Estrategia de Pruebas

1. Aplicación Bajo Pruebas

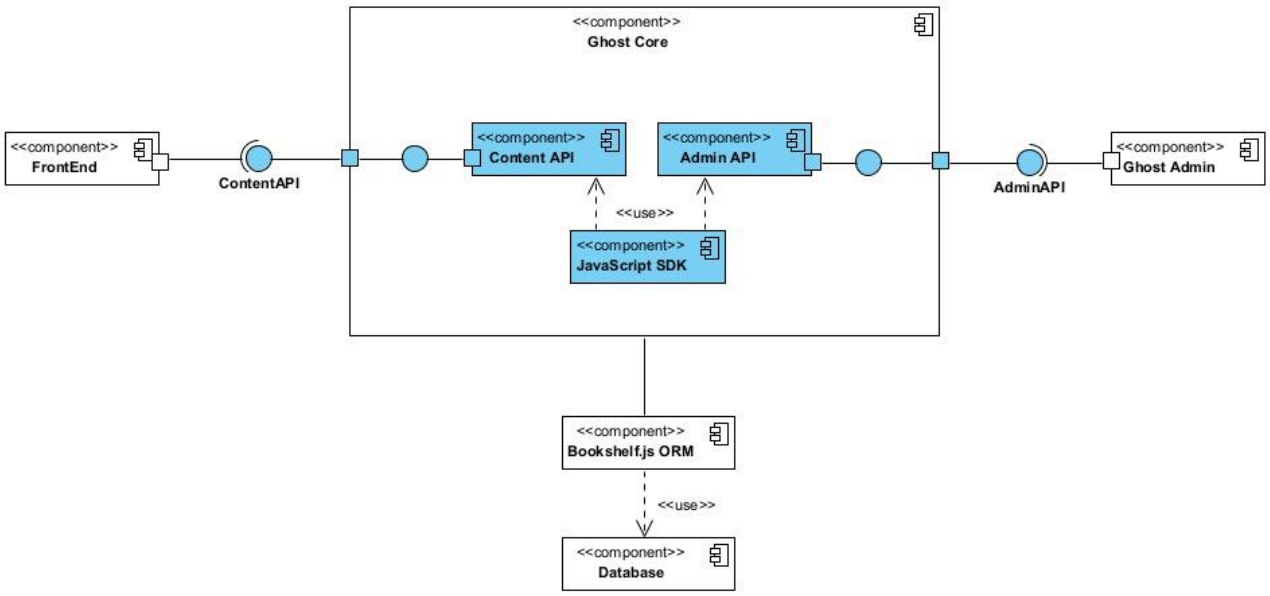
1.1. **Nombre Aplicación:** Ghost
1.2. **Versión:** 4.41.3
1.3. **Descripción**:

Ghost es una aplicación profesional open source para crear publicaciones web, construida sobre un conjunto de herramientas desarrolladas en node.js

