Dodo Inc.

Plan de Calidad

Duoc Maps

Fecha:04/12/2024

Versión Doc.:3.5

Información de control de documentos

Ajustes	Valor
Título del documento:	Plan de calidad
Título del proyecto:	Duoc Maps
Autor:	Dodo Inc.
Propietario del proyecto (PO):	Valeska Jeria
Gerente de Proyecto (PM):	Ignacio Villarroel
Versión Doc.:	3.5
Sensibilidad:	Limitado
Fecha Creación:	09/04/2024
Fecha Finalización:	16/12/2024

Aprobador(es)y revisor(es) del Documento:

Nombre	Rol	Acción	Fecha Inicial	Fecha Final
Ignacio Villarroel	Gerente	Aprobar y Gestionar	09/04/2024	16/12/2024
Miguel Navarrete	Equipo de Trabajo	Desarrollar y Ejecutar	04/09/2024	16/12/2024
Felipe Cornejo	Equipo de Trabajo	Desarrollar y Ejecutar	04/09/2024	16/12/2024
Sebastián González	Equipo de Trabajo	Desarrollar y Ejecutar	04/09/2024	16/12/2024
Nicolás Zúñiga	Equipo de Trabajo	Desarrollar y Ejecutar	04/09/2024	16/12/2024

Historial del documento:

Revisión	Fecha	Creado por	Breve descripción de los cambios
1.0	04/09/2024	Sebastián González, Nicolás Zúñiga, Miguel Navarrete, Felipe Cornejo	Versión Inicial.
1.5	04/09/2024	Miguel Navarrete, Felipe Cornejo	Se editaron las cosas necesarias para usar, se creó la introducción.
2.0	05/09/2024	Miguel Navarrete, Felipe Cornejo	Se crearon los objetivos de calidad.
2.5	23/09/2024	Sebastián González, Nicolás Zúñiga	Se continuaron los Procesos de Gestión de Calidad.
3.0	12/10/2024	Nicolas Zuñiga, Miguel Navarrete, Felipe Cornejo, Sebastián González	Profundizar en las pruebas a realizar en los sistemas (escritorio y web).
3.5	04/12/2024	Sebastián González	Crear el apartado de bibliografía y citar contenidos.

TABLA DE CONTENIDO

1. INTRODUCCIÓN

El proyecto Duoc Maps tiene el objetivo de proporcionar una solución interactiva para los usuarios de Duoc UC, facilitandoles el acceso a la información importante dentro de la sede San Bernardo. A continuación, Detallaremos los objetivos y las expectativas del proyecto:

- Descripción del Proyecto:

  El proyecto tiene como meta ofrecer un mapa 360 grados de la sede San Bernardo. Este mapa permitirá a los usuarios buscar información sobre profesores, consejeros de carrera, salas y puntos de interés, mostrando imágenes e información detallada sobre cada uno. Además, se integrará un calendario para visualizar los eventos de la sede, que estará disponible en un tótem de forma segura.

- Objetivos del Proyecto:
  - Crear web de "Duoc Maps" el cual va a contener un mapa 360° de la Sede, igualmente va a permitir a los usuarios buscar información sobre los consejeros de carrera, salas, puntos de interés y profesores, incluyendo imagen e información detallada.
  - Integrar un calendario de eventos.
  - Crear una aplicación de administración que permita modificar, agregar, leer y eliminar información relacionada con la Web (Duoc Maps).

- Expectativa del proyecto:
  - Las aplicaciones deben ser funcionales y libres de errores, proporcionando una experiencia precisa y útil para los usuarios.
  - Se da prioridad a la disponibilidad y la calidad del servicio para cumplir con la expectativa del cliente.

2. OBJETIVOS DE LA GESTIÓN DE LA CALIDAD

La gestión de la calidad del proyecto tiene como objetivo garantizar que el proyecto actual cumplirá con los resultados esperados de la manera más eficiente y que los entregables serán aceptados por las partes interesadas relevantes. Implica supervisar todas las actividades necesarias para mantener un nivel deseado de excelencia. Esto incluye la creación e implementación de planes y garantías de calidad, así como el control de calidad y la mejora de la calidad.

Este proyecto seguirá un proceso de gestión de calidad que comprende las actividades relacionadas con la identificación, planificación, ejecución y seguimiento al igual que el control de las actividades relacionadas con la calidad del proyecto.

