Casos de Prueba para Postman - MinGO API

Esta guia contiene todos los casos de prueba para validar los endpoints de la API MinGO usando Postman.

Configuracion Inicial

Variables de Entorno

Configura las siguientes variables en Postman:

Variable	Valor Inicial	Descripcion
`base_url`	`http://localhost:3000`	URL base de la API
`access_token`	(vacio)	Token JWT de acceso
`refresh_token`	(vacio)	Token JWT de refresco
`user_id`	(vacio)	ID del usuario autenticado
`child_id`	(vacio)	ID del hijo creado
`class_id`	(vacio)	ID de la clase creada
`class_code`	(vacio)	Codigo de la clase
`verification_token`	(vacio)	Token de verificacion de email
`reset_token`	(vacio)	Token de reset de password

Headers Comunes

Para endpoints protegidos, usar:

```
Authorization: Bearer {{access_token}}
Content-Type: application/json
```

---

1. AUTH - Autenticacion

1.1 Registro de Usuario

Endpoint: `POST {{base_url}}/auth/register`

Caso 1.1.1: Registro exitoso de PADRE

```json
{
  "email": "padre@test.com",
  "password": "Test123!@",
  "name": "Juan Perez",
  "role": "PADRE"
}
```

Respuesta esperada: `201 Created`

```
{
  "accessToken": "eyJhbG...",
  "refreshToken": "eyJhbG...",
  "user": {
    "id": "uuid",
    "email": "padre@test.com",
    "name": "Juan Perez",
    "role": "PADRE",
    "isActive": true,
    "emailVerified": false,
    "createdAt": "2024-01-01T00:00:00.000Z"
  }
}
```

Post-script:

```
if (pm.response.code === 201) {
    var json = pm.response.json();
    pm.environment.set("access_token", json.accessToken);
    pm.environment.set("refresh_token", json.refreshToken);
    pm.environment.set("user_id", json.user.id);
}
```

Caso 1.1.2: Registro exitoso de DOCENTE

```
{
  "email": "docente@test.com",
  "password": "Test123!@",
  "name": "Maria Garcia",
  "role": "DOCENTE"
}
```

Respuesta esperada: `201 Created`

Caso 1.1.3: Error - Email ya registrado

```
{
  "email": "padre@test.com",
  "password": "Test123!@",
  "name": "Otro Usuario",
  "role": "PADRE"
}
```

Respuesta esperada: `400 Bad Request`

```
{
  "statusCode": 400,
  "message": "Email already registered"
}
```

Caso 1.1.4: Error - Email invalido

```
{
  "email": "email-invalido",
  "password": "Test123!@",
  "name": "Usuario",
  "role": "PADRE"
}
```

Respuesta esperada: `400 Bad Request`

Caso 1.1.5: Error - Password muy corta

```
{
  "email": "nuevo@test.com",
  "password": "123",
  "name": "Usuario",
  "role": "PADRE"
}
```

Respuesta esperada: `400 Bad Request`

Caso 1.1.6: Error - Nombre muy corto

```
{
  "email": "nuevo@test.com",
  "password": "Test123!@",
  "name": "A",
  "role": "PADRE"
}
```

Respuesta esperada: `400 Bad Request`

Caso 1.1.7: Error - Rol invalido

```
{
  "email": "nuevo@test.com",
  "password": "Test123!@",
  "name": "Usuario",
  "role": "INVALID_ROLE"
}
```

Respuesta esperada: `400 Bad Request`

1.2 Login

Endpoint: `POST {{base_url}}/auth/login`

Caso 1.2.1: Login exitoso

```
{
  "email": "padre@test.com",
  "password": "Test123!@"
}
```

Respuesta esperada: `200 OK`

```
{
  "accessToken": "eyJhbG...",
  "refreshToken": "eyJhbG...",
  "user": {
    "id": "uuid",
    "email": "padre@test.com",
    "name": "Juan Perez",
    "role": "PADRE"
  }
}
```

Post-script:

```
if (pm.response.code === 200) {
    var json = pm.response.json();
    pm.environment.set("access_token", json.accessToken);
    pm.environment.set("refresh_token", json.refreshToken);
    pm.environment.set("user_id", json.user.id);
}
```

