Estrategia de Pruebas

# Aplicación Bajo Pruebas

## Nombre Aplicación

Ghost

## Versión

3.42.9

## Descripción

Sistema de gestión de contenido (CMS) de código abierto diseñado específicamente para blogs y publicaciones online, siendo una plataforma para escritores y editores que quieren centrarse en la creación de contenido. Sistema que se tomará para realizar pruebas de automatización de acuerdo con una estrategia alineada a un presupuesto.

## Funcionalidades Core

* Editor de contenido
* Gestión de contenido
* Tema personalizable
* Optimización para SEO (optimización de lenguajes de contenido)
* Paginas Adaptables
* Gestión de usuarios
* Integraciones con otros sistemas
* Seguridad
* Rendimiento
* Soporte Técnico

## Diagrama de Arquitectura

[architecture-diagram.png](https://1drv.ms/i/s!AjoHFEIAVlsbjsUmKWOtFfexKZ-B6g?e=h6rMlS)

## Diagrama de Contexto

[context-diagram.png](https://1drv.ms/i/s!AjoHFEIAVlsbjsUlJZ6jexoYJySVWA?e=7H8UNX)

## Modelo de Datos

[class-diagram.png](https://1drv.ms/i/s!AjoHFEIAVlsbjsUoBInczF-KXDsBVg?e=WV0Egi)

## Modelo de GUI

[GUI diagram.png](https://1drv.ms/i/s!AjoHFEIAVlsbjsUnpqMaVFJKirOSuw?e=QIsVCP)

# Contexto de la estrategia de pruebas

## Objetivos

Durante el periodo de pruebas, se realizará una evaluación integral, priorizando los aspectos funcionales tomando como enfoque principal de este sprint la validación del correcto funcionamiento de los módulos de creación, edición y publicación de contenido en el sitio web, así como en asegurar la operatividad y seguridad de las capacidades de gestión de usuarios, incluyendo la creación de cuentas y asignación de roles.

En este sprint no se harán pruebas exploratorias, ya que se han realizado previamente, lo que permitirá centrar el esfuerzo en desarrollar pruebas unitarias y de integración, elevar la cobertura de código a un 80 % y garantizar un producto de la más alta calidad.

Adicionalmente, mediante pruebas de usabilidad, aplicadas aprovechando las facilidades ofrecidas por AWS, se buscará garantizar un diseño intuitivo y de fácil interacción en la interfaz de usuario implementada para los administradores y editores. Asimismo, se realizará una evaluación en el rendimiento del sitio web, asegurando respuestas rápidas y fluidas, mediante pruebas de velocidad de carga en las de páginas, y se verificará su escalabilidad para manejar grandes volúmenes de contenido y tráfico, sin evidenciar degradación en el del rendimiento.

Teniendo en cuenta lo anteriormente mencionado estos serán los aspectos clave durante este sprint de pruebas:

1. **Validación de Funcionalidades:**
   1. Enfoque: Funcional.
   2. Objetivo: Garantizar que las funcionalidades de creación, edición y publicación de contenido operen correctamente en el sitio web.
   3. Métricas de éxito: Todos los casos de prueba de funcionalidades principales han sido ejecutados y aprobados sin errores significativos.
2. **Gestión de Usuarios:**
   1. Enfoque: Funcional y Seguridad.
   2. Objetivo: Asegurar que, tanto la capacidad de gestión de usuarios, como la creación de cuentas y la asignación de roles, estén operativas y seguras.
   3. Métricas de éxito: Todos los casos de prueba relacionados con la gestión de usuarios han sido ejecutados y aprobados sin errores significativos. No se han identificado vulnerabilidades de seguridad durante las pruebas.
3. **Pruebas de Rendimiento:**
   1. Enfoque: No Funcional (Rendimiento).
   2. Objetivo: Evaluar la velocidad de carga de las páginas del sitio web para garantizar respuestas rápidas, y fluidez en la interacción.
   3. Métricas de éxito: Se han realizado pruebas de rendimiento y se han cumplido los objetivos de tiempo de carga establecidos. No se han identificado cuellos de botella que afecten significativamente el rendimiento del sitio.
4. **Comprobación de Escalabilidad:**
   1. Enfoque: No Funcional (Escalabilidad).
   2. Objetivo: Verificar la capacidad del sitio web para manejar grandes cantidades de contenido y tráfico sin presentar degradación en el rendimiento.
   3. Métricas de éxito: Se han llevado a cabo pruebas de estrés y el sitio web ha demostrado ser tener la capacidad de manejar volúmenes de contenido y tráfico significativos, sin mostrar signos de degradación en el del rendimiento del aplicativo. No se han identificado problemas de escalabilidad durante las pruebas.

