

Ingeniería en Computación

Aseguramiento de la Calidad del Software

I Planificación Proyecto

Profesor:

Marlen Treviño Villalobos

Estudiantes:

Hernandez Ruiz Maikel Alonso

Martinez Molina Justin Alonso

Rodriguez Navarro Jose Alejandro

Rojas Ugalde Fabian Alonso

Sede San Carlos

06/09/2024

[**Introducción 3**](#_grcmt7b9lbqo)

[**Especificaciones del software 4**](#_95v8vyz8gx0h)

[**Descripción de requerimientos o historias de usuario 6**](#_o0d8yf5fu719)

[**Problema 6**](#_i2rfc3x3dv44)

[**Objetivos, metas e indicadores 6**](#_2mkicgd3qavc)

[**Recursos disponibles 6**](#_75z0y6bd0wo2)

[**Cronograma de actividades 6**](#_iqfeob251qye)

[**Bibliografía 6**](#_pxduhxh260nr)

## 

## 

## 

## 

## 

## 

## 

## 

## Introducción

El aseguramiento de la calidad del software es un aspecto fundamental en el ciclo de vida del desarrollo de software, garantizando que las aplicaciones cumplan con los estándares de calidad esperados y funcionen correctamente en entornos reales.

La calidad del software se define como el conjunto de características que determinan su utilidad y existencia, y es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad. Según la norma ISO 9000:2015, la calidad es el grado en el que un conjunto de características inherentes de un objeto cumple con los requisitos establecidos(Martínez,2023). La calidad del software es medible y varía según el tipo de sistema o programa, siendo crucial para garantizar que el software cumpla con las expectativas del usuario y sea confiable y seguro (Carrasco, León, Benavides, 1995).

En el presente documento, se detalla el plan de trabajo para la evaluación de la calidad de un proyecto de software seleccionado de un repositorio público de GitHub.

El proyecto elegido es "DURAR", un sistema ERP/CRM de código abierto, basado en la avanzada pila tecnológica MERN (Node.js, Express.js, MongoDB, React.js), complementado con Ant Design (AntD) y Redux. Este proyecto ha sido seleccionado por su complejidad y relevancia, ya que incluye funcionalidades esenciales como el sistema de login y varios formularios críticos con más de cinco campos, lo que lo convierte en un candidato ideal para aplicar diversas pruebas de calidad.

El propósito de este trabajo es identificar y evaluar la calidad de los módulos clave del sistema, como la gestión de facturas, inventarios, contabilidad y recursos humanos, entre otros. A través de la aplicación de diferentes métodos de pruebas, se busca garantizar que el sistema no solo cumpla con los requisitos funcionales, sino que también mantenga un alto nivel de rendimiento, seguridad y usabilidad.

La metodología a emplear incluye la selección y definición de casos de prueba específicos para cada módulo, la ejecución de pruebas automatizadas y manuales, y la posterior revisión y análisis de los resultados obtenidos. Además, se delinearán las limitaciones del estudio, tales como el alcance de las pruebas y el tiempo disponible para su ejecución, así como los desafíos potenciales en la implementación de las pruebas.

Este proyecto no solo tiene como objetivo encontrar la mejora de la calidad del software evaluado, sino también proporcionar un aprendizaje práctico y profundo en las técnicas de aseguramiento de la calidad, preparando a los participantes para enfrentar desafíos reales en el campo del desarrollo de software.

## Especificaciones del software

El proyecto DURAR es una solución ERP/CRM de código abierto que se basa en una combinación de tecnologías modernas y robustas. A continuación se detallan las especificaciones del software, incluyendo el hardware y las tecnologías utilizadas en el desarrollo de la aplicación:

1. **Lenguajes de Programación y Tecnologías:**
   * **JavaScript**: Utilizado en el desarrollo del frontend y backend de la aplicación. Representa el 95.3% del código total del proyecto.
   * **Pug**: Motor de plantillas HTML que se utiliza en la parte del servidor para generar contenido dinámico. Constituye el 3.5% del código del proyecto.
   * **CSS**: Utilizado para el diseño y estilización del frontend. Contribuye con el 1.1% del código del proyecto.
2. **Frameworks y Librerías:**
   * **Node.js**: Entorno de ejecución para JavaScript en el servidor. Permite la creación de aplicaciones escalables y rápidas.
   * **Express.js**: Framework web minimalista para Node.js, utilizado para manejar las rutas y la lógica del backend.
   * **MongoDB**: Base de datos NoSQL orientada a documentos, empleada para el almacenamiento de datos de la aplicación.
   * **React.js**: Librería de JavaScript para la construcción de interfaces de usuario, utilizada en el desarrollo del frontend.
   * **Ant Design (AntD)**: Sistema de diseño para React, utilizado para crear interfaces de usuario elegantes y consistentes.
   * **Redux**: Librería de gestión de estados para aplicaciones JavaScript, utilizada junto con React.js para manejar los estados de la aplicación de manera centralizada.
3. **Licencia:**
   * **IDURAR License 1.0**: DURAR se distribuye bajo esta licencia de código abierto "fair-code", lo que significa que es libre para su uso y modificación bajo los términos especificados en dicha licencia.

