Tecnológico de Monterrey

Desarrollo e implantación de sistemas de software

Grupo 106

Diseño de pruebas

Profesores:

Adan Octavio Ruiz Martinez

Jorge Alvarez Bujanos

Leonardo S. Gamez Peña

Lorena Guadalupe Gómez Martínez

Alberto Emmanuel Benavides Contreras

Equipo 6:

Eugenia Uresti Arriaga A01284839

Mónica Parra Franco A01251941

Gerardo Manzur Morales A01177622

Jorge Nuñez Gurrola A00833455

Javier Banegas A00827812

13 de junio de 2024

1. Introducción

1.1 Propósito

El propósito principal de las pruebas es asegurar que el Oracle Java Chat Bot cumpla con todos los requerimientos funcionales y no funcionales especificados en el documento de Especificaciones de Requerimientos de Software (SRS). Buscamos identificar y corregir cualquier defecto en las etapas de desarrollo para minimizar los costos de corrección y evitar impactos negativos en la calidad del proyecto final.

1.2 Alcance

El alcance de las pruebas se enfoca en garantizar la calidad y eficiencia del sistema mediante una serie exhaustiva de pruebas. Estas pruebas incluyen:

- **Pruebas de funcionalidad**: Validar que el software cumpla con todas las funciones y requisitos especificados. Esto implica probar la entrada y salida de datos, así como funciones individuales y su interacción dentro del sistema.
- **Pruebas de rendimiento**: Validar que el sistema responda bajo diferentes condiciones de carga, midiendo aspectos como la velocidad, escalabilidad, estabilidad y eficiencia.
- **Pruebas de integración**: Verificar la comunicación adecuada entre todos los componentes y sistemas externos, asegurando su integración adecuada en el entorno operativo.
- **Pruebas de usabilidad**: Asegurar una experiencia de usuario intuitiva.

Este enfoque integral busca cubrir todos los aspectos críticos del sistema, desde su funcionalidad y seguridad hasta su rendimiento y usabilidad.

1.3 Definiciones

- CI/CD: Integración continua y Entregas continuas
- API: Interfaz de programación de aplicaciones
- UAT: Pruebas de Aceptación de Usuario

1.4 Referencias

- IEEE Std 730-1998, IEEE Standard for Software Quality Assurance Plans.
- Documento de Requisitos del Sistema SRS, basado en IEEE Std 830-1998.
- Documento de Diseño del Sistema SDD, basado en IEEE Std 1016-2009.

2. Enfoque de Pruebas

2.1 Tipo de Pruebas

2.1.1 Pruebas Funcionales

Validar que el sistema cumpla con todos los requisitos del usuario. Estas pruebas se centran en verificar las funcionalidades especificadas en el SRS. Se verifica cada funcionalidad individual para asegurar que el sistema opere según lo esperado. Se utiliza la herramienta Selenium para pruebas de interfaz de usuario.

2.1.2 Pruebas de Integración

Asegurar que la interacción entre todos los componentes del sistema sea correcta y que las dependencias entre módulos estén adecuadamente manejadas. Docker se utilizará para mantener entornos consistentes, y Spring Boot se emplea para manejar microservicios.

2.1.3 Pruebas de Sistema

Validar el sistema en su totalidad, asegurando que los requerimientos no funcionales y funcionales estén funcionando correctamente. Herramientas como Docker, Spring Boot y Oracle Cloud Infrastructure se utilizarán para esta tarea.

2.1.4 Pruebas de Rendimiento

Verificar que el sistema maneje cargas de trabajo de al menos 60 usuarios simultáneos sin degradar el rendimiento. Se medirán métricas como el tiempo de respuesta, la velocidad de procesamiento y la estabilidad bajo carga.

2.1.5 Pruebas de Usabilidad

Validar que la interfaz sea eficiente y fácil de usar, asegurando que los usuarios puedan completar las tareas sin dificultades. Se realizarán pruebas manuales con usuarios reales.

2.1.6 Pruebas de Seguridad

Validar que los datos están encriptados en tránsito y en reposo durante la comunicación y en la base de datos. OWASP ZAP se utilizará para realizar pruebas de seguridad y detectar vulnerabilidades.

2.2 Estrategia de Pruebas

- **Automatización de Pruebas**: Se utilizarán herramientas de automatización como Selenium para ejecutar pruebas unitarias y de integración.

