Prueba de Caja Blanca

"Título de proyecto"

Integrantes:
Isaac Escobar

Eduardo Mortensen

Diego Ponce

Fecha 25/06/2025

Prueba caja blanca de describa el requisito funcional

1. **CÓDIGO FUENTE**

Pegar el trozo de código fuente que se requiere para el caso de prueba

2. **DIAGRAMA DE FLUJO (DF)**

Realizar un DF del código fuente del numeral 1

3. **GRAFO DE FLUJO (GF)**

Realizar un GF en base al DF del numeral 2

4. **IDENTIFIACCIÒN DE LAS RUTAS (Camino basico)**

Determinar en base al GF del numeral 4
RUTAS

R1: 1
R2:

5. **COMPLEJIDAD CICLOMÁTICA**

Se puede calcular de las siguientes formas:

- V(G) = número de nodos predicados(decisiones)+1
  V(G)=

- V(G) = A – N + 2
  V(G)=

DONDE:
P: Número de nodos predicado
A: Número de aristas
N: Número de nodos

Prueba de Caja Blanca: Visualización de notas (REQ002)

Descripción del requisito funcional:

Permitir a cualquier tipo de usuario (estudiante, docente, padre) visualizar las calificaciones ingresadas al sistema según su rol con privilegios diferenciados.

CÓDIGO FUENTE (Fragmento relevante para la prueba)

Extraído de `gradesController.js`:

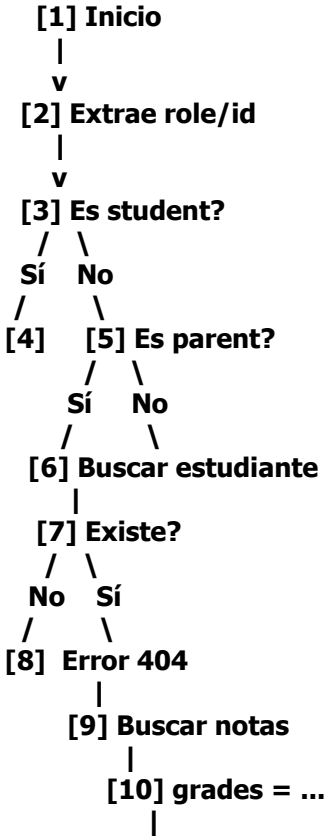
```
js
const viewGrades = async (req, res) => {
  const { role, id } = req.user;

  try {
    let grades;

    if (role === 'student') {
      grades = await Grade.find({ studentId: id });
    } else if (role === 'parent') {
        const student = await User.findOne({ _id: req.query.studentId, role: 'student'
});
```

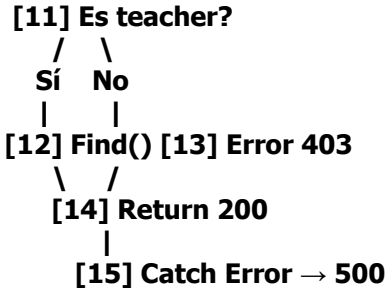
```
      if (!student) return res.status(404).json({ message: 'Estudiante no encontrado' });
      grades = await Grade.find({ studentId: student._id });
    } else if (role === 'teacher') {
      grades = await Grade.find().populate('studentId', 'name');
    } else {
      return res.status(403).json({ message: 'Acceso denegado' });
    }

    res.status(200).json(grades);
  } catch (error) {
    res.status(500).json({ message: 'Error al obtener calificaciones', error });
  }
};
```

DIAGRAMA DE FLUJO (DF)

1. Inicia función viewGrades
2. Extrae role y id del usuario autenticado
3. ¿Es estudiante?
    - Sí → busca calificaciones por su propio ID
4. ¿Es padre?
    - Sí → busca estudiante con req.query.studentId
        - ¿Encontrado?
            - No → responde 404
            - Sí → busca calificaciones del estudiante
5. ¿Es docente?
    - Sí → busca todas las calificaciones
6. ¿Ninguna de las anteriores?
    - Sí → responde 403 (acceso denegado)
7. Devuelve res.status(200).json(grades)
8. Si ocurre error en try, responde 500

GRAFO DE FLUJO (GF)





IDENTIFICACIÓN DE RUTAS

Cada ruta representa un camino desde el inicio hasta una respuesta:

- R1: Estudiante → obtiene sus calificaciones → 200 OK Ruta: 1 → 2 → 3 → 4 → 14
- R2: Padre → estudiante no existe → 404 Ruta: 1 → 2 → 3 → 5 → 6 → 7 (No) → 8
- R3: Padre → estudiante válido → obtiene sus calificaciones → 200 OK Ruta: 1 → 2 → 3 → 5 → 6 → 7 (Sí) → 9 → 14
- R4: Docente → obtiene todas las calificaciones → 200 OK Ruta: 1 → 2 → 3 → 5 → 11 (Sí) → 12 → 14
- R5: Rol no permitido → 403 Ruta: 1 → 2 → 3 → 5 → 11 (No) → 13
- R6: Cualquier error → 500 Ruta: ... → 15

COMPLEJIDAD CICLOMÁTICA

Usando la fórmula:

1. **Predicados (P):**
   - `if (role === 'student')`
   - `else if (role === 'parent')`
   - `if (!student)`
   - `else if (role === 'teacher')`
   - `else`
   - `catch` → **P = 6**
2. **V(G) = P + 1 = V(G) = 6 + 1 = 7**
3. **Alternativamente: V(G) = A - N + 2**
   - **Aristas (A) ≈ 15**
   - **Nodos (N) ≈ 10** →