Caso 1.2.2: Error - Credenciales invalidas

```
{
  "email": "padre@test.com",
  "password": "wrongpassword"
}
```

Respuesta esperada: `401 Unauthorized`

```
{
  "statusCode": 401,
  "message": "Invalid credentials"
}
```

Caso 1.2.3: Error - Usuario no existe

```
{
  "email": "noexiste@test.com",
  "password": "Test123!@"
}
```

Respuesta esperada: `401 Unauthorized`

1.3 Refresh Token

Endpoint: `POST {{base_url}}/auth/refresh`

Caso 1.3.1: Refresh exitoso

```
{
  "refreshToken": "{{refresh_token}}"
}
```

Respuesta esperada: `200 OK` **Post-script:**

```
if (pm.response.code === 200) {
    var json = pm.response.json();
    pm.environment.set("access_token", json.accessToken);
    pm.environment.set("refresh_token", json.refreshToken);
}
```

Caso 1.3.2: Error - Token invalido

```
{
  "refreshToken": "token-invalido"
}
```

Respuesta esperada: `401 Unauthorized`

1.4 Obtener Perfil

Endpoint: `GET {{base_url}}/auth/profile` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 1.4.1: Perfil obtenido exitosamente **Respuesta esperada:** `200 OK`

```
{
  "id": "uuid",
  "email": "padre@test.com",
  "name": "Juan Perez",
  "role": "PADRE",
  "isActive": true,
  "emailVerified": false,
  "createdAt": "2024-01-01T00:00:00.000Z"
}
```

Caso 1.4.2: Error - Sin token **Headers:** (sin Authorization) **Respuesta esperada:** `401 Unauthorized`

Caso 1.4.3: Error - Token expirado/invalido **Headers:** `Authorization: Bearer token-invalido` **Respuesta esperada:** `401 Unauthorized`

1.5 Verificacion de Email

Endpoint: `POST {{base_url}}/auth/verify-email`

Caso 1.5.1: Verificacion exitosa

```
{
  "token": "{{verification_token}}"
}
```

Respuesta esperada: `200 OK`

```
{
  "success": true,
  "message": "Email verified successfully. Welcome to MinGO!"
}
```

Caso 1.5.2: Error - Token invalido

```
{
  "token": "token-invalido"
}
```

Respuesta esperada: `400 Bad Request`

```
{
  "statusCode": 400,
  "message": "Invalid or expired verification token"
}
```

1.6 Reenviar Verificacion

Endpoint: `POST {{base_url}}/auth/resend-verification`

Caso 1.6.1: Reenvio exitoso

```
{
  "email": "padre@test.com"
}
```

Respuesta esperada: `200 OK`

```
{
  "success": true,
  "message": "If an account exists with this email, a verification link has been sent."
}
```

Caso 1.6.2: Email no existente (mismo mensaje por seguridad)

```
{
  "email": "noexiste@test.com"
}
```

Respuesta esperada: `200 OK`

1.7 Olvide mi Contrasena

Endpoint: `POST {{base_url}}/auth/forgot-password`

Caso 1.7.1: Solicitud exitosa

```
{
  "email": "padre@test.com"
}
```

Respuesta esperada: `200 OK`

```
{
  "success": true,
  "message": "If an account exists with this email, a password reset link has been sent."
}
```

1.8 Restablecer Contrasena

Endpoint: `POST {{base_url}}/auth/reset-password`

Caso 1.8.1: Reset exitoso

```
{
  "token": "{{reset_token}}",
  "newPassword": "NewPass123!@"
}
```

Respuesta esperada: `200 OK`

```
{
  "success": true,
  "message": "Password has been reset successfully. You can now login with your new passwor
}
```

Caso 1.8.2: Error - Token invalido

```
{
  "token": "token-invalido",
  "newPassword": "NewPass123!@"
}
```

Respuesta esperada: `400 Bad Request`

Caso 1.8.3: Error - Password debil

```
{
  "token": "{{reset_token}}",
  "newPassword": "123"
}
```

Respuesta esperada: `400 Bad Request`

---

2. CHILDREN - Gestion de Hijos

Nota: Todos los endpoints requieren autenticacion y rol PADRE.