- **Pruebas Manuales**: Realizar pruebas manuales para escenarios complejos y pruebas de aceptación del usuario.Esto incluye validar casos de uso críticos que requieren juicio humano y escenarios que no pueden ser fácilmente automatizados. Documentación detallada y scripts de prueba serán utilizados para guiar estas pruebas.
- **Pruebas en Entorno de Staging**: Realizar pruebas en un entorno que refleje la configuración de producción para asegurar la validez de los resultados. Se utilizarán herramientas como Docker y Oracle Cloud Infrastructure para mantener la consistencia y relevancia del entorno de pruebas.

3. Ambiente de Pruebas

3.1 Hardware

- Computadoras: Para el equipo de desarrollo y pruebas, se requieren computadoras con capacidad
- Teléfonos Móviles: Smartphones con diversos sistemas operativos (Android, iOS) para pruebas de interacción con el bot de Telegram y garantizar compatibilidad y rendimiento.

3.2 Software

- Docker: Para la creación y gestión de contenedores, permitiendo un ambiente de pruebas consistente.
- Java SE Development Kit (JDK) 8: Se utilizará Java Development Kit versión 1.8 para el desarrollo del bot.
- Apache Maven: Para la gestión de dependencias y automatización de la construcción del proyecto.
- Telegram Bot API: Para interactuar con el bot durante las pruebas.
- Spring Boot: Para el desarrollo del back-end del bot, utilizando Spring Boot para facilitar la creación de microservicios escalables.
- Oracle Cloud Infrastructure y Oracle Autonomous Database: Para alojar la aplicación y la base de datos, asegurando el acceso durante las pruebas a los servicios en la nube de Oracle.
- Selenium: Utilizado para pruebas de interfaz de usuario en Telegram.
- OWASP ZAP: Herramienta para realizar pruebas de seguridad y detectar vulnerabilidades, asegurando que los datos estén encriptados tanto en tránsito como en reposo.

4. Especificaciones de Pruebas

4.1 Casos de Uso

Número	Pre condiciones	Descripción	Flujo del caso de prueba	Resultado esperado
1	Estar registrado en la base de datos	Inicio de sesión	El usuario inicia sesión con su cuenta de oracle, ya sea con cuenta de desarrollador o manager	El usuario es dirigido a la interfaz principal que le corresponde
2	Entrar con perfil de manager	El usuario (manager) podrá crear una nueva tarea, llenando diferentes campos que el ChatBot solicitara	El usuario da click en el el boton de “Agregar tarea”, y se le pide llenar los siguientes campos: - Título - Descripción - Prioridad - Estatus - Fecha límite - Responsable	La tarea es registrada correctamente en la base de datos
3	Entrar con perfil de desarrollador	El usuario (desarrollador) puede crear una nueva tarea, llenando diferentes campos solicitados por el ChatBot	El usuario da click en el botón de “Agregar tarea”, y se le solicita llenar los siguientes campos: - Título - Descripción - Prioridad - Estatus - Fecha límite - Responsable	La tarea se registra correctamente en la base de datos

4	Entrar con perfil de desarrollador	Listado de tareas dividido entre las tareas completadas y no completadas	El usuario da click en el botón de “Ver tareas”	El Chatbot responde con el listado de todas las tareas del usuario, donde en la parte inferior se muestran las tareas ya completadas y en la parte superior se muestran las tareas sin completar
5	Entrar con perfil de manager	Listado de tareas de todo el equipo, dividido entre las tareas completadas y no completadas	El usuario da click en el botón de “Ver tareas”	El Chatbot responde con el listado de todas las tareas de cada desarrollador bajo su responsabilidad, donde en la parte inferior se muestran las tareas ya completadas y en la parte superior se muestran las tareas sin

				completar
6	Entrar con perfil de manager o desarrollador	Modificación de tareas	El usuario da click en el botón “Modificar tarea” y el ChatBot solicita indicar qué campo modificar y a que se va a modificar	Se modifica y se guarda la tarea correctamente en la base de datos
7	Entrar con perfil de manager o desarrollador	Eliminar tarea	El usuario da click en el botón “Eliminar tarea” y el ChatBot solicita el título de la tarea a eliminar	La tarea se elimina correctamente de la base de datos y no vuelve a aparecer en el listado de tareas
8	Entrar con perfil de manager o desarrollador	Marcar tarea como completada	El usuario da click en el botón “Done” y el ChatBot solicita la tarea a marcar como completada	La tarea se actualiza correctamente en la base de datos y aparece en la sección de tareas completadas en el listado de tareas
9	Entrar con perfil de manager o desarrollador	Reactivar tarea ya completada, como pendiente	El usuario da click en el botón “Undone” y solicita la tarea a reactivar	Se cambia el estatus a pendiente correctamente

