Plan de pruebas - API

# Editor:

Reinaldo Espinel Torres

Plataforma de Orquestación para Automatizaciones RPA

**05/06/2024**

1. **Introducción**

El propósito de este plan de pruebas es definir las estrategias y métodos que se utilizarán para validar la funcionalidad, seguridad, rendimiento y mantenibilidad de la API de la Plataforma de Orquestación para Automatizaciones RPA. Esta API está diseñada para gestionar ejecuciones de automatizaciones, simular claves seguras y autorizar ejecuciones en máquinas específicas.

1. **Alcance**

El alcance de las pruebas incluye los siguientes módulos:

1. **Gestión de Ejecuciones de Automatizaciones**
   * Iniciar y finalizar ejecuciones.
2. **Simulación de Claves Seguras de Microsoft**
   * Generación y validación de claves seguras.
3. **Autorización de Ejecuciones en Máquinas Específicas**
   * Permitir o denegar ejecuciones en máquinas específicas.

**3. Estrategia de Pruebas**

**3.1 Tipos de Pruebas**

1. **Pruebas Funcionales**: Validan la funcionalidad de cada endpoint.
2. **Pruebas de Integración**: Aseguran que los módulos interactúan correctamente entre sí.
3. **Pruebas de Rendimiento**: Evaluan la capacidad de respuesta bajo diferentes condiciones de carga.
4. **Pruebas de Seguridad**: Verifican que la API maneje adecuadamente la autenticación, autorización y protección de datos.
5. **Pruebas de Mantenibilidad**: Comprueban que la documentación esté completa y las pruebas de regresión estén implementadas.

**3.2 Herramientas de Pruebas**

Dado que todas las pruebas serán manuales, se utilizarán herramientas básicas de desarrollo y postman para la ejecución y validación de los endpoints de la API.

**4. Plan de Ejecución de Pruebas**

**4.1 Pruebas Funcionales**

**4.1.1 Gestión de Ejecuciones de Automatizaciones**

* **POST /api/executions/start**
  + Validar respuesta con datos válidos: Enviar una solicitud POST con un JSON válido y verificar que la API responde con un código de éxito y los datos esperados.
  + Probar manejo de errores con datos inválidos: Enviar una solicitud POST con un JSON inválido y verificar que la API responde con un error apropiado.
* **POST /api/executions/end**
  + Verificar respuesta con un ID de ejecución válido: Enviar una solicitud POST con un ID de ejecución válido y verificar la respuesta de éxito.
  + Probar manejo de errores con ID inexistente: Enviar una solicitud POST con un ID de ejecución inexistente y verificar que la API responde con un error adecuado.

**4.1.2 Simulación de Claves Seguras de Microsoft**

* **POST /api/keys/generate**
  + Comprobar generación de clave con datos válidos: Enviar una solicitud POST con los detalles necesarios y verificar que se genera una clave segura correctamente.
* **POST /api/keys/validate**
  + Validar clave con datos correctos: Enviar una solicitud POST con una clave válida y verificar que la validación es exitosa.
  + Probar manejo de errores con clave inválida: Enviar una solicitud POST con una clave inválida y verificar que la API responde con un error de validación.

**4.1.3 Autorización de Ejecuciones en Máquinas Específicas**

* **POST /api/authorization/grant**
  + Verificar autorización con datos correctos: Enviar una solicitud POST con los detalles de la máquina y la automatización, y verificar que se concede la autorización.
* **POST /api/authorization/deny**
  + Validar denegación con datos correctos: Enviar una solicitud POST con los detalles de la máquina y la automatización, y verificar que se deniega la autorización.

**4.2 Pruebas de Integración**

* Validar el flujo completo de una ejecución de automatización, desde el inicio hasta el fin, asegurándose de que todas las partes del sistema funcionen juntas sin problemas.
* Probar la integración entre la generación y validación de claves, asegurando que las claves generadas puedan ser validadas correctamente.
* Verificar el proceso de autorización para asegurarse de que las decisiones de permiso/denegación se manejen correctamente a lo largo del sistema.

**4.3 Pruebas de Rendimiento**

* Simular múltiples solicitudes concurrentes manualmente para evaluar el tiempo de respuesta de la API bajo diferentes condiciones de carga.
* Medir el tiempo de respuesta para las operaciones críticas, asegurando que sean menores a 500ms en condiciones normales.

**4.4 Pruebas de Seguridad**

* Verificar manualmente el uso de tokens JWT para autenticación y autorización.
* Asegurar que los datos sensibles están encriptados en tránsito y en reposo mediante la verificación de las configuraciones y el análisis de tráfico.
* Probar la resistencia a ataques comunes como inyección SQL, XSS y CSRF mediante la introducción de entradas maliciosas y la observación de la respuesta del sistema.

**4.5 Pruebas de Mantenibilidad**

* Revisar manualmente la documentación de la API para asegurarse de que esté actualizada y completa.
* Implementar pruebas de regresión manuales para asegurar que nuevas actualizaciones no rompan funcionalidades existentes.