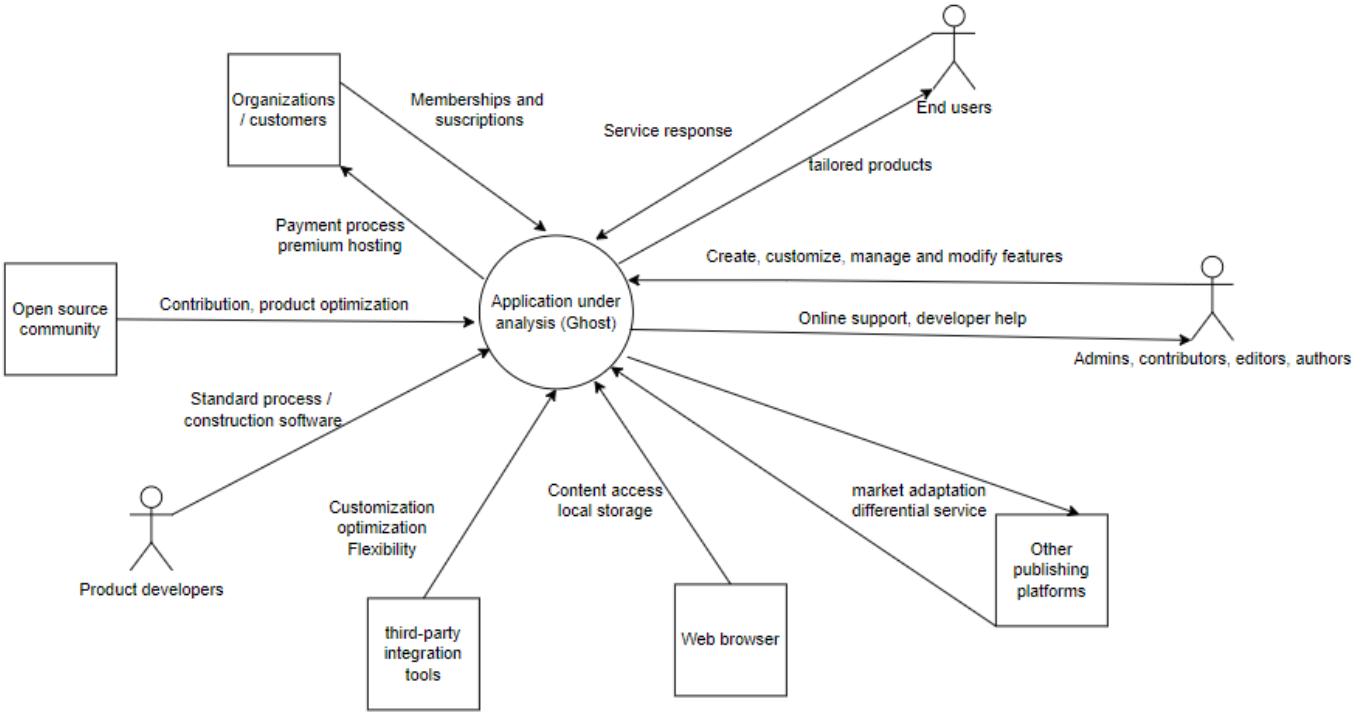
1.4. **Funcionalidades Core**:

- Autenticación
- Ver sitio
- Posts
- Filtrar posts
- Vistas
- Páginas
- Tags
- Miembros
- Menú perfil
- Settings

1.5. **Diagrama de Arquitectura:**



1.6. Diagrama de Contexto:



1.7. Modelo de Datos:

El modelo se encuentra alojado en el siguiente link:

ModeloDatos.png

1.8. Modelo de GUI:

El modelo GUI se encuentra alojado en los siguientes links:

Modelo_GUI

2. Contexto de la estrategia de pruebas

2.1. Objetivos:

- Realizar pruebas a nivel de sistema y aceptación con el fin de validar el correcto funcionamiento de la aplicación, mediante el uso de diferentes herramientas de testing como API's de automatización, GUI rippers y monkeys.

- Diseñar y ejecutar un conjunto de pruebas de tipo funcional y no funcional que validen la respuesta del sistema ante diferentes tipos de entradas, en escenarios esperados e inesperados.

2.2. Duración de la iteración de pruebas:

A continuación, se detallan las fechas y duración del proyecto:

Fecha de inicio: 23 de mayo de 2022

Fecha final de entrega: 17 de julio de 2022

Duración: 8 semanas

2.3. Presupuesto de pruebas:

2.3.1. Recursos Humanos

Se contará con el apoyo de cuatro ingenieros: son ingenieros de automatización de pruebas senior con 5 años de experiencia en diferentes áreas y herramientas de pruebas como:

- Análisis, diseño, construcción y ejecución de pruebas automatizadas
- Definición de casos de prueba manuales que no pueden ser automatizados, ya sea por limitación de recursos o tiempo
- Habilidades para realizar la configuración de ambientes de pruebas y de herramientas de pruebas de reconocimiento tipo Ripper y Monkey.
- Destreza en la implementación de pruebas de regresión visual VRT a partir de imágenes obtenidas en pruebas e2e u otros recursos visuales.
- Experiencia en la definición y ejecución de escenarios de prueba a partir de generación de datos mediante diferentes tipos de origen: datos a priori, datos pseudoaleatorios y datos aleatorios.