				en la base de datos y en el listado de tareas vuelve a aparecer en la sección de tareas sin completar
10	Entrar con perfil de manager o desarrollador	Filtrar tareas	El usuario da click en el botón “Filtrar” y el ChatBot solicita el campo a filtrar	Se muestra un listado de las tareas filtradas
11	Entrar con perfil de manager o desarrollador	Recibir sugerencias/atajos después de cada comando	El usuario termina de agregar, eliminar, modificar, filtrar o ver tareas	El ChatBot responde con algún atajo o sugerencia a lo que se puede hacer para el siguiente paso
12	Entrar con perfil de manager o desarrollador	Recibir ayuda después de ingresar comandos incorrectos	El usuario solicita un comando incorrecto o no existente	El ChatBot responde con un mensaje de aviso y ayuda
13	Entrar con perfil de manager o desarrollador	Regresar a pantalla principal	El usuario da click en el botón “Regresar a pantalla principal”	El usuario es dirigido a la interfaz principal correctamente

4.2 Casos de Prueba

Núm.	Tipo de prueba	Descripción	Duración	Personas requeridas
1	Funcional	Inicio de sesión	3 horas	Equipo de desarrolladores y usuarios finales
2	Funcional	El usuario (manager) podrá crear una nueva tarea, llenando diferentes campos que el ChatBot solicitara	4 horas	Equipo de desarrolladores y usuarios finales
3	Funcional	El usuario (desarrollador) puede crear una nueva tarea, llenando diferentes campos solicitados por el ChatBot	4 horas	Equipo de desarrolladores y usuarios finales
4	Funcional	Listado de tareas dividido entre las tareas completadas y no completadas	2 horas	Equipo de desarrolladores y usuarios finales
5	Funcional	Listado de tareas de todo el equipo, dividido entre las tareas completadas y no completadas	2 horas	Equipo de desarrolladores y usuarios finales
6	Funcional	Modificación de tareas	4 horas	Equipo de desarrolladores y usuarios finales
7	Funcional	Eliminar tarea	2 horas	Equipo de

				desarrolladores y usuarios finales
8	Funcional	Marcar tarea como completada	2 horas	Equipo de desarrolladores y usuarios finales
9	Funcional	Reactivar tarea ya completada, como pendiente	2 horas	Equipo de desarrolladores y usuarios finales
10	Funcional	Filtrar tareas	4 horas	Equipo de desarrolladores y usuarios finales
11	Funcional	Recibir sugerencias/atajos después de cada comando	3 horas	Equipo de desarrolladores y usuarios finales
12	Funcional	Recibir ayuda después de ingresar comandos incorrectos	3 horas	Equipo de desarrolladores y usuarios finales
13	Funcional	Regresar a pantalla principal	2 horas	Equipo de desarrolladores y usuarios finales
14	Integración	La interacción entre todos los componentes del sistema es correcta	5 horas	Equipo de desarrolladores y usuarios finales
15	Rendimiento	El sistema maneja cargas de trabajo para mínimo 60 usuarios	5 horas	Equipo de desarrolladores y usuarios finales

5. **Criterios de Aceptación/Rechazo**

Pruebas exitosas:

- Criterio de aceptación: Todas las pruebas, incluyendo unitarias, de integración, de rendimiento y de seguridad deben pasar sin errores.
- Criterio de rechazo: El fallo en alguna de estas pruebas indicará problemas y se va a requerir revisión.

Participación de usuarios en pruebas:

- Criterio de aceptación: Todos los usuarios identificados como necesarios deben estar presentes y participar en todas las sesiones de prueba. Además, las pruebas en las que participen deben concluirse exitosamente.
- Criterio de rechazo: La ausencia de uno o más usuarios clave durante las sesiones de prueba, o la participación de los usuarios en pruebas que no concluyan exitosamente, se considerará como un motivo de rechazo.