Los principales objetivos de calidad del proyecto son:

- Satisfacción del usuario.
- Documentación clara y accesible.
- Mantenimiento de la seguridad.
- Monitoreo Continuo.
- Reducción de Errores.
- Cumplimientos de plazos.
- Mejora Continua.
- Los entregables son aceptados por las partes interesadas relevantes según los criterios de calidad/aceptación definidos.

3. Proceso de Gestión de Calidad

El proceso de gestión de la calidad del proyecto comprende todas las actividades (relacionadas tanto con los procesos como con los entregables) que aumentarán la capacidad de cumplir los resultados esperados del proyecto identificados en la Carta del Proyecto.

El proceso de gestión de calidad para este proyecto consta de cinco pasos clave:

- Definir características de calidad (del proyecto);
- Realizar el aseguramiento de la calidad;
- Realizar control de calidad;
- Realizar la aceptación de los entregables; y
- Realizar la Aceptación Final (del Proyecto).

1. Objetivos, Enfoque y Requisitos de Calidad

Objetivos de Calidad:

- **Usabilidad:** Garantizar que la aplicación sea intuitiva y fácil de usar para todos los usuarios (alumnos, profesores, personal administrativo y visitantes).
- **Rendimiento:** Asegurar tiempos de respuesta rápidos para la carga de mapas y resultados de búsqueda.
- **Exactitud:** Asegurar que la información geográfica (ubicaciones, rutas, puntos de interés) esté siempre actualizada y sea precisa.
- **Accesibilidad:** Cumplir con las normativas WCAG 2.1 para accesibilidad web y que los tótems sean accesibles para personas con discapacidad.
- **Fiabilidad:** Asegurar la disponibilidad del sistema incluso bajo alta demanda o en situaciones imprevistas.
- **Seguridad:** Proteger los datos y evitar accesos no autorizados al sistema de administración.

Enfoque de Calidad:

- **Prevención:** Detectar problemas en las primeras fases del proyecto mediante pruebas continuas y retroalimentación temprana.
- **Verificación continua:** Implementar revisiones periódicas del progreso del proyecto a través de auditorías y controles.
- **Cumplimiento de normas:** Asegurarse de que el proyecto cumpla con los estándares de calidad internacionales aplicables a software interactivo.

Requisitos de Calidad:

- Los Mapas interactivos deben cargar en menos de 3 segundos en el 95% de las consultas.
- La información de ubicación debe ser precisa en al menos un 98%.
- La aplicación debe ser intuitiva, con una tasa de error de usuario mínima en las pruebas de usabilidad.
- El sistema debe soportar un mínimo de 100 usuarios simultáneos sin afectar el rendimiento.
- Implementar una autenticación segura para el administrador y un sistema de roles para gestionar accesos.

2. Normas de Calidad, Directrices, Herramientas y Técnicas

Normas de Calidad:

- **ISO/IEC 25010**: Modelo de calidad del software que cubre características como funcionalidad, eficiencia, usabilidad, fiabilidad, y seguridad.
- **PMBOK** (Project Management Body of Knowledge): Se utilizará como referencia para las mejores prácticas de gestión de proyectos y calidad.

Directrices:

- Revisiones de código y pruebas de seguridad deben realizarse cada dos semanas durante el desarrollo.
- Realizar pruebas de rendimiento bajo diferentes condiciones de carga para asegurar que el sistema pueda manejar escenarios de uso realista.

Herramientas y Técnicas:

- **Lighthouse:**
    - Pruebas de Rendimiento.
    - Pruebas de Accesibilidad.
    - Enseña informes sobre mejores prácticas.
    - Pruebas de Optimización para motores de búsqueda.
    - Verifica si la página es progresiva (Funcionalidad offline, instalable en dispositivos móviles).
- **Configuraciones del Quiosco:**
    - Nombre del quiosco
    - Seleccionar sistema con el que trabajará la aplicación (Microsoft Edge)
    - Asignar la url de tu aplicación, para ser representada en el quiosco (Web).
    - Explorador publicó o pantalla interactiva
    - Asignar el tiempo de Reinicio o refresh del quiosco (5 min - 1hr)
- **MSTest:**
    - Pruebas unitarias.
    - Pruebas de integración.
    - Pruebas funcionales.
    - Pruebas de Interfaz de usuario.
    - Pruebas de Regresión.
    - Pruebas de Carga.
    - Pruebas de Rendimiento.
    - Pruebas basadas en Datos.
    - Pruebas de Módulos o componentes.
- **Firebase Firestore:**
    - Administración y gestión de contenido multimedia.
- **Lista de Verificación de Revisión de Calidad:**
    - Revisión de las interfaces de usuario (tótem y web) para verificar que son fáciles de usar.
    - Verificación de tiempos de carga.