## Duración de la iteración de pruebas

8 semanas ó 8 sprints de una semana

## Presupuesto de pruebas

### Recursos Humanos

Para este proyecto se recomienda el perfil de un ingeniero automatizador de pruebas senior, el cual cuente con mínimo tres (3) años de experiencia, desempeñando sus labores con las siguientes herramientas: Jenkins, Bitbucket, Cypress, y Docker. Debido al tamaño y la complejidad del proyecto se deberán contratar mínimo tres personas con este perfil, los ingenieros por contratar deberán interactuar con el proyecto durante 8 horas semanales.

### Recursos Computacionales

Para llevar a cabo este plan de pruebas automatizadas, se requerirían mínimo los siguientes recursos computacionales:

Un computador portatil para cada ingeniero automatizador. Las especificaciones sugeridas para el equipo son: Laptop ASUS ROG Strix G15, con procesador Intel Core i7 de 10ª generación, 16 GB de RAM y almacenamiento SSD de 1 TB o un similar.

Se requerirán dispositivos móviles para realizar pruebas de integración en diferentes dispositivos y plataformas. Se sugiere contar con al menos un dispositivo Android y un dispositivo iOS cualquiera (teléfono o tableta) capaz de actualizar a la última versión del sistema operativo del fabricante.

Un dispositivo de grabación de pantalla para cada equipo portátil, para registrar las pruebas de interfaz de usuario y revisarlas posteriormente. Se sugiere la Grabadora de pantalla Elgato Game Capture HD60 S.

### Recursos Económicos para la contratación de servicios/personal

De acuerdo con la estrategia que se quiere implementar para el proceso de pruebas, es necesaria la contratación de mínimo tres ingenieros automatizadores junior, que interactúen con el proyecto 40 horas por sprint, y un automatizador senior que disponga un mínimo de 20 horas por sprint para el proyecto y que además sea capaz de implementar liderazgo y enseñanza a los demás y así lograr efectivamente agilidad y efectividad a las pruebas que se desean realizar.

Recurso económico personal:

1. **Senior tester automatizador: $17.000 (peso colombiano) / hora** (<https://co.talent.com/salary?job=tester+automatizador+senior>)

Tras realizarse un sondeo de mercado se llega a la conclusión que se puede contratar un automatizador tester senior ofreciéndole un salario de 17.000/hora, lo cual es un salario mayor al promedio manejado en el mercado (Talent.com, s/f).

De acuerdo con la necesidad se ha analizado lo siguiente:

Al requerir un automatizador Senior, se estima 8 horas/persona por cada semana. De acuerdo con el valor redondeado de $17.000 pesos colombianos es decir que, al ser cuatro seniors, serían $3.264.000 pesos por las 8 semanas.

De acuerdo con lo anterior, por sprint tenemos el resultante para utilizar de $419.936 pesos en recursos de máquina. Es decir, realizando la conversión ya que maquinas en nube se maneja con la moneda del dólar, se tendría 107.15 USD, lo cual es suficiente por Sprint para utilizar maquina en nube ya que la mayoría de las pruebas también se pueden sobrellevar en recursos locales como portátiles o computadores de mesa de la empresa.

Se pretende usar 200 horas del servicio de AWS CodeBuild con una configuración estimada general1.medium con 7 GB de memoria, para un tiempo de compilación de un proyecto de tamaño moderado, tendría un costo de 0,01 USD que se cobra por cada minuto de compilación, por lo que 200 horas máquina con este servicio costarían 120 USD.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Recurso** | **Cantidad** | **Horas** | **Valor/Hora** | **Total 8 Semanas** |
| Automatizador Senior | 3 | 64 | $ 17,000.00 | $ 3,264,000.00 |
| AWS Code Build | 1 | 200 | $ 2,400.00 | $ 3,840,000.00 |
| Recursos de maquina en la nube | 1 | 8 | $ 52,492.00 | $ 3,359,488.00 |
|  |  |  |  | $ 10,463,488.00 |

*Tabla de presupuesto. Autoría Propia*.

## TNT (Técnicas, Niveles y Tipos) de pruebas

De acuerdo con los objetivos planteados se ha evaluado la relación correspondiente que existe para cada objetivo el aplicar TNT. En la siguiente tabla podemos demostrar a través de los objetivos, el nivel, el tipo y la técnica que se ha escogido.