Finalización de pruebas:

- Criterio de aceptación: Todas las pruebas planificadas, incluidas las pruebas unitarias, de integración, de rendimiento, de seguridad y de usuario, deben ser completadas y aprobadas.
- Criterio de rechazo: La no finalización de alguna de las pruebas planificadas o la finalización de pruebas con resultados insatisfactorios indica que el software aún tiene problemas pendientes que deben abordarse antes de su lanzamiento.

Acceso y Autenticación:

- Criterio de aceptación: El sistema debe permitir el acceso solo a usuarios autenticados, mostrando mensajes claros y específicos para errores de autenticación, como "Usuario no encontrado" o "Contraseña incorrecta".
- Criterio de rechazo: Fallas en la autenticación o mensajes de error inexistentes.

Gestión de Tareas:

- Aceptación: Los usuarios deben poder crear, visualizar, editar, y eliminar tareas fácilmente. Al intentar operaciones sobre tareas no existentes, se deben mostrar mensajes explicativos, como "Tarea no encontrada".
- Rechazo: Errores o dificultades en la gestión de tareas, incluyendo la falta de retroalimentación adecuada en situaciones de error.

Flujo de Trabajo de Tareas:

- Aceptación: Marcado correcto de tareas como completadas o pendientes, con la posibilidad de reasignar o cambiar el estado de las tareas eficientemente.
- Rechazo: Incapacidad para cambiar el estado de las tareas o asignar tareas a usuarios específicos correctamente.

Rendimiento:

- Criterio de aceptación: Respuestas del bot en menos de 2 segundos bajo condiciones normales de carga, manteniendo estabilidad hasta 60 usuarios simultáneos.
- Criterio de rechazo: Degradación del rendimiento o tiempos de respuesta mayores a 2 segundos.

Seguridad:

- Criterio de aceptación: Implementación de medidas de seguridad robustas, incluida la encriptación de datos sensibles y la protección contra accesos no autorizados.
- Criterio de rechazo: Vulnerabilidades de seguridad detectadas o fallas en la encriptación de datos.

Usabilidad:

- Criterio de aceptación: Interfaz intuitiva y mensajes de error claros, permitiendo a usuarios sin experiencia previa utilizar el bot eficientemente.
- Criterio de rechazo: Interfaz confusa o mensajes de error que no proporcionan información útil para el usuario.

Estabilidad:

- Criterio de aceptación: Operación continua del bot con una tasa de fallos inferior al 0.1% durante las pruebas.
- Criterio de rechazo: Interrupciones inesperadas del servicio o una tasa de fallos superior al 0.1%.

Compatibilidad:

- Criterio de aceptación: Funcionamiento correcto en las últimas versiones de los sistemas operativos móviles y compatibilidad con versiones actuales de Telegram.
- Criterio de rechazo: Problemas de compatibilidad o fallos en ciertas versiones de sistemas operativos o Telegram.

5.1 Reporte de errores y proceso de corrección

En el caso de encontrar errores o problemas en el código o el funcionamiento del sistemas al realizar las distintas pruebas se deberá generar un informe sobre el mismo que incluya lo siguiente:

- Fecha en la que se registró el error.
- Ambiente y condiciones en las que se presentó el error.
- Una captura de pantalla de lo que el error despliega.
- Una descripción detallada del proceso que se siguió para llegar al error.
- El componente o componentes al que corresponde.
- Cualquier información adicional que sea considerada relevante.

Todos los reportes se deberán almacenar, clasificar según su relevancia y estatus y generar un plan para su resolución asignándoles los recursos que sean necesarios según su nivel de impacto y urgencia. Antes de la finalización del proyecto será necesario asegurarse que el estatus de todos los problemas y errores están como “resueltos”.

5.2 Control de código

El control de código se realizará mediante la herramienta GitHub. Asimismo utilizaremos la metodología de “GitFlow”, una estrategia de branching estructurada que separa claramente el desarrollo de nuevas características, la preparación de lanzamientos y las correcciones urgentes. También se plantean protocolos de seguridad para evitar la corrupción del código y el conflicto entre las distintas versiones en distintas ramas de trabajo. Una de estas será el requerimiento de autorización de dos autores más para la aceptación de procesos “merge”, de tal forma que cualquier cambio siempre sea supervisado por más de un miembro para así minimizar el riesgo de errores y la pérdida de código.

