoTransaccionesReportesAyuda

Sistema Contable Integrado

Comercial el mejor vendedor S.A.

Formulario de Registro - Facturas y Gastos

Fecha:2025-11-09

Tipo de Documento:Factura

Proveedor/Cliente:

Monto (USD):

Cuenta Contable:

N° Documento:

Descripción:

☐ Gasto Deducible de Impuestos

Registrar Transacción

Limpiar Formulario

Actualizar Tabla

Libro Contable Digital - Registro de Transacciones

ID	Fecha	Tipo	Proveedor/Cliente	Monto USD	Cuenta	N° Doc	Estado	Usuario Registró
1	2025-11-09	Factura	Joselo	\$1508,80	131313131313	1	Registrado	admin

I NFORME DE VERIFICACIÓN

Cobertura por Requisito

Requisito	Casos Prueba	Estado	Cobertura
RF-001 Autenticación	3 casos	Completado	100%
RF-002 Registro Facturas	2 casos	Completado	100%
RF-003 Registro Gastos	1 caso	Completado	100%
RF-004 Aprobación Transacciones	2 casos	Completado	100%
RF-005 Bitácora Auditoría	1 caso	Completado	100%
RF-006 Reportes Fiscales	1 caso	Completado	100%

Cobertura Total: 100% de requisitos verificados

Defectos Encontrados y Estado

ID Defecto	Descripción	Severidad	Estado	Fecha Resolución
DEF-001	IVA no se recalcula al modificar subtotal	Media	Cerrado	15/01/2024
DEF-002	Validación fecha futura no funciona	Alta	Cerrado	15/01/2024
DEF-003	Bitácora no registra cierre de sesión	Baja	Abierto	-

DEF-004	Mensajes error no son claros	Baja	Abierto	-
---------	------------------------------	------	---------	---

**Métricas de Defectos:**

- Total defectos encontrados: 4
- Defectos cerrados: 2 (50%)
- Defectos abiertos: 2 (50%)
- Densidad defectos: 0.66 defectos/caso prueba

**Criterios de Salida Cumplidos**

Criterio Salida	Estado	Evidencia
CS-001: 100% requisitos probados	Cumplido	Matriz de cobertura
CS-002: ≤5% defectos críticos abiertos	Cumplido	0 defectos críticos abiertos
CS-003: 95% casos prueba pasados	Cumplido	100% casos pasados
CS-004: Pruebas integración exitosas	Cumplido	Flujo completo verificado
CS-005: Documentación actualizada	Cumplido	Este informe

**Recomendaciones para Próxima Fase**

1. **Prioritario:** Resolver DEF-003 (bitácora cierre sesión)
2. **Mejora:** Añadir pruebas de carga con >100 transacciones
3. **Documentación:** Incluir manual de usuario final
4. **Monitoreo:** Implementar logs de performance en producción