|  |  |  |  |
| --- | --- | --- | --- |
| Nivel | Tipo | Técnica | Objetivo |
| Pruebas exploratorias | **Pruebas funcionales** | **Pruebas manuales** | Realizar una revisión general de las funcionalidades del sitio, para familiarizarse con él y encontrar fallos detectables a primera vista. |
| Pruebas unitarias | **Pruebas Funcionales** | **Apis de automatización** | **x** Garantizar que las funcionalidades de creación, edición y publicación de contenido funcionen correctamente en el sitio web. |
| Pruebas integración | **Pruebas Funcionales** | **Pruebas con Ripping** | **x** Hay que asegurar que las capacidades de gestión de usuarios, como la creación de cuentas y la asignación de roles, estén operativas y seguras. |
| Pruebas de Sistema | **Pruebas No Funcionales** | **Pruebas con Monkey testing** | Evaluar la velocidad de carga de páginas del sitio web para garantizar respuestas rápidas y fluidez. |
| Pruebas de integración | **Pruebas No funcionales** | **Pruebas E2E con Cypress** | Verificar la capacidad del sitio web para manejar grandes cantidades de contenido y tráfico sin degradación del rendimiento. |
| Pruebas de regresión | **Pruebas Funcionales** | **Pruebas VRT con ResembleJS** | Validar que durante el proceso de desarrollo no se generen efectos colaterales en la GUI como resultado de cambios que se realicen el código |

## Distribución de Esfuerzo

### Para realizar una distribución de esfuerzo legible, y gran eficacia se manejará la estrategia de pirámide y lograr llevar un orden.

### Semana 1

El enfoque principal de esta semana se dirigirá al desarrollo de pruebas exploratorias manuales, para complementar el desarrollo de la primera semana de prueba de los desarrolladores. Así mismo identificar las pruebas funcionales y no funcionales que pueden presentarse en el CMS para contemplar un pool de escenarios de prueba.

* Pruebas exploratorias de GUI y creación de escenarios:
  + Realización de pruebas manuales y exploratorias del CMS GHOST para identificar posibles escenarios de prueba.
  + Enfoque en descubrir fallos no anticipados en la interfaz de usuario.
  + Preparación de conjunto inicial de casos de prueba.

### Semana 2

Pruebas de unidad automatizadas:

* Desarrollo de pruebas unitarias automatizadas para garantizar que las funciones individuales del CMS funcionen correctamente.
* Hay que asegurar que las funcionalidades funcionen correctamente en el CMS.
* Diseño y desarrollo de nuevos casos de pruebas basados en las funcionalidades identificadas.

### Semana 3

Pruebas de unidad automatizadas (2 días y medio):

* Continuación de las pruebas unitarias automatizadas para completar la cobertura de todas las funcionalidades críticas.
* Enfoque en probar componentes aislados para asegurar su correcto funcionamiento.

Pruebas automatizadas de componentes (2 días y medio):

* Pruebas de componentes individuales del CMS verificando su integración con otros componentes.

### Semana 4

Componentes. Pruebas automatizadas de componentes (2 días, total 4 días y medio):

* Finalización de las pruebas de componentes, asegurando que cada parte del sistema funcione correctamente cuando se combine con otros componentes.

Pruebas automatizadas de integración (3 días):

* Verificación de la interacción entre diferentes componentes del sistema, utilizando las técnicas correspondientes.

### Semana 5

Correspondientes. Pruebas automatizadas de integración (3 días, en total 6 días):

* Continuación de las pruebas de integración para garantice que todos los componentes funcionen juntos sin problemas.
* Uso de pruebas e2e con Cypress para verificar la capacidad del sitio web para manejar diferentes escenarios con grandes cantidades de contenido y trafico sin degradación del rendimiento.
* Creación de casos de prueba para pruebas automatizadas por api, asegurando una cobertura completa de las funcionalidades críticas.

### Semana 6

Pruebas e2e (5 días):

* Realización de pruebas e2e para validar los flujos completos del usuario en el CMS con los escenarios contemplados y creados con todo el equipo.
* Asegurarse de que el sistema funcione de extremo a extremo sin fallos.
* Realizar reporte de fallos a través de Issues con un formato planteado para identificar específicamente el bug.
* Utilizar cypress para ejecutar pruebas que simulen el comportamiento del usuario final.

### Semana 7

Inicio pruebas de regresión VRT (5 días):

* + Ejecución de pruebas de regresión visual (VRT) con ResembleJS para asegurar que los cambios en el código no afecten negativamente la GUI.
  + Validación de que no se generen efectos negativos en la interfaz de usuario debido a cambios en el CMS por versionamiento.

### Semana 8

Refinamiento de pruebas y documentación (5 días):

* + Refinamiento de las pruebas realizadas en las semanas anteriores.
  + Elaboración de un documento recopilatorio de las pruebas que sirva como base para futuro desarrollos.

# Referencias

1. Salario de un Automatizador QA Senior - "Talent.com. (s/f). Información sobre el salario promedio para Tester Automatizador Senior en Colombia. Recuperado de <https://co.talent.com/salary?job=tester+automatizador+senior>."
2. Precios de AWS CodeBuild. Recuperado de <https://aws.amazon.com/es/codebuild/pricing> el 21 de abril de 2024.