6. Entrenamiento

Para poder poder crear un sistema que cumpla con todos los requerimientos establecidos y un plan de pruebas y aseguramiento de calidad que lo verifique estaremos en un

entrenamiento constante. Este estará dividido en las distintas áreas necesarias para poder implementar el sistema como arquitectura de software, desarrollo web, SQL y bases de datos, aseguramiento de calidad y administración de proyectos. También se tomarán certificaciones impartidas por Oracle en bases de datos, DevOps y otros temas relevantes para el desarrollo del proyecto. Mediante este proceso de entrenamiento continuo podremos verificar los avances que se vayan realizando y asegurarnos que nuestro producto final cumpla con todos los requisitos definidos.

7. Manejo de Riesgos

Tras identificar los posibles riesgos que puede correr el desarrollo del proyecto se les clasificó mediante el uso de una matriz de riesgo para categorizarlos según su nivel de impacto que podrían tener en el desarrollo y la probabilidad de que estos se presenten. De esta forma pudimos identificar cuáles serían las partes más vulnerables de toda la cadena de desarrollo y establecer planes de contingencia dependiendo de qué tan necesarios sean. Gracias a esto logramos que todos los riesgos que logramos identificar estén como mínimo controlados sino resueltos de tal forma que el riesgo de no poder entregar el sistema en tanto tiempo y forma sea mínimo. Y gracias a este proceso también estamos preparados para hacer lo mismo si se presentaran nuevos riesgos.

8. Bitácora de ejecución de pruebas

8.1 Pruebas Manuales

Las pruebas manuales son esenciales para validar escenarios complejos y específicos que no pueden ser fácilmente automatizados. En este proyecto, hemos llevado a cabo una serie de pruebas manuales para verificar la funcionalidad y usabilidad del sistema.

Descripción	Resultado
Inicio de sesión	P
El usuario (manager) podrá crear una nueva tarea, llenando diferentes campos que el ChatBot solicitara	P
El usuario (desarrollador) puede crear una nueva tarea, llenando diferentes campos solicitados por el ChatBot	P
El listado de tareas muestra en la parte superior las tareas sin completar y en la parte inferior se muestran las ya completadas (Desarrollador)	P
Listado de tareas de todo el equipo, muestra en la parte superior las tareas sin completar y en la parte inferior se muestran las ya completadas (Manager)	P
Modificación de tareas	F
Eliminar tarea	P
Marcar tarea como completada	P
Reactivar tarea ya completada, como pendiente	P
Filtrar tareas	F
Recibir sugerencias/atajos después de cada comando	P
Recibir ayuda después de ingresar comandos incorrectos	F
Regresar a pantalla principal	P
La interacción entre todos los componentes del sistema es correcta	P
El sistema maneja cargas de trabajo para mínimo 60 usuarios	P

Enlace: Bitácora Pruebas Manuales

8.2 Pruebas Automatizadas

Las pruebas automatizadas son clave para garantizar la eficiencia y consistencia en la validación de funcionalidades recurrentes y críticas del sistema. En este proyecto, utilizamos Selenium para la automatización de pruebas de interfaz de usuario y otros frameworks adecuados para pruebas de backend.

Nombre de Endpoint Probado	Resultado de Prueba			
http://140.84.190.210/listaTarea	Endpoint pasa la prueba		Pruebas Exitosas	32
http://140.84.190.210/usuarios	Endpoint pasa la prueba		Pruebas Fallidas	0
http://140.84.190.210/usuarioPorId/1	Endpoint pasa la prueba		Pruebas Totales	32
http://140.84.190.210/usuarioPorId/2	Endpoint pasa la prueba			
http://140.84.190.210/usuarioPorId/3	Endpoint pasa la prueba			
http://140.84.190.210/usuarioPorId/4	Endpoint pasa la prueba			
http://140.84.190.210/usuarioPorId/5	Endpoint pasa la prueba			
http://140.84.190.210/usuarioPorId/6	Endpoint pasa la prueba			
http://140.84.190.210/usuarioPorId/7	Endpoint pasa la prueba			
http://140.84.190.210/usuarioPorId/8	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/43	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/41	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/42	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/50	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/51	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/52	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/53	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/17	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/64	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/44	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/45	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/46	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/47	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/48	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/49	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/28	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/55	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/56	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/57	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/81	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/82	Endpoint pasa la prueba			
http://140.84.190.210/listaTarea/101	Endpoint pasa la prueba			

Enlace: Bitácora Pruebas Automatizadas