Estrategia de Pruebas

1. Aplicación Bajo Pruebas

1.1. Nombre Aplicación: Ghost
1.2. Versión: 3.41.1
1.3. Versión VRT: 4.44
1.4. Descripción:

Ghost es una plataforma de blogs *open-source*, escrita en *JavaScript* y distribuida bajo la Licencia MIT. La aplicación está diseñada simplificar el proceso de construir un sitio web, publicar contenido, enviar boletines y ofrecer suscripciones pagadas a los miembros.

1.5. Funcionalidades Core:

- Crear, editar y eliminar *posts*
- Crear, editar y eliminar borradores
- Crear, editar y eliminar páginas
- Crear, editar y eliminar tags
- Crear, editar y eliminar miembros de la página
- Crear, editar y eliminar staff de la página
- Editar el diseño del sitio web
- Administrar el pago de suscripciones de los miembros de la página
- Administrar las integraciones con aplicaciones de terceros
- Administrar los boletines que se envían a los miembros de la página

1.6. Diagrama de Arquitectura:

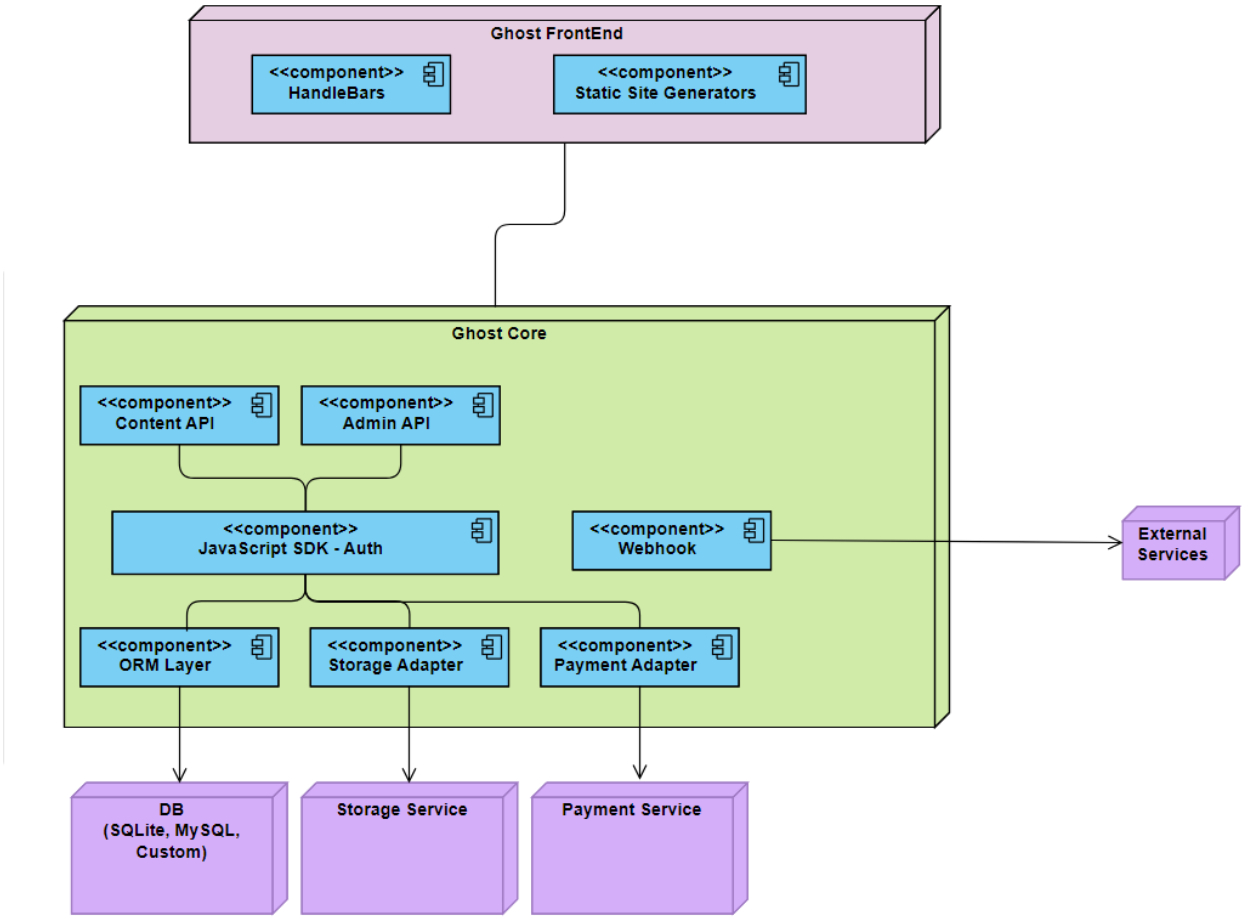

Imagen: diagrama de arquitectura para aplicación GHOST.

1.7. Diagrama de Contexto:

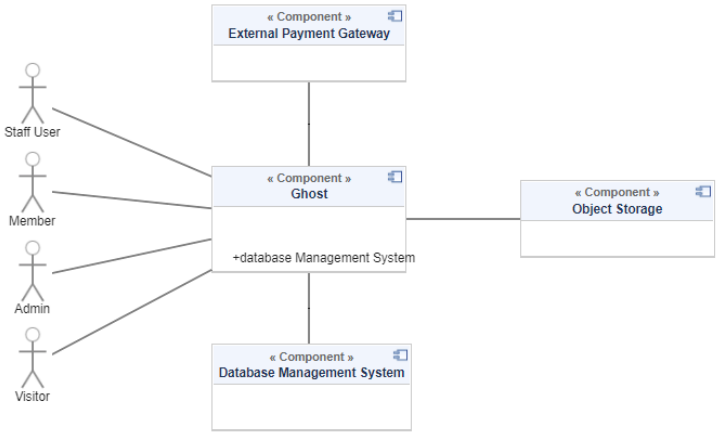

Imagen: diagrama de contexto para aplicación GHOST.