    - Evaluación de la precisión de la información de las rutas.
- **Lista de Verificación de Aceptación de Entregables:**
    - Confirmar que el mapa interactivo está funcional y ofrece una representación precisa de la sede.
    - Verificar que la funcionalidad de anuncios esté operativa y permita la actualización de eventos.
    - Asegurar que la aplicación web y la versión para tótem estén sincronizadas y reflejen los mismos datos.

3. Actividades de Aseguramiento de Calidad y Responsabilidades

Reuniones de Revisión de Proyectos:

- **Frecuencia:** Cada dos semanas durante el desarrollo.
- **Participantes:** Equipo de desarrollo y QA (Equipo de Trabajo).
- **Objetivo:** Revisar el progreso del proyecto, identificar posibles áreas de mejora y asegurar el cumplimiento de los estándares de calidad.

Informes Periódicos de Actividades:

- **Contenido:** Estado del proyecto, resultados de pruebas de rendimiento, pruebas de usabilidad, y auditorías de calidad realizadas.
- **Responsable:** Equipo de Trabajo
- **Destinatarios:** Stakeholders, equipo de desarrollo, responsables de QA.

Verificación de Cumplimiento y Auditorías de Contratistas:

- Realizar auditorías periódicas para verificar que las actividades de desarrollo realizadas por contratistas externos cumplan con los estándares de calidad definidos.
- Las auditorías revisarán tanto la codificación como la usabilidad del sistema.

4. Actividades de Control de Calidad para la Mejora Continua

Revisión de Artefactos de Gestión de Proyectos:

- **Documentos clave:** Especificaciones técnicas, manuales de usuario, informes de prueba, diagramas de arquitectura.
- **Frecuencia:** Se revisan cada vez que hay una actualización significativa del sistema o de los requisitos del proyecto.

Revisiones de Planes de Calidad:

- Evaluar el plan de calidad trimestralmente para asegurarse de que se sigue cumpliendo con los estándares y objetivos del proyecto.
- Realizar ajustes al plan de calidad basados en retroalimentación de usuarios o descubrimientos de problemas durante las fases de pruebas.

5. Procedimiento de Configuración Relacionado con los Artefactos y Entregables del Proyecto

Configuración del Sistema:

- **Entregables principales:**
    - Aplicación web y tótem interactivo.
    - Base de datos con información actualizable de salas y puntos de interés.
    - Sistema de gestión de contenidos para anuncios de la institución.
    - Manuales de usuario para administradores y personal de soporte.

Gestión de Versiones:

- **Control de versiones de software**: Cada cambio debe ser documentado y versionado. Se utilizarán herramientas como Git para gestionar el código y las versiones del sistema.
- **Backups periódicos**: Se implementarán políticas de backup automáticos para garantizar que no se pierda información importante en caso de fallos.

Entrega de Artefactos Finales:

- **Revisión final de entregables**: Antes del lanzamiento oficial, se deben revisar y validar todos los entregables, incluyendo la funcionalidad de búsqueda de salas, el mapa interactivo y los anuncios institucionales.
- **Pruebas de aceptación del cliente**: Una vez que el sistema esté completamente desarrollado, los stakeholders y administradores deben realizar una revisión exhaustiva del sistema para garantizar que cumpla con los requisitos iniciales antes de la implementación final.

