# **ESTRATEGIA DE PRUEBAS**

## **APLICACIÓN BAJO PRUEBAS**

### **NOMBRE DE LA APLICACIÓN**

GHOST

### **VERSIÓN**

3.42.5

### **DESCRIPCIÓN**

Ghost es un software para blogs de código abierto, escrito en JavaScript y distribuido bajo una licencia MIT, fue diseñado para simplificar el proceso de generación de blogs individuales.

El desarrollo de esta aplicación se basa en NodeJs convirtiendo en una aplicación multi-plataforma, ofreciendo alto rendimiento, buena capacidad de personalización y administración.

### **FUNCIONALIDADES**

* **Home**: Modulo principal donde se da la visualización de la aplicación de cara a los usuarios que visitan el blog.
* **Administración de post:** Modulo de administración de post, en donde se pueden evidenciar todos los posts actualmente publicados, agendados o borradores.
* **Administración de paginas**: Modulo de administración de paginas, similar a los posts, las paginas permitirán un espacio único para el contenido que se desee en un URL general.
* **Administración de tags**: Modulo de administración de tags, es el encargado de tipificar las publicaciones que se desee, teniendo tipificación para tags externos o tags internos.
* **Administración de staff**: Modulo para invitar usuario que estaran a cargo del blog con diferentes roles y responsabilidades como lo pueden ser
  + - Autores
    - Administradores
    - Editores
    - Contribuyente
* **Configuraciones generales:** Modulo encargado de las configuraciones generales del sitio configuraciones tales como:
  + **Información de publicaciones**
    - **Titulo**: Titulo general para la pagina
    - **Meta data**: los datos de los datos para brindar una mayor información
    - **Zona horaria**: Zona horaria utilizada en la pagina, útil para las publicaciones programadas.
    - **Idioma**: Lenguaje que manejara la pagina
  + **Identidad de publicaciones**
    - **Icono de la publicación**: Icono general de la pagina
    - **Logo**: Logo general de la pagina
    - **Cubierta**: Imagen de cubierta de la pagina
    - **Redes Sociales:** Información a redes sociales como twitter y Facebook
    - **Configuración avanzada**
      * Sitio privado
* **Diseño:** Modulo de administración para la parte visual de la pagina, comprende componentes tales como:
  + **URL**: Rutas en las cuales estarán hospedas las paginas
  + **Navegación**: rutas de navegación de la aplicación
  + **Temas**: Diseño grafico para la pagina.
* **Administración de pagina mediante código:** Modulo encargado de la administración avanzada
* **Plugins**: Modulo para la administración avanzada de plugins (herramientas) que pueden ser integrados en la plataforma.
* **Características experimentales.**

## **DIAGRAMA DE ARQUITECTURA**

[Diagrama de arquitectura](https://drive.google.com/file/d/1yrkW0yAqFrAXJ7HpVQYVlpZU-Yb7wjmD/view?usp=sharing)

## **DIAGRAMA DE CONTEXTO**

[Diagrama de contexto](https://drive.google.com/file/d/1mlEjfxI0pCviCoqZLbtu2H0LrJaGRore/view)

## **MODELO DE DATOS**

Application, table, Word

Description automatically generated

Para más detalle: [Diagrama externo](https://drive.google.com/file/d/1Yk27_3SJ2kHz-sffND7W_bcl5oRR2xIP/view?usp=sharing)

## **MODELO GUI**

[Modelo externo](https://drive.google.com/file/d/1knIfq70KS1AoryD5OYmvlA2ya1LGN4GA/view?usp=sharing)

## **CONTEXTO DE LA ESTRATEGIA DE PRUEBAS**

### **VIDEO**

[Video](https://youtu.be/UPaL75sm8O4)

### **OBJETIVOS**

1. Validar correctamente las funcionalidades Core del sistema de blog Ghost
2. Identificar errores que puedan afectar las funcionalidades principales
3. Realizar las pruebas acordes al presupuesto entregado
4. Compartir el plan idóneo de pruebas

### **DISTRIBUCIÓN DE ESFUERZO**

Para la distribución del esfuerzo se utilizará una estrategia enfocada al artefacto Ghost y todos sus **requerimientos funcionales** con un patrón piramidal; Distribuido de la siguiente manera.

![A picture containing logo

Description automatically generated]()

Se omiten del marco de referencia las pruebas más fáciles que son las unitarias puesto que el artefacto ya se encuentra en producción. (supuesto), cada estrategia se evidenciará el **por qué**, del componente a testear, **cómo** se realizará la prueba, la toma de **resultados** y la **conclusión** de dicha prueba.

#### **ESTRATEGIA DE PRUEBAS**

##### **PRUEBAS MANUALES**

* **Home**
  + **¿Por qué?** Home es componente principal donde se reflejan el resultado de nuestro artefacto.
  + **¿Cómo?** Validar la data actual que posee Ghost dentro de la navegabilidad de la aplicación.