2.1 Crear Hijo

Endpoint: `POST {{base_url}}/children` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 2.1.1: Creacion exitosa

```
{
  "name": "Pedrito Perez",
  "birthDate": "2019-05-15",
  "disabilityType": "AUDITIVA",
  "notes": "Le gusta aprender jugando"
}
```

Respuesta esperada: `201 Created`

```
{
  "id": "uuid",
  "parentId": "uuid",
  "name": "Pedrito Perez",
  "birthDate": "2019-05-15T00:00:00.000Z",
  "age": 5,
  "ageCategory": "3-5 años",
  "disabilityType": "AUDITIVA",
  "notes": "Le gusta aprender jugando",
  "createdAt": "...",
```

```
  "updatedAt": "..."
}
```

Post-script:

```
if (pm.response.code === 201) {
    var json = pm.response.json();
    pm.environment.set("child_id", json.id);
}
```

Caso 2.1.2: Creacion sin tipo de discapacidad

```
{
  "name": "Maria Perez",
  "birthDate": "2020-08-20"
}
```

Respuesta esperada: `201 Created`

Caso 2.1.3: Error - Nombre vacio

```
{
  "name": "",
  "birthDate": "2019-05-15"
}
```

Respuesta esperada: `400 Bad Request`

Caso 2.1.4: Error - Edad fuera de rango (muy pequeno)

```
{
  "name": "Bebe",
  "birthDate": "2024-01-01"
}
```

Respuesta esperada: `400 Bad Request`

Caso 2.1.5: Error - Edad fuera de rango (muy grande)

```
{
  "name": "Adulto",
  "birthDate": "2000-01-01"
}
```

Respuesta esperada: `400 Bad Request`

Caso 2.1.6: Error - Tipo de discapacidad invalido

```
{
  "name": "Test",
```

```
  "birthDate": "2019-05-15",
  "disabilityType": "INVALIDO"
}
```

Respuesta esperada: `400 Bad Request`

Caso 2.1.7: Error - Usuario no es PADRE **Nota:** Hacer login como DOCENTE primero

```
{
  "name": "Test",
  "birthDate": "2019-05-15"
}
```

Respuesta esperada: `403 Forbidden`

2.2 Listar Hijos

Endpoint: `GET {{base_url}}/children` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 2.2.1: Lista exitosa **Respuesta esperada:** `200 OK`

```
{
  "children": [
    {
      "id": "uuid",
      "parentId": "uuid",
      "name": "Pedrito Perez",
      "birthDate": "2019-05-15T00:00:00.000Z",
      "age": 5,
      "ageCategory": "3-5 años",
      "disabilityType": "AUDITIVA"
    }
  ],
  "total": 1
}
```

Caso 2.2.2: Lista vacia (usuario sin hijos) **Respuesta esperada:** `200 OK`

```
{
  "children": [],
  "total": 0
}
```

2.3 Obtener Hijo por ID

Endpoint: `GET {{base_url}}/children/{{child_id}}` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 2.3.1: Hijo encontrado **Respuesta esperada:** `200 OK`

Caso 2.3.2: Error - Hijo no encontrado **Endpoint:** `GET {{base_url}}/children/uuid-no-existente` **Respuesta esperada:** `404 Not Found`

Caso 2.3.3: Error - Hijo de otro padre **Nota:** Intentar acceder al hijo de otro usuario **Respuesta esperada:** `403 Forbidden`

2.4 Actualizar Hijo

Endpoint: `PUT {{base_url}}/children/{{child_id}}` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 2.4.1: Actualizacion exitosa

```
{
  "name": "Pedro Perez Garcia",
  "notes": "Notas actualizadas"
}
```

Respuesta esperada: `200 OK`

Caso 2.4.2: Actualizar tipo de discapacidad

```
{
  "disabilityType": "AMBAS"
}
```