3.1. Funciones y responsabilidades de la gestión de calidad

La siguiente tabla define las responsabilidades de aquellos involucrados en la gestión de la calidad:

Responsabilidades	Nombres
Plan de Gestión de Calidad	Valeska Jeria, Ignacio Villarroel
Plan de aceptación de entregables	Valeska Jeria, Ignacio Villarroel
Realizar control de calidad	Ignacio Villarroel
Realizar control de calidad	Valeska Jeria
Realizar la aceptación de los entregables	Ignacio Villarroel
Realizar la aceptación final	Valeska Jeria

4. Herramientas a Utilizar

4.1. Lighthouse

Lighthouse es una herramienta automatizada de Google que audita páginas web y proporciona informes detallados en diversas áreas clave. Los tipos de prueba que se pueden realizar con Lighthouse cubren cinco categorías principales:

4.1.1. Prueba de Rendimiento

Evalúa la velocidad de carga y el rendimiento general de la página web. Incluye métricas como:

- Cuánto tarda el navegador en renderizar el primer contenido visual.
- Cuánto tarda en cargar el contenido más grande visible en la ventana.
- Tiempo que tarda la página en ser completamente interactiva.
- Tiempo total en el que la página está bloqueada y no responde a la interacción.
- Medir la estabilidad visual, evaluando si los elementos de la página se desplazan inesperadamente.

4.1.2. Pruebas de Accesibilidad

Revisa qué tan accesible es el contenido de la página para personas con discapacidades, como usuarios que dependen de lectores de pantalla. Entre los aspectos revisados están:

- Textos alternativos en imágenes.
- Contrastes de colores entre texto y fondo.
- Uso adecuado de etiquetas HTML semánticas.
- Navegación mediante teclado.

4.1.3. Mejores prácticas

Evalúa si la página web sigue las mejores prácticas recomendadas para la web moderna. Algunos puntos que revisa:

- Uso de HTTPS en lugar de HTTP.
- No usar bibliotecas vulnerables de JavaScript.
- Correcta implementación de características de seguridad como el uso de certificados válidos.

4.1.4. Prueba de Optimización para motores de búsqueda

Evalúa si la página está optimizada para los motores de búsqueda. Algunos de los factores que revisa son:

- Uso adecuado de metaetiquetas (como el título y la descripción).
- Correcto uso de etiquetas hreflang en sitios multilingües.
- Indexabilidad de la página por parte de los motores de búsqueda.
- Presencia de una estructura HTML válida y rastreable.

4.1.5. Aplicación WEB progresiva

Verifica si la página cumple con los criterios para ser considerada una aplicación web progresiva. Algunas de las características que revisa son:

- Si la página es instalable en dispositivos móviles.
- Funcionalidad offline.
- Uso de Service Workers para mejorar la experiencia de usuario.

4.2. JUnit

- Prueban unidades de código específicas, lo hace de manera individual en la cual las clases se van testeando de una en una.
- De esta manera se pueden encontrar errores de forma temprana en los módulos, así logrando prevenirlos, lo que también nos lleva a que el código no posea regresiones o algo que consiga un efecto negativo como una mala consecuencia.

4.2.1. Pruebas Unitarias

- Prueban unidades de código específicas, como métodos o funciones individuales, asegurando que se comporten como se espera bajo diversas condiciones.
- Estas pruebas suelen enfocarse en métodos individuales de las clases sin interactuar con dependencias externas (como bases de datos o servicios externos).

4.2.2. Pruebas de Integración

- Verifican que varios componentes de la aplicación funcione correctamente en conjunto. Estas pruebas son más amplias que las unitarias y pueden involucrar interacciones entre módulos o clases.
- Se realizan en un entorno que simula el entorno de producción, donde se pueden usar bases de datos, archivos, o APIs reales.

4.2.3. Pruebas Funcionales

- Prueban una funcionalidad completa del sistema, evaluando si una característica específica cumple con los requisitos funcionales.
- Estas pruebas se realizan a mayor nivel que las pruebas unitarias, y pueden involucrar el flujo de trabajo completo de la aplicación.

4.2.4. Pruebas de Regresión

- Verifican que los cambios recientes en el código no hayan roto ninguna funcionalidad existente.
- Estas pruebas se ejecutan cada vez que se realiza una actualización o refactorización del código.

4.2.5. Pruebas de Rendimiento

- Evalúan el rendimiento de una función o parte específica de la aplicación, midiendo el tiempo de ejecución y el uso de recursos.
- Visual Studio ofrece soporte para realizar pruebas de rendimiento, como la medición de tiempo de ejecución o consumo de memoria.