## Descripción de requerimientos o historias de usuario

A continuación se detalla una lista de historias de usuario que serán probadas durante el desarrollo del proyecto.

**Historia de Usuario 1: Visualización de la Lista de Clientes**

* Como usuario de la aplicación quiero ver una lista de clientes con detalles como Tipo, Nombre, País, Teléfono y Correo Electrónico para poder gestionar la información de los clientes de manera eficiente.

**Criterios de Aceptación:**

* La lista debe mostrar las columnas de Tipo, Nombre, País, Teléfono y Correo Electrónico.
* Si no hay clientes registrados, debe aparecer el mensaje "No hay datos".
* Debe haber un campo de búsqueda que permita filtrar los clientes.
* Debe haber un botón para actualizar la lista de clientes.

**Historia de Usuario 2: Agregar un Nuevo Cliente**

* Como usuario de la aplicación quiero poder agregar un nuevo cliente a través de un formulario para mantener actualizada la base de datos de clientes.

**Criterios de Aceptación:**

* El formulario debe incluir los campos necesarios: Tipo, Personas y Empresa.
* El botón de "Enviar" solo debe estar habilitado si todos los campos requeridos están completos.
* Al enviar el formulario, el nuevo cliente debe aparecer en la lista de clientes y el formulario debe cerrarse automáticamente.

**Historia de Usuario 3: Botón "Agregar Nuevo Cliente"**

* Como usuario de la aplicación quiero tener un botón que me permita abrir el formulario para agregar un nuevo cliente para poder añadir clientes de manera rápida y sencilla.

**Criterios de Aceptación:**

* El botón debe estar visible en la pantalla de la lista de clientes.
* Al hacer clic en el botón, debe abrirse el formulario para agregar un nuevo cliente en un modal deslizante desde la derecha.

**Historia de Usuario 4: Buscador de Clientes**

* Como usuario de la aplicación quiero buscar clientes por nombre u otro campo relevante para encontrar rápidamente la información que necesito.

**Criterios de Aceptación:**

* El campo de búsqueda debe permitir la entrada de texto.
* Los resultados de la lista de clientes deben filtrarse en tiempo real según el criterio de búsqueda ingresado.

**Historia de Usuario 5: Crear una Nueva Factura**

* Como usuario del sistema quiero poder crear una nueva factura para registrar una venta a un cliente.

**Criterios de Aceptación:**

* Debe existir un botón o enlace para iniciar el proceso de creación de una nueva factura.
* El formulario de factura debe incluir los campos requeridos para capturar la información de la venta.

**Historia de Usuario 6: Selección de Cliente para Factura**

* Como usuario del sistema necesito seleccionar un cliente existente o buscar uno para asociar el cliente a la factura que estoy creando.

**Criterios de Aceptación:**

* Debe haber un campo de selección o búsqueda para encontrar clientes existentes.
* El cliente seleccionado debe asociarse automáticamente a la factura en creación.

**Historia de Usuario 7: Ingreso de Detalles Básicos de la Factura**

* Como usuario del sistema quiero ingresar los detalles básicos de la factura para completar la información necesaria como número, año, moneda y estado.

**Criterios de Aceptación:**

* El formulario debe permitir la entrada de número de factura, año, moneda, y estado (e.g., borrador, finalizada).
* El sistema debe validar que estos campos están completos antes de avanzar.

**Historia de Usuario 8: Especificar Fechas de la Factura**

* Como usuario del sistema necesito especificar las fechas relevantes de la factura para establecer la fecha de emisión y la fecha de vencimiento.

**Criterios de Aceptación:**

* El formulario de la factura debe incluir campos para la fecha de emisión y la fecha de vencimiento.
* El sistema debe validar que las fechas son correctas y lógicas (e.g., la fecha de vencimiento debe ser posterior a la fecha de emisión).