Respuesta esperada: `200 OK`

Caso 2.4.3: Error - Nombre muy largo

```
{
  "name": "A".repeat(101)
}
```

Respuesta esperada: `400 Bad Request`

2.5 Eliminar Hijo

Endpoint: `DELETE {{base_url}}/children/{{child_id}}` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 2.5.1: Eliminacion exitosa **Respuesta esperada:** `204 No Content`

Caso 2.5.2: Error - Hijo no encontrado **Respuesta esperada:** `404 Not Found`

---

3. PROGRESS - Progreso de Aprendizaje

3.1 Registrar Intento

Endpoint: `POST {{base_url}}/progress/attempt` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 3.1.1: Intento correcto

```
{
  "lessonId": "{{lesson_id}}",
  "activityId": "{{activity_id}}",
  "correct": true,
  "timeSpent": 30
}
```

Respuesta esperada: `201 Created`

```
{
  "id": "uuid",
  "userId": "uuid",
  "lessonId": "uuid",
  "completed": false,
  "accuracy": 100,
  "totalAttempts": 1,
  "correctAttempts": 1,
  "timeSpent": 30
}
```

Caso 3.1.2: Intento incorrecto

```
{
  "lessonId": "{{lesson_id}}",
  "activityId": "{{activity_id}}",
  "correct": false,
  "timeSpent": 45
}
```

Respuesta esperada: `201 Created`

Caso 3.1.3: Error - Tiempo negativo

```
{
  "lessonId": "{{lesson_id}}",
  "activityId": "{{activity_id}}",
  "correct": true,
  "timeSpent": -10
}
```

Respuesta esperada: `400 Bad Request`

3.2 Completar Leccion

Endpoint: `POST {{base_url}}/progress/complete` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 3.2.1: Leccion completada exitosamente

```
{
  "lessonId": "{{lesson_id}}"
}
```

Respuesta esperada: `200 OK`

```
{
  "progress": {
    "id": "uuid",
    "completed": true,
    "completedAt": "2024-01-01T00:00:00.000Z"
  },
  "levelUnlocked": null
}
```

Caso 3.2.2: Leccion completada desbloquea nivel **Respuesta esperada:** `200 OK`

```
{
  "progress": { ... },
  "levelUnlocked": {
    "id": "uuid",
    "levelSectionId": "uuid",
    "levelName": "Intermedio",
    "unlockedAt": "..."
  }
}
```

3.3 Obtener Progreso

Endpoint: `GET {{base_url}}/progress` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 3.3.1: Obtener todo el progreso **Respuesta esperada:** `200 OK`

Caso 3.3.2: Filtrar por leccion **Endpoint:** `GET {{base_url}}/progress?lessonId={{lesson_id}}` **Respuesta esperada:** `200 OK`

3.4 Obtener Estadisticas

Endpoint: `GET {{base_url}}/progress/stats` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 3.4.1: Estadisticas del usuario **Respuesta esperada:** `200 OK`

```
{
  "userId": "uuid",
  "totalLessonsStarted": 5,
  "totalLessonsCompleted": 3,
  "averageAccuracy": 85.5,
  "totalTimeSpent": 1800,
  "unlockedLevels": ["Principiante", "Intermedio"]
}
```

---

4. CONTENT - Contenido Educativo

4.1 Obtener Categorias de Edad (Publico)

Endpoint: `GET {{base_url}}/content/age-categories`

Caso 4.1.1: Listar categorias **Respuesta esperada:** `200 OK`

```
[
  {
    "id": "uuid",
    "name": "1-3 años",
    "minAge": 1,
    "maxAge": 3,
    "orderIndex": 1
  },
```

```
  {
    "id": "uuid",
    "name": "3-5 años",
    "minAge": 3,
    "maxAge": 5,
    "orderIndex": 2
  }
]
```

4.2 Obtener Niveles

Endpoint: `GET {{base_url}}/content/levels` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 4.2.1: Listar niveles con estado de desbloqueo **Respuesta esperada:** `200 OK`

```
[
  {
    "id": "uuid",
    "name": "Principiante",
    "level": "PRINCIPIANTE",
    "orderIndex": 1,
    "isUnlocked": true
  },
  {
    "id": "uuid",
    "name": "Intermedio",
    "level": "INTERMEDIO",
    "orderIndex": 2,
    "isUnlocked": false
  }
]
```