4.2.6. Pruebas de Excepciones

- Aseguran que una parte del código maneje correctamente las excepciones esperadas y lanza las excepciones correctas cuando ocurren condiciones de error.
- Se verifica que las excepciones sean lanzadas adecuadamente y que los bloques try-catch manejan los errores correctamente.

4.2.7. Pruebas Basadas en Datos

- Estas pruebas permiten ejecutar un mismo conjunto de pruebas con múltiples valores de entrada, ideal para validar cómo el sistema responde a diferentes escenarios.
- Se utilizan archivos de datos como JSON, XML o bases de datos para alimentar las pruebas con diferentes entradas.

4.2.8. Pruebas de Módulo o Componente

- Verifican el correcto funcionamiento de módulos o componentes individuales de la aplicación.
- Similar a las pruebas unitarias, pero enfocadas en componentes específicos de la aplicación.

4.3. TestFX

- Se pueden simular interacciones del usuario, como clics de botones, ingreso de texto, desplazamiento y otras acciones, y luego verificar el comportamiento de la interfaz para asegurar que la aplicación funcione como se espera.
- De esta manera se pueden ir automatizando pruebas de interfaz gráfica, como por ejemplo, si funciona o si existe algo realmente dentro de la vista, adicionalmente se puede aplicar en diferentes frameworks, siempre y cuando se enfoque para el apartado gráfico, ya que esta es su única función.

4.4. Pruebas de Gestión de cuentas administrativas (Con BBDD)

Utilizaremos MySQL para el guardado y gestión de las cuentas de los usuarios administradores, para que estos puedan iniciar sesión, cerrar sesión y demás.

4.4.1. Pruebas de Inicio de Sesión (Login)

- **Credenciales válidas:** Verificar que un usuario pueda iniciar sesión con credenciales correctas.
- **Contraseña incorrecta:** Intentar iniciar sesión con un correo válido pero una contraseña incorrecta (Entregar un mensaje por pantalla).
- **Usuario inexistente:** Intentar iniciar sesión con un correo o nombre de usuario que no existe (Entregar un mensaje).
- **Case sensitivity:** Verificar si las credenciales son sensibles a mayúsculas y minúsculas según sea necesario.
- **Tokens de sesión:** Validar que se generen correctamente y tengan la duración esperada.

4.4.2. Pruebas de Cierre de Sesión (Log Out)

- **Cierre de sesión exitoso**: Comprobar que un usuario puede cerrar sesión correctamente.
- **Token inválido:** Realizar operaciones protegidas después de cerrar sesión para validar que el token ya no es válido.
- **Reinicio de sesión:** Busca validar que después de cerrar sesión, el usuario pueda volver a iniciar sesión.

4.4.3. Solicitud de Recuperación de Contraseña

- **Correo válido registrado:** Comprobar que se envía correctamente un enlace o código de recuperación cuando se introduce un correo asociado a una cuenta existente.
- **Correo no registrado:** Verificar que al ingresar un correo que no está registrado, se muestre un mensaje adecuado (por ejemplo: "Si el correo existe, se enviará un enlace").
- **Formato del correo electrónico:** Probar ingresar un correo con formato incorrecto (sin @, sin dominio, etc.) para asegurarte de que el sistema lo valide correctamente.
- **Frecuencia de solicitudes:** Intentar enviar múltiples solicitudes consecutivas y asegúrate de que exista un límite para evitar abusos (por ejemplo, una solicitud cada X minutos).

4.4.4. Enlace o Código de Recuperación

- **Formato del enlace o código:** Asegúrate de que el enlace o código de recuperación generado sea único y seguro.
- **Duración del enlace/código:** Verifica que el enlace o código de recuperación expire después de un tiempo predefinido (por ejemplo, 15 minutos, 1 hora, etc.).
- **Reutilización:** Intenta reutilizar un enlace o código después de que haya sido usado o expirado para asegurarte de que no funcione.
- **Protección de datos:** Comprueba que el enlace no revele información sensible sobre el usuario o el sistema.
- **Redirección segura:** Asegúrate de que el enlace pueda redirigir únicamente a dominios válidos de tu aplicación y no a sitios maliciosos.