**Historia de Usuario 9: Agregar Múltiples Artículos a la Factura**

* Como usuario del sistema quiero poder agregar múltiples artículos a la factura para especificar el nombre, descripción, cantidad y precio unitario de cada artículo.

**Criterios de Aceptación:**

* Debe haber un formulario repetible para agregar varios artículos a la factura.
* Cada artículo debe permitir la entrada de nombre, descripción, cantidad, y precio unitario.

**Historia de Usuario 10: Cálculo Automático del Total por Artículo y Subtotal de Factura**

* Como usuario del sistema necesito que el sistema calcule automáticamente el total por cada artículo para obtener el subtotal de la factura.

**Criterios de Aceptación:**

* El sistema debe calcular el total de cada artículo multiplicando la cantidad por el precio unitario.
* El subtotal de la factura debe actualizarse automáticamente al agregar o modificar artículos.

**Historia de Usuario 11: Agregar Campos Adicionales a la Factura**

* Como usuario del sistema quiero tener la opción de agregar campos adicionales a la factura para capturar información extra según sea necesario.

**Criterios de Aceptación:**

* El sistema debe permitir agregar campos adicionales opcionales a la factura.
* Los campos adicionales deben ser configurables y reflejarse correctamente en la factura final.

**Historia de Usuario 12: Cálculo del Total de la Factura Incluyendo Cargos y Descuentos**

* Como usuario del sistema necesito ver el cálculo automático del total de la factura para incluir el subtotal, cargos adicionales y descuentos seleccionables.

**Criterios de Aceptación:**

* El sistema debe calcular el total de la factura, sumando el subtotal y aplicando cargos o descuentos.
* Los cargos y descuentos deben ser configurables y reflejarse correctamente en el total final.

**Historia de Usuario 13: Guardar Factura como Borrador o Finalizar**

* Como usuario del sistema quiero poder guardar la factura como borrado para continuarla después o finalizar directamente.

**Criterios de Aceptación:**

* Debe haber opciones para guardar la factura como borrador o finalizarla.
* Al guardar como borrador, la factura debe poder ser editada más tarde.

### **Historia de Usuario 14: Agregar una Nueva Empresa**

* Como usuario del sistema quiero poder agregar una nueva empresa para expandir la base de datos de empresas con las que interactúa la organización.

**Criterios de Aceptación:**

* Debe existir un botón o enlace claramente visible para iniciar el proceso de agregar una nueva empresa.
* El formulario para agregar una nueva empresa debe incluir campos para el nombre, contacto, país, teléfono, correo electrónico y sitio web.
* Todos los campos, excepto el sitio web, deben ser validados para asegurar que los datos ingresados son correctos (por ejemplo, el formato del correo electrónico).

### **Historia de Usuario 15: Buscar Empresa**

* Como usuario del sistema quiero poder buscar empresas específicas en la lista para facilitar el manejo y acceso a los detalles de las empresas.

**Criterios de Aceptación:**

* Debe existir un campo de búsqueda que permita filtrar empresas por nombre, contacto, país o correo electrónico.
* Los resultados de la búsqueda deben actualizarse en tiempo real a medida que el usuario escribe.
* Si no se encuentran resultados, debe mostrarse un mensaje indicativo como "No hay datos".

### **Historia de Usuario 16: Actualizar Información de Empresa**

* Como usuario del sistema quiero poder actualizar la información de una empresa existente para mantener actualizados los datos de las empresas registradas.

**Criterios de Aceptación:**

* Desde la lista de empresas, debe ser posible seleccionar una empresa y editar su información.
* El formulario de edición debe pre-cargar los datos actuales de la empresa seleccionada.
* Debe haber una opción para guardar cambios que valide los datos antes de actualizarlos en la base de datos.