**Resultados** Captura de pantalla del resultado de la navegación en el home.

* + **Conclusión** Los valores deberán verse reflejados tal y como se tienen en el módulo administrativo.
* **Administración de post, pages y tags** 
  + **¿Por qué?** Post, pages y tags al ser parte del Core de Ghost debe ser validado en totalidad
  + **¿Cómo?** Se debe realizar pruebas del componente dentro de un **happy path** y un **sad path** con datos generados a-priori[[1]](#footnote-2) con la estrategia de datos definidas en el repositorio.
  + **Resultados** la evidencia de esta ejecución se debe ver suministrada en el formato de pruebas[[2]](#footnote-3).
  + **Conclusión** Los valores generados permitirán validar el correcto funcionamiento de la administración de post, pages y tags.
* **Administración de staff**
  + **¿Por qué?** Staff es parte del Core de Ghost el cual permite acceso a Ghost
  + **¿Cómo?** Se debe realizar pruebas del componente dentro de un **happy path** y un **sad path** con datos generados a-priori de manera manual se deben crear correos validos e inválidos.
  + **Resultados** la evidencia de esta ejecución se debe ver suministrada en el formato de pruebas con la captura de pantalla del correo enviado.
  + **Conclusión** las invitaciones son un factor importante puesto que sin las invitaciones no se podrá administrar Ghost.
* **Administración de staff**
  + **¿Por qué?** Staff es parte del Core de Ghost el cual permite acceso a Ghost
  + **¿Cómo?** Se debe realizar pruebas del componente dentro de un **happy path** y un **sad path** con datos generados a-priori de manera manual se deben crear correos validos e inválidos.
  + **Resultados** la evidencia de esta ejecución se debe ver suministrada en el formato de pruebas con la captura de pantalla del correo enviado.
  + **Conclusión** las invitaciones son un factor importante puesto que sin las invitaciones no se podrá administrar Ghost.
* **Configuración general** (Información de publicaciones)
  + **¿Por qué?** El modulo de configuración general es el encargado de atributos tales como **Titulo**, **Metadatos**, **Zonas horarias** e **Idiomas**, por ende, se deben realizar las validaciones de las configuraciones generales para validar su funcionamiento.
  + **¿Cómo?** Se debe realizar pruebas del componente dentro de un **happy path** siguiendo la documentación suministrada por Ghost. [Documentación](https://ghost.org/help/topic/setting-up/)
  + **Resultados** la evidencia de esta ejecución se debe ver suministrada en el formato de pruebas con la captura de las configuraciones aplicadas.
  + **Conclusión** las configuraciones deben realizar cambios sobre el comportamiento de Ghost.
* **Configuración general** (Identidad de publicaciones)
  + **¿Por qué?** El modulo de configuración general es el encargado de atributos tales como **Icono de la publicación**, **Logo**, **Cubierta** e **Redes Sociales y Configuración avanzada**, estos comprenden configuraciones especificas por ende los escenarios no son muchos.
  + **¿Cómo?** Se debe realizar pruebas del componente dentro de un **happy path** siguiendo la documentación suministrada por Ghost. [Documentación](https://ghost.org/help/topic/setting-up/) se debe variar la configuración especifica y documentar este cambio aplicado en el artefacto.
  + **Resultados** la evidencia de esta ejecución se debe ver suministrada en el formato de pruebas con la captura de las configuraciones aplicadas.
  + **Conclusión** las configuraciones deben realizar cambios sobre el comportamiento de Ghost.
* **Diseño** 
  + **¿Por qué?** El módulo de diseño es el encargado de atributos tales como **URL, Navegación** y **Temas** para configurar atributos visuales de la aplicación.
  + **¿Cómo?** Se deben ajustar las URL por un happy path y validar su navegabilidad. Se debe poder cambiar exitosamente un Tema
  + **Resultados** la navegabilidad debe verse validada en sus happy path sin fallos aparentes.
  + **Conclusión** la configuración aplicada sobre el artefacto no debe tener afectaciones en su navegabilidad o diseño aparente.
* **Administración de página mediante código**
  + **¿Por qué?** Al ser un módulo avanzado debe poseer unas pruebas extensas.
  + **¿Cómo?** Realizar la inyección de código correspondiente a un escenario básico de código.
  + **Resultados** el artefacto debe funcionar exitosamente después de la inyección de código.
  + **Conclusión** la inyección de código debe ser útil y sin fallos en el artefacto.
* **Plugins**
  + **¿Por qué?** El módulo de plugins permite integraciones de herramientas para agregar más funcionalidades al artefacto
  + **¿Cómo?** Realizar la vinculación exitosa de un nuevo plugin.
  + **Resultados** El artefacto deberá seguir funcionando con su funcionalidad base más el añadido por el plugin
  + **Conclusión** El correcto funcionamiento del artefacto con su funcionalidad añadida

##### **AUTOMATIZACIÓN POR EXPLORACIÓN**

A través de las pruebas automatizadas por reconocimiento y exploración buscamos lograr de forma autónoma explorar la aplicación mediante la generación aleatoria de eventos, entradas sobre la interfaz gráfica y mediante el recorrido sistemático de la GUI.