1.8. Modelo de Datos:

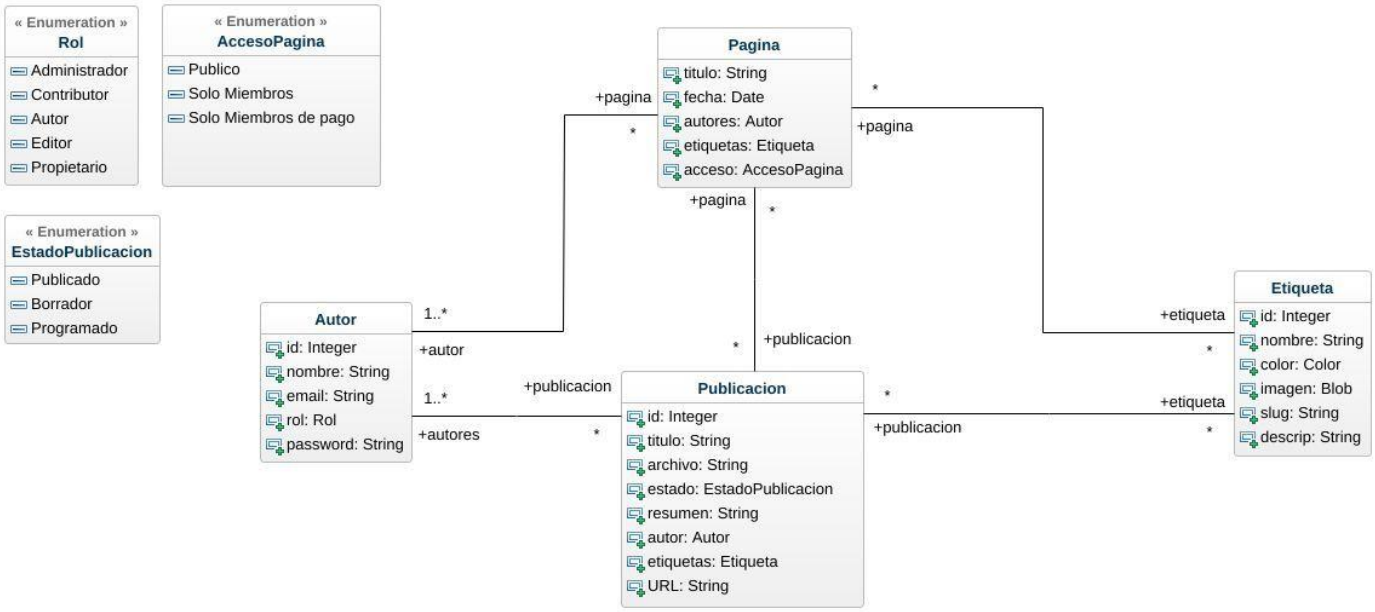

Imagen: diagrama de clases aplicación GHOST.

1.9. Modelo de GUI:

El modelo GUI se encuentra disponible en el siguiente enlace.

2. Contexto de la estrategia de pruebas

2.1. Objetivos:

- Identificar errores en la funcionalidad de creación y edición de posts, asegurando la correcta funcionalidad y validación de datos.
- Identificar errores en la funcionalidad de creación, edición y eliminación de miembros, garantizando la integridad y precisión de los datos relacionados.
- Identificar errores en la funcionalidad de creación y edición de tags, verificando la consistencia y adecuada gestión de etiquetas.
- Identificar errores en la funcionalidad de creación y edición de pages, asegurando la correcta visualización y edición de páginas.
- Identificar errores en la funcionalidad de Inyección de código, evitando vulnerabilidades y asegurando la seguridad del sistema.
- Identificar errores en la sección del portal de suscriptores, asegurando el correcto funcionamiento y acceso de los suscriptores.
- Identificar errores en la funcionalidad de cambio de contraseña para un usuario autenticado en la aplicación, verificando que los cambios se realicen de manera segura y sin problemas.
- Identificar errores utilizando generación de datos sobre la creación y edición de posts, validando la correcta manipulación y procesamiento de datos generados automáticamente.
- Identificar errores utilizando generación de datos sobre la creación y edición de tags, verificando el manejo adecuado de datos generados de forma automática.
- Identificar errores utilizando generación de datos sobre el cambio de contraseña para un usuario autenticado en la aplicación, asegurando que los cambios se realicen correctamente y sin fallos.
- Identificar cambios visuales que puedan surgir en la transición de la versión 3.41.1 a 4.44 para las funcionalidades de tags, posts y cambio de contraseña del usuario, verificando la consistencia y apariencia adecuada de la interfaz de usuario.
- Implementar pruebas unitarias para la creación, edición y eliminación de posts que cubran al menos el 15% del código correspondiente, con el fin de mejorar la calidad y la fiabilidad del software.

2.2. Duración de la iteración de pruebas:

Se planea utilizar 8 semanas de lunes a viernes, empezando el 3 de abril de 2023 y finalizando el 26 de mayo del 2023.

2.3. Presupuesto de pruebas:

2.3.1. Recursos Humanos

Para esta estrategia de pruebas se contará con la disponibilidad de cuatro testers seniors, con una dedicación de 8 horas semanales por persona. Cada uno de los testers, tiene conocimientos en pruebas exploratorias, pruebas automatizadas con monkeys, rippers, APIs de automatización y de regresión visual. Además, son expertos en herramientas como Cypress, Kraken y Resemble.js, y cuentan con un conocimiento básico sobre Playwright, Puppeter y Backstop.

2.3.2. Recursos Computacionales

Cada ingeniero cuenta con una máquina personal para la realización de pruebas automatizadas y pueden ser ocupadas durante las horas de descanso de cada uno de ellos.