4.4.5. Restablecimiento de Contraseña

- **Contraseña válida:** Verifica que el usuario pueda establecer una nueva contraseña cumpliendo con las reglas de seguridad (longitud mínima, caracteres especiales, etc.).
- **Contraseña igual a la anterior:** Intenta restablecer la contraseña con la misma que tenía anteriormente (si esto no está permitido).
- **Confirmación de contraseña:** Validar que el sistema realice la comparación de la nueva contraseña con el campo de confirmación antes de proceder.
- **Longitud y formato:** Probar con contraseñas demasiado largas, vacías o con caracteres no permitidos para verificar la validación.
- **Múltiples intentos:** Intentar restablecer la contraseña varias veces con el mismo enlace/código para asegurarte de que solo se permite una vez.

4.4.6. Pruebas de seguridad

- **Inyección SQL:** Realizar intentos de inyección SQL en los campos de inicio de sesión, registro y recuperación de contraseña.
- **Hashing de contraseñas:** Verificar que las contraseñas no se almacenan en texto plano en la base de datos.
- **Accesos no autorizados:** Intentar acceder a rutas o funciones protegidas sin estar autenticado.
- **Reseteo de contraseñas:** Probar el sistema de recuperación de contraseñas (si lo tienes), verificando enlaces temporales o códigos enviados por correo.

4.4.7. Pruebas de Integridad de Datos

- **Consistencia en la base de datos:** Comprobar que los datos del usuario (contraseña, correo, roles) se almacenen correctamente y no se corrompan.
- **Actualización de información:** Probar cambiar datos del perfil del usuario y verificar que se actualicen correctamente en la base de datos.
- **Eliminación de usuarios:** Probar eliminar usuarios y verificar que las relaciones o datos asociados también se gestionen correctamente.

4.4.8. Pruebas de Concurrencia

- **Registros simultáneos:** Intentar registrar múltiples usuarios al mismo tiempo para comprobar posibles colisiones en la base de datos.
- **Múltiples sesiones:** Verificar que un usuario pueda iniciar sesión en múltiples dispositivos simultáneamente (si está permitido) y que las sesiones funcionen correctamente.

4.4.9. Pruebas de Rendimiento

- **Carga de usuarios:** Simular una gran cantidad de registros, inicios de sesión y consultas simultáneas para evaluar el desempeño de la base de datos (No creo que haya una gran cantidad de usuarios administrativos, así que no habrá problema con esta prueba).
- **Tiempo de respuesta:** Mide el tiempo necesario para realizar operaciones como el registro, inicio de sesión o recuperación de contraseñas.

4.4.10. Pruebas de Usabilidad

- **Mensajes de error:** Asegurarse de que los mensajes de error sean claros y útiles (por ejemplo, "Usuario no encontrado" en lugar de "Error desconocido").
- **Feedback visual:** Comprobar que la aplicación proporcione indicadores claros durante procesos como inicio de sesión o registro.

5. BIBLIOGRAFIA

- Pantaleo, G. (2011). Calidad en el desarrollo de software. McGraw-Hill.
- Mejía Trejo, J. (2024). Principios de aseguramiento de calidad para el diseño de software: Innovación de procesos en las tecnologías de información. Alfaomega.
- Google. (n.d.). Lighthouse: Una herramienta para mejorar la calidad web. Chrome Developers. https://developer.chrome.com/docs/lighthouse/overview?hl=es-419
- OpenJFX. (n.d.). OpenJFX: Modern, fully-featured Java toolkit for developing rich client applications. Gluon. https://openjfx.io/
- Oracle. (n.d.). JavaFX documentation. Oracle. https://www.oracle.com/java/technologies/javase/javafx-docs.html
- TestFX. (n.d.). TestFX: Simple and clean testing for JavaFX. GitHub. https://github.com/TestFX/TestFX
- JUnit. (n.d.). JUnit 5: Framework for Java testing. JUnit. https://junit.org/junit5/
- Google. (n.d.). PageSpeed Insights: Análisis de página 2 para duocmaps. Google. https://pagespeed.web.dev/analysis/http-duocmaps-s3-website-us-east-1-amazonaws-com/fd0py3vgnl?hl=es&form_factor=desktop
- Google. (n.d.).PageSpeed Insights: Análisis de página 3 para duocmaps https://admin-duocmaps-web-app. Recuperado de https://pagespeed.web.dev/analysis/https-admin-duocmaps-web-app/byfr02dct1?hl=es-419&form_factor=desktop