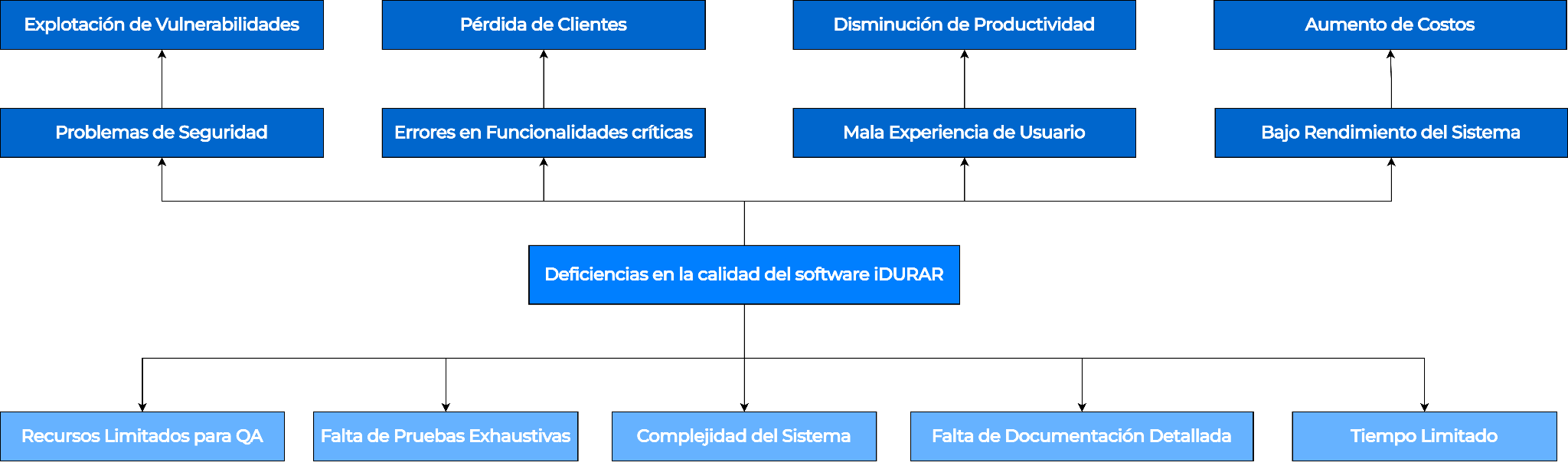
## 

## Problema

El principal problema a resolver en este proyecto es asegurar que todos los componentes críticos del sistema funcionen de manera coherente y eficiente, cumpliendo con los estándares de calidad esperados. Este enfoque busca corregir fallas y prevenirlas mediante un análisis profundo de las causas y sus efectos en la operación del sistema.

Figura 1

*Árbol de Problemas Principal: Evaluación de Calidad del Software DURAR*



Nota. El Árbol de Problemas Principal ilustra las deficiencias en la calidad del software DURAR, destacando cómo las causas subyacentes, como la falta de pruebas adecuadas y la incorrecta gestión de recursos, conducen a fallas funcionales en el sistema. Los efectos de estos problemas impactan en el rendimiento, la usabilidad y la seguridad del software.

Para abordar estos problemas, es importante realizar un buen análisis de las razones que podrían afectar negativamente la calidad del software y su impacto en el funcionamiento general del sistema. Un análisis de los componentes más importantes ayudará a encontrar fallas funcionales y mejorar aspectos como el rendimiento, la usabilidad y la seguridad. Para garantizar que el sistema iDURAR cumpla con las expectativas de sus usuarios y funcione de manera confiable en entornos reales, serán necesarias pruebas adecuadas y una gestión eficiente de recursos.

## Objetivos, metas e indicadores

**Objetivo General:**

* Asegurar la calidad del sistema ERP/CRM DURAR mediante la aplicación de pruebas automatizadas y manuales en los módulos clave, garantizando que el software cumpla con los estándares de rendimiento, seguridad y usabilidad, y funcione correctamente en entornos reales.

**Objetivos Específicos:**

1. **Evaluar** la funcionalidad de los módulos críticos del sistema DURAR (facturación, inventarios, contabilidad, recursos humanos) a través de pruebas automatizadas y manuales, para garantizar que cumplan con los requisitos funcionales y operen de manera eficiente.
2. **Identificar** defectos críticos en los componentes principales del sistema DURAR y **mejorar** el rendimiento y la seguridad del software antes de su implementación final mediante el análisis de los resultados de las pruebas.

**Metas**

**Meta 1:**

Definir y documentar los casos de prueba para cada uno de los módulos clave del sistema DURAR (facturación, inventarios, contabilidad, recursos humanos), asegurando que cubran los principales flujos funcionales y no funcionales del sistema.

**Meta 2:**

* Ejecutar pruebas automatizadas y manuales en los módulos críticos del sistema DURAR, identificando defectos en tiempo real para su posterior documentación y asignación al equipo de desarrollo.

**Meta 3:**

* Analizar los resultados obtenidos en las pruebas para identificar defectos críticos en rendimiento, seguridad y usabilidad, clasificarlos y priorizarlos para su corrección por el equipo de desarrollo.