Además, el presupuesto destinado a la infraestructura en la nube será de **400 USD** y cubrirá los costos asociados con la implementación y configuración de los recursos necesarios en AWS, como servidores virtuales (instancias EC2), almacenamiento (EBS), redes y otros servicios relacionados.

2.4. TNT (Técnicas, Niveles y Tipos) de pruebas:

Nivel	Tipo	Técnica	Objetivo
Sistema	Funcional	Pruebas exploratorias	Identificar errores en la funcionalidad de creación y edición de posts.
Sistema	Funcional	Pruebas exploratorias	Identificar errores en la funcionalidad de creación, edición y eliminación de miembros.
Sistema	Funcional	Pruebas exploratorias	Identificar errores en la funcionalidad de creación y edición de tags.
Sistema	Funcional	Pruebas exploratorias	Identificar errores en la funcionalidad de creación y edición de pages.
Sistema	Funcional	Pruebas exploratorias	Identificar errores en la funcionalidad de Inyección de código
Sistema	Funcional	Pruebas exploratorias	Identificar errores en la personalización del portal de suscriptores
Integración	Negativas	Pruebas de reconocimiento: Monkeys y Rippers	Identificar errores en la funcionalidad de creación y edición de posts.
Integración	Negativas	Pruebas de reconocimiento: Monkeys y Rippers	Identificar errores en la funcionalidad de creación y edición de tags.
Integración	Negativas	Pruebas de reconocimiento: Monkeys y Rippers	Identificar errores en la funcionalidad de cambio de contraseña para un usuario autenticado en la aplicación.
Sistema	Funcional	API de automatización: Cypress y Kraken	Identificar errores en la funcionalidad de creación y edición de posts.
Sistema	Funcional	API de automatización: Cypress y Kraken	Identificar errores en la funcionalidad de creación y edición de tags.
Sistema	Funcional	API de automatización: Cypress y Kraken	Identificar errores en la funcionalidad de cambio de

			contraseña para un usuario autenticado en la aplicación.
Sistema	Negativa	API de automatización: Cypress y Kraken	Identificar errores en la funcionalidad utilizando generación de datos sobre la creación y edición de posts.
Sistema	Negativa	API de automatización: Cypress y Kraken	Identificar errores en la funcionalidad utilizando generación de datos sobre la creación y edición de tags.
Sistema	Negativa	API de automatización: Cypress y Kraken	Identificar errores en la funcionalidad utilizando generación de datos sobre el cambio de contraseña para un usuario autenticado en la aplicación.
Sistema	Caja negra	Resemble JS	Identificar cambios visuales que puedan generar en el cambio de versión de 3.41.1 a 4.44 para las funcionalidades de tags, posts y cambio de contraseña para el usuario.
Unidad	Funcional	API de automatización: Jasmine y Karma	Implementar pruebas unitarias para la creación, edición y eliminación de posts que cubran al menos el 15% del código correspondiente, con el fin de mejorar la calidad y la fiabilidad del software.

2.5. Distribución de Esfuerzo

A partir de los recursos previstos para la estrategia de pruebas, es evidente que hay una mayor disponibilidad de recursos humanos en contraste a los recursos computacionales. Sacando provecho de esta situación se hará uso de la distribución de pruebas basada en cono, donde se crearán y ejecutarán pruebas manuales y automatizadas sobre la interfaz gráfica de la plataforma Ghost.

Al contar con cuatro ingenieros senior, se espera tener una buena base de pruebas para validar la calidad de la plataforma. La descripción del esfuerzo a lo largo de las 8 semanas se puede ver a continuación.