La disponibilidad de tiempo de los ingenieros será la siguiente:

Tiempo disponible por Ingeniero: 8 horas / semana.

Tiempo total disponible por Ingeniero: 64 horas.

2.3.2. Recursos Computacionales

Para la realización del proyecto se contará con los siguientes recursos computacionales y tecnológicos:

- 4 computadores
  - Procesador Intel Core I7 o superior
  - 16 GB RAM
  - Disco duro 1TB
- Servidores en la nube
  - (600 horas/máquina)
  - AWS EC2
  - Procesadores Intel Core I7 con 3.2 GHz (turbo de 4,6 GHz)
  - 6 núcleos físicos (12 núcleos lógicos)
  - 32 GB de memoria
- Software
  - Editor de código: Visual Studio Code
  - Pruebas e2e: cypress, Kraken
  - Pruebas de reconocimiento: RIPuppet, Monkey-Cypress
  - Control de versiones: GIT, Github

2.3.3. Recursos Económicos para la contratación de servicios/personal:

Para el proyecto no se contará con recursos monetarios para contratación de servicios externos o de outsourcing. Por lo tanto, se aprovechará al máximo los recursos computacionales mencionados anteriormente, para lograr una mayor cobertura de pruebas.

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

Con el fin de describir y relacionar el conjunto de pruebas utilizadas en esta estrategia, se muestra la tabla a continuación:

Nivel	Tipo	Técnica	Objetivo
Sistema	Funcionales Caja negra Positivas Negativas	Manuales	Se enfocan en pruebas smoke para garantizar que la aplicación funcione y no tenga defectos bloqueantes. Adicionalmente, escenarios de prueba críticos para la aplicación y aquellos que son costosos de automatizar por su tiempo de implementación y dificultad de replicar.
Sistema	Funcionales Caja negra Positivas, negativas	GUI Ripping	Explorar de manera sistemática la aplicación para lograr una mayor cobertura de código bajo escenarios de prueba aleatorios.
Sistema	Funcionales Caja negra Positivas, negativas	Monkey testing	Se utilizará en la fase inicial para detectar eventos inesperados que generen algún tipo de crash en la aplicación haciendo uso de data aleatoria.
Aceptación	Funcionales Caja negra Positivas	API's de automatización	Validar el correcto funcionamiento de la aplicación mediante la ejecución de pruebas de extremo a extremo interactuando con

			varias funcionalidades de forma y en un ambiente real.
Aceptación	No funcionales Caja negra Positivas, negativas	Regresión visual	Detectar la presencia de errores visuales o cambios inesperados que se hayan podido insertar en la interfaz web de la aplicación al momento de generar una nueva versión, y que puedan impactar la experiencia de usuario.
Aceptación	Funcionales Caja negra Positivas Negativas	API's de automatización con generación de datos	Validar el correcto funcionamiento de la aplicación mediante la ejecución de pruebas de extremo a extremo, haciendo uso de datos generados a partir de diferentes tipos de estructuras (datos a priori, datos pseudoaleatorios, datos aleatorios)

Con respecto a los niveles de pruebas, se consideraron utilizar diferentes niveles en la estrategia a los ya planteados en la tabla anterior. Sin embargo, algunos fueron descartados debido a ciertas consideraciones y restricciones:

- **Pruebas de unidad:** Nos permitirían lograr la mayor cobertura de las funcionalidades de la aplicación. Aunque no harán parte del presupuesto de pruebas, se espera que durante la etapa construcción y desarrollo de la aplicación hayan sido implementadas.