**Indicadores**

**Indicador 1:** Cobertura de pruebas

Especificación: Porcentaje de módulos clave (facturación, inventarios, contabilidad, recursos humanos) cubiertos por pruebas automatizadas y manuales respecto al total de módulos del sistema.

Cuantitativo: 100% de los módulos clave deben ser evaluados mediante casos de prueba.

Población objetivo: Módulos críticos del sistema DURAR.

Tiempo: Dentro del período de los primeros dos meses de ejecución.

Calidad: Las pruebas deben cubrir los flujos funcionales y no funcionales más importantes.

**Indicador 2:** Tasa de defectos críticos corregidos

Especificación: Proporción de defectos críticos identificados en las pruebas y corregidos en el sistema, respecto al total de defectos críticos encontrados.

Cuantitativo: Se debe corregir al menos el 90% de los defectos críticos.

Población objetivo: Defectos críticos encontrados en los módulos clave del sistema DURAR.

Tiempo: Corrección dentro del plazo de los tres meses de ejecución del proyecto.

Calidad: Las correcciones deben ser validadas mediante pruebas de regresión para asegurar que no introducen nuevos defectos.

**Indicador 3:** Índice de rendimiento del sistema

Especificación: Medición del tiempo de respuesta del sistema DURAR bajo diferentes escenarios de carga, antes y después de aplicar mejoras.

Cuantitativo: Se espera una mejora del 20% en el rendimiento (reducción del tiempo de respuesta bajo carga).

Población objetivo: Módulos clave del sistema DURAR.

Tiempo: Medición final al concluir el tercer mes del proyecto.

Calidad: El sistema debe mantener un rendimiento consistente bajo las condiciones de carga esperadas en producción.

## 

## Recursos disponibles

**Personas / Equipo del Proyecto**

Maikel Alonso Hernández Ruiz

Justin Alonso Martínez Molina

José Alejandro Rodríguez Navarro

Fabián Alonso Rojas Ugalde

**Equipos y Materiales.**

Para el desarrollo y ejecución de las pruebas, se dispone del siguiente hardware y software:

Conexión a internet de alta velocidad para la gestión remota del repositorio en GitHub.

Herramientas de automatización de pruebas como Selenium y Jest, las cuales no requieren de licencias adicionales para su uso en proyectos educativos.

**Software y Licencias.**

El proyecto DURAR es de código abierto y se distribuye bajo la licencia IDURAR License 1.0. Por tanto, no es necesario adquirir licencias comerciales para el uso o modificación del software. Además, se utilizarán herramientas de código abierto como Node.js, React.js, MongoDB, Ant Design y Redux, todas disponibles gratuitamente bajo licencias abiertas.

**Recursos Financieros.**

No se requieren grandes inversiones financieras para la ejecución de este proyecto, ya que los recursos de hardware y software están disponibles a través de las licencias de código abierto. Los costos asociados son mínimos y se limitan a la disponibilidad de hardware personal y conexión a internet.

## 

## Cronograma de actividades

| **Cronograma** | **Semanas** | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Actividades** | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| Identificación del Problema |  |  |  |  |  |  |  |  |  |  |
| Definición de Objetivos |  |  |  |  |  |  |  |  |  |  |
| Selección de Tipo de Pruebas |  |  |  |  |  |  |  |  |  |  |
| Definición de Casos de Prueba |  |  |  |  |  |  |  |  |  |  |
| Desarrollo del Plan de Pruebas |  |  |  |  |  |  |  |  |  |  |
| Documentación de Defectos |  |  |  |  |  |  |  |  |  |  |
| Automatización de Pruebas |  |  |  |  |  |  |  |  |  |  |
| Informe de Pruebas |  |  |  |  |  |  |  |  |  |  |
| Presentación de las pruebas |  |  |  |  |  |  |  |  |  |  |

## 

## 

## Bibliografía

Fernández Carrasco, Oscar M, García León, Delba, & Beltrán Benavides, Alfa. (1995). Un enfoque actual sobre la calidad del software. *ACIMED*, *3*(3), 40-42. Recuperado en 06 de septiembre de 2024, de <http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94351995000300005&lng=es&tlng=es>

Martínez, G. A. (2023, 30 mayo). *La calidad en el software: Definiciones*. Brave Developer. <https://bravedeveloper.com/2023/05/30/la-calidad-en-el-software-definiciones/>