	Ingeniero 1, 2, 3 y 4
Semana 1	**8 horas cada ingeniero** • Creación del ambiente para el despliegue local de la plataforma Ghost. • Pruebas exploratorias sobre cada una de las funcionalidades. Cada ingeniero será responsable de al menos una funcionalidad
Semana 2	**8 horas cada ingeniero** • Continuación de pruebas exploratorias • Creación y ejecución de monkeys sobre cada una de las funcionalidades. Los ingenieros continuaran sus pruebas sobre las funcionalidades de posts, tags y autenticación. • **En esta semana se sacará provecho a las máquinas personales para la ejecución de pruebas de reconocimiento**
Semana 3	**8 horas cada ingeniero** • Creación y ejecución de rippers sobre cada una de las funcionalidades. Los ingenieros continuaran sus pruebas sobre las funcionalidades con las que iniciaron en la semana 2. • **En esta semana se sacará provecho a las máquinas personales para la ejecución de pruebas de reconocimiento**
Semana 4	**8 horas cada ingeniero** • Creación y ejecución de pruebas de extremo a extremo a través de las aplicaciones Cypress y Kraken. Cada ingeniero será encargado de crear las pruebas tanto en Cypress como en Kraken de la funcionalidad que le sea asignada. • Implementación y configuración del entorno de prueba en AWS: $50 USD. • Ejecución de pruebas e2e iniciales para identificar errores principales: $50 USD.
Semana 5	**8 horas cada ingeniero** • Creación de pruebas de regresión visual sobre la versión 4.44. Los ingenieros podrán utilizar las pruebas creadas en las semanas anteriores sobre las APIs de automatización para generar la comparación de imágenes. • Ejecución de pruebas e2e más exhaustivas para abarcar todas las funcionalidades principales: $80 USD. • Análisis y solución de problemas encontrados durante las pruebas: $20 USD.

Semana 6	**8 horas cada ingeniero** • Adición de pruebas de extremo a extremo para asegurar que las funcionalidades de creación y edición de posts, tags y manejo de contraseñas cumplan con las validaciones necesarias sobre datos correctos e incorrectos. • Ejecución de pruebas e2e adicionales para validar correcciones de errores: $70 USD. • Optimización y refinamiento de los scripts de prueba: $30 USD.
Semana 7	**8 horas cada ingeniero** • Basados en los resultados de pruebas automatizadas, se realizarán nuevas pruebas de exploración con el fin de detectar posibles errores que no se habían detectado inicialmente
Semana 8	**8 horas cada ingeniero** • Desarrollo de pruebas unitarias sobre el módulo de posts para su creación y eliminación • Ejecución de pruebas e2e finales y generación de informes de resultados: $60 USD. • Reserva de fondos adicionales para posibles pruebas de regresión o casos especiales no contemplados: $40 USD.
Total Horas	**64 horas cada ingeniero**

Tabla: distribución de tiempo para las actividades del Ingenieros Automatizadores.

Los recursos computacionales se utilizarán de la siguiente manera:

	AWS EC2-1	AWS EC2-2	AWS EC2-3	AWS EC2-4
Semana 4	**25 horas**	**25 horas**	**25 horas**	**25 horas**
	Ejecución Pruebas de unidad automatizadas	Ejecución Pruebas automatizadas de componentes	Monkey Test empleando librería Cypress	Pruebas Exploratoria sistemáticas empleando librería Rippers
Semana 5	**25 horas**	**25 horas**	**25 horas**	**25 horas**
	Ejecución Pruebas de unidad automatizadas	Pruebas automatizadas de integración	Monkey Test empleando librería Cypress	Pruebas Exploratoria sistemáticas empleando librería Rippers

Semana 6	**25 horas**	**25 horas**	**25 horas**	**25 horas**
	Ejecución Pruebas de unidad automatizadas	Pruebas automatizadas empleando APIs de automatización	Monkey Test empleando librería Cypress	Pruebas Exploratoria sistemáticas empleando librería Rippers
Semana 8	**25 horas**	**25 horas**	**25 horas**	**25 horas**
	Ejecución Pruebas de unidad automatizadas	Ejecución Pruebas de automatización E2E.	Monkey Test empleando librería Cypress	Pruebas Exploratoria sistemáticas empleando librería Rippers
Total horas	**100 horas**	**100 horas**	**100 horas**	**100 horas**