2.5. Distribución de Esfuerzo

La distribución de esfuerzo que se realizará durante las 8 semanas que dura el proyecto contempla diferentes actividades enfocadas en el análisis, diseño, implementación y ejecución de pruebas tanto manuales como automatizadas. Así mismo, se destina un tiempo al final del proyecto para realizar el análisis de resultados y la generación de los reportes a entregar a los diferentes stakeholders vinculados. En la siguiente tabla se detallan las tareas mencionadas:

Semana	Tareas	I1	I2	I3	I4	Horas totales	Horas máquinas virtuales
1	- Análisis y diseño de pruebas	4		4		8	
1	- Instalación y configuración de ambientes de prueba		4		4	8	
1	- Pruebas manuales exploratorias	4		4		8	
1	- Pruebas de reconocimiento con Monkey Testing		4		4	8	
2	- Pruebas de reconocimiento con GUI Ripping	8	8	8	8	32	100
3	- Ejecución pruebas E2E Grupo funcionalidades # 1	8	8	8	8	32	100
4	- Ejecución pruebas E2E Grupo funcionalidades # 2	8	4	8	4	24	100
4	- Análisis de resultados parciales - Iteración 1		4		4	8	
5	- Ejecución de pruebas de regresión visual versión **4.26.1** *	4	8	4	8	24	100
5	- Ejecución pruebas manuales – casos especiales	4		4		8	

6	- Ejecución de pruebas E2E con escenarios de data	8	8	8	8	32	100
7	- Retesting casos fallidos y fixeados	8	8	8	8	32	100
8	- Análisis de resultados finales	8		8		16	
8	- Generación de reportes		8		8	16	
TOTAL		**64**	**64**	**64**	**64**	**256**	**600**

* En las pruebas de regresión visual VRT, vale aclarar que en nuestra estrategia suponemos que en la semana # 5 se desplegará una nueva versión de la aplicación bajo pruebas. Sin embargo, el código de las pruebas VRT disponible en el repositorio para ejecutar estas pruebas fue implementado en la versión 4.26.1.

A continuación, describimos algunas de las decisiones tomadas para escoger esta distribución:

- En la primera semana se han planeado varias actividades con el objetivo de explorar la aplicación y entender cómo se encuentra construida, así como configurar los ambientes de ejecución tanto en los equipos físicos como en las máquinas virtuales que correrán en los servidores AWS. Por esta razón, no se hace inician aún las pruebas en las máquinas virtuales, las cuales serán utilizadas de manera continua desde la segunda semana hasta el final de la iteración.

- La distribución se realizó basada en el trabajo por pares, es decir, que los ingenieros trabajarán en gran parte del tiempo en parejas. Esto, con el fin de que puedan recibir feedback continuo de su compañero de equipo y, por otro lado, para lograr abarcar todas las tareas planeadas por semana. Sin embargo, existen algunas tareas que deberán realizarse de manera conjunta por parte de todos los miembros, como los son el análisis de resultados y generación de reportes.

- Aunque todos los ingenieros tienen experiencia en las técnicas y herramientas de pruebas que se utilizarán, cada uno tiene diferentes fortalezas que son importantes para garantizar una mayor calidad de trabajo. Por esta razón, estás se tendrán en cuenta al momento de distribuir las tareas dentro del equipo.

- Se asignó un pequeño espacio de tiempo para ejecutar pruebas manuales (semana 4) con el fin de probar escenarios que no hayan podido ser automatizados ya sea por limitaciones tecnológicas o por su difícil replicación.

- El tiempo de ejecución de pruebas asignado a las máquinas virtuales comprende la ejecución de las pruebas bajo diferentes combinaciones de sistemas operativos y navegadores así: Mac-Safari, Mac-Chrome, Windows-Edge y Windows-Chrome, Safari y Edge. Esta combinación ha sido elegida dado que son los navegadores nativos de cada sistema operativo y Google Chrome por ser el navegador más usado, permitiéndonos ampliar el dominio de entradas de la aplicación bajo pruebas.