4.3 Obtener Categorias de Contenido (Publico)

Endpoint: `GET {{base_url}}/content/categories`

Caso 4.3.1: Listar categorias **Respuesta esperada:** `200 OK`

```
[
  {
    "id": "uuid",
    "name": "Alfabeto",
```

```
    "description": "Aprende el alfabeto en LSE",
    "orderIndex": 1
  }
]
```

4.4 Obtener Modulos

Endpoint: `GET {{base_url}}/content/modules` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 4.4.1: Listar todos los modulos **Respuesta esperada:** `200 OK`

Caso 4.4.2: Filtrar por nivel **Endpoint:** `GET {{base_url}}/content/modules?levelSectionId={{leve`
Respuesta esperada: `200 OK`

4.5 Obtener Lecciones de un Modulo

Endpoint: `GET {{base_url}}/content/modules/{{module_id}}/lessons`
Headers: `Authorization: Bearer {{access_token}}`

Caso 4.5.1: Listar lecciones **Respuesta esperada:** `200 OK`

```
[
  {
    "id": "uuid",
    "moduleId": "uuid",
    "title": "Leccion 1: Saludos",
    "orderIndex": 1,
    "duration": 15,
    "isActive": true,
    "activitiesCount": 5
  }
]
```

4.6 Obtener Leccion por ID

Endpoint: `GET {{base_url}}/content/lessons/{{lesson_id}}` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 4.6.1: Leccion encontrada **Respuesta esperada:** `200 OK`

Caso 4.6.2: Error - Leccion no encontrada **Respuesta esperada:** `404 Not Found`

4.7 Obtener Actividades de una Leccion

Endpoint: `GET {{base_url}}/content/lessons/{{lesson_id}}/activities`
Headers: `Authorization: Bearer {{access_token}}`

Caso 4.7.1: Listar actividades **Respuesta esperada:** `200 OK`

```
[
  {
    "id": "uuid",
    "lessonId": "uuid",
    "title": "Practica: Saludo formal",
    "activityType": "VIDEO_PRACTICE",
    "orderIndex": 1,
    "points": 10,
    "isActive": true
  }
]
```

---

5. CLASSES - Gestion de Clases

5.1 Crear Clase (DOCENTE/ADMIN)

Endpoint: `POST {{base_url}}/classes` **Headers:** `Authorization: Bearer {{access_token}}` **Nota:** Requiere login como DOCENTE

Caso 5.1.1: Creacion exitosa

```
{
  "name": "Clase de LSE Basico",
  "description": "Clase para principiantes"
}
```

Respuesta esperada: `201 Created`

```
{
  "id": "uuid",
  "teacherId": "uuid",
  "teacherName": "Maria Garcia",
  "name": "Clase de LSE Basico",
  "code": "ABC123",
  "description": "Clase para principiantes",
```

```
  "isActive": true,
  "studentsCount": 0,
  "createdAt": "..."
}
```

Post-script:

```
if (pm.response.code === 201) {
    var json = pm.response.json();
    pm.environment.set("class_id", json.id);
    pm.environment.set("class_code", json.code);
}
```

Caso 5.1.2: Error - Usuario PADRE intenta crear **Nota:** Login como PADRE **Respuesta esperada:** `403 Forbidden`

Caso 5.1.3: Error - Nombre muy corto

```
{
  "name": "A"
}
```

Respuesta esperada: `400 Bad Request`

5.2 Listar Clases que Enseno

Endpoint: `GET {{base_url}}/classes/teaching` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.2.1: Listar clases del docente **Respuesta esperada:** `200 OK`

```
{
  "classes": [
    {
      "id": "uuid",
      "name": "Clase de LSE Basico",
      "code": "ABC123",
      "studentsCount": 5
    }
  ],
  "total": 1
}
```

5.3 Actualizar Clase

Endpoint: `PUT {{base_url}}/classes/{{class_id}}` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.3.1: Actualizacion exitosa

```
{
  "name": "Clase de LSE Basico - Actualizada",
  "description": "Descripcion actualizada"
}
```

Respuesta esperada: `200 OK`

Caso 5.3.2: Error - Clase de otro docente **Respuesta esperada:** `403 Forbidden`

5.4 Eliminar Clase

Endpoint: `DELETE {{base_url}}/classes/{{class_id}}` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.4.1: Eliminacion exitosa **Respuesta esperada:** `204 No Content`

5.5 Crear Asignacion

Endpoint: `POST {{base_url}}/classes/{{class_id}}/assignments` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.5.1: Asignacion exitosa

```
{
  "lessonId": "{{lesson_id}}",
  "title": "Tarea: Practica de saludos",
  "description": "Completar la leccion de saludos",
  "dueDate": "2024-12-31T23:59:59.000Z"
}
```

Respuesta esperada: `201 Created`

Caso 5.5.2: Asignacion sin fecha limite

```
{
  "lessonId": "{{lesson_id}}",
  "title": "Practica opcional"
}
```

Respuesta esperada: `201 Created`

5.6 Unirse a Clase (Estudiante)

Endpoint: `POST {{base_url}}/classes/join` **Headers:** `Authorization: Bearer {{access_token}}` **Nota:** Login como PADRE

Caso 5.6.1: Union exitosa

```
{
  "code": "{{class_code}}"
}
```

Respuesta esperada: `201 Created`

```
{
  "id": "uuid",
  "classId": "uuid",
  "studentId": "uuid",
  "studentName": "Juan Perez",
  "enrolledAt": "..."
}
```

Caso 5.6.2: Error - Codigo invalido

```
{
  "code": "INVALID"
}
```

Respuesta esperada: `404 Not Found`

Caso 5.6.3: Error - Ya inscrito

```
{
  "code": "{{class_code}}"
}
```

Respuesta esperada: `400 Bad Request`

5.7 Salir de Clase

Endpoint: `DELETE {{base_url}}/classes/{{class_id}}/leave` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.7.1: Salida exitosa **Respuesta esperada:** `204 No Content`

5.8 Listar Clases Inscritas

Endpoint: `GET {{base_url}}/classes/enrolled` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.8.1: Listar clases **Respuesta esperada:** `200 OK`

```
{
  "classes": [...],
  "total": 2
}
```

5.9 Obtener Clase por ID

Endpoint: `GET {{base_url}}/classes/{{class_id}}` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.9.1: Clase encontrada **Respuesta esperada:** `200 OK`

Caso 5.9.2: Error - Clase no encontrada **Respuesta esperada:** `404 Not Found`

5.10 Listar Asignaciones de Clase

Endpoint: `GET {{base_url}}/classes/{{class_id}}/assignments` **Headers:** `Authorization: Bearer {{access_token}}`

Caso 5.10.1: Listar asignaciones **Respuesta esperada:** `200 OK`

```
[
  {
    "id": "uuid",
    "classId": "uuid",
    "lessonId": "uuid",
    "lessonTitle": "Leccion 1: Saludos",
    "title": "Tarea: Practica de saludos",
    "dueDate": "2024-12-31T23:59:59.000Z",
    "isOverdue": false,
    "createdAt": "..."
  }
]
```

Resumen de Casos de Prueba

Modulo	Casos	Exitosos	Errores
Auth	20	10	10
Children	12	6	6
Progress	8	6	2
Content	8	8	0
Classes	15	10	5
Total	**63**	**40**	**23**

Coleccion Postman

Para importar estos casos en Postman:

1. Crear nueva coleccion "MinGO API Tests"
2. Configurar variables de entorno
3. Crear carpetas por modulo (Auth, Children, Progress, Content, Classes)
4. Agregar requests siguiendo los casos documentados
5. Agregar tests automatizados usando los post-scripts proporcionados

Script de Pre-request Global

```
// Verificar si tenemos token valido
if (!pm.environment.get("access_token")) {
    console.log("No hay token de acceso configurado");
}
```

Script de Test Global para Errores

```
// Verificar respuestas de error tienen formato correcto
if (pm.response.code >= 400) {
    pm.test("Error response has correct format", function() {
        var json = pm.response.json();
        pm.expect(json).to.have.property("statusCode");
        pm.expect(json).to.have.property("message");
    });
}
```