1. **Login**:
   * **¿Por qué?** se busca sobre este flujo verificar que la aplicación brinda las validaciones suficientes para el ingreso a la aplicación.
   * **¿Cómo?** Haciendo inserción de texto aleatorio sobre la estructura del email de entrada.
   * **Resultados** Mensajes guía sobre el error en la estructura del email
   * **Conclusión** El formulario de entrada realiza las validaciones sobre la estructura de la cadena.

* **Páginas**:
  + **¿Por qué?** Buscamos validar aleatoriamente las opciones de creación y edición que nos brinda el flujo de pages.
  + **¿Cómo?** Monkeys y rippers que a través de esta opción evaluar el acceso, ingreso de texto y las diferentes opciones disponibles para configurar.
  + **Resultados** Se puede manipular correctamente las configuraciones para la creación de páginas.
  + **Conclusión** Las opciones de configuración en la página funcionan y no se generan errores al utilizarlas.
* **Posts:**
  + **¿Por qué?** Buscamos validar aleatoriamente las opciones de creación y edición que nos brinda el flujo de posts.
  + **¿Cómo?** Monkeys y rippers que a través de esta opción evaluar el acceso, ingreso de texto y las diferentes opciones disponibles para configurar.
  + **Resultados** Se puede manipular correctamente las configuraciones para la creación de posts.
  + **Conclusión** Las opciones de configuración en la página funcionan y no se generan errores al utilizarlas.
* **Staff**:
  + **¿Por qué?** Buscamos validar aleatoriamente las opciones de edición de un usuario dentro de Staff.
  + **¿Cómo?** Ingresando a la opción de edición de usuario lanzando monkeys y rippers Tomando el usuario de administración para realizar su edición.
  + **Resultados** No se generan errores inesperados con el uso de las opciones del flujo
  + **Conclusión** La edición del usuario guarda los cambios correctamente y el flujo es correcto.
* **Tags**:
  + **¿Por qué?** Buscamos validar aleatoriamente las validaciones sobre cada campo de ingreso de datos.
  + **¿Cómo?** Realizando el ingreso de información aleatoria que permita evaluar los tipos de datos recibidos y sus respectivas validaciones.
  + **Resultados** Los campos realizan las validaciones necesarias
  + **Conclusión** Los campos definidos solicitan el tipo de datos definidos según cada una de sus etiquetas.

##### **AUTOMATIZACIÓN POR COMPONENTES**

Buscamos a través de la colocación de las pruebas por componentes lograr separar las funcionalidades de manera que podamos aislar la funcionalidad específica a evaluar, y separarla de su contexto o afectación histórica de otras funciones.

**Características para evaluar:**

1. **Login:** tratamos de realizar el aislamiento de componentes y la primera característica que encontramos en la aplicación es el login de entrada del aplicativo, este componente o funcionalidad es la más fácil de aislar del resto de funciones de la aplicación, ya que no tiene dependencia de otro flujo.
   1. **Escenario Falla Estructura Email**
      * **¿Por qué?** se busca sobre este escenario verificar que la aplicación brinda las validaciones suficientes
      * **¿Cómo?** Haciendo inserción de erróneos sobre la estructura del email de entrada.
      * **Resultados** Mensajes guía sobre el error en la estructura del email
      * **Conclusión** El formulario de entrada realiza las validaciones sobre la estructura de la cadena
   2. **Escenario Falla Password**
      * **¿Por qué?** Se busca en este escenario la evaluación de la seguridad de la aplicación.
      * **¿Cómo?** Con la creación de un perfil con anterioridad e ingreso de contraseñas que no pertenezcan a dicho perfil.
      * **Resultados** Esto se realiza con el fin de validar el tipo de estructura de cadenas validas sobre el formato expuesto y a su vez las restricciones sobre la cantidad de veces que se permite realizar ingreso de una contraseña errada en su número de intentos.
      * **Conclusión** Da instrucciones para guiar al cambio de contraseña o reporte de errores.
   3. **Escenario Usuario Errado**
      * **¿Por qué?** Para este escenario tenemos que validar el ingreso de un usuario no existente.
      * **¿Cómo?** Por medio de la introducción de usuarios inválidos.
      * **Resultados** Se espera que la respuesta de la aplicación logre guiar al usuario específicamente sobre el error presente.
      * **Conclusión** Ya que no debe confundirse con un error de contraseña u otros.
   4. **Escenario Login Ok**
      * **¿Por qué?** En este escenario especifico buscamos validar el acceso correcto a la aplicación.
      * **¿Cómo?** Con un usuario previamente definido.
      * **Resultados** Se espera el correcto ingreso del usuario definido.
      * **Conclusión** El ingreso sobre la aplicación se realiza correctamente
2. **Paginas:** sobre esta función buscamos probar la generación, edición y actualización de páginas con el fin de realizar las asociaciones necesarias, cuando tengamos el post definido en publicación y necesitemos asociar las diferentes paginas disponibles para el post.
   1. **Escenario Configuraciones sobre la opción páginas**
      * **¿Por qué?** Cuando estamos realizando la creación de páginas tenemos una herramienta disponible para utilizar que son las configuraciones de la misma.
      * **¿Cómo?** Buscamos a través de este escenario evaluar el acceso a esta opción para páginas y también evaluar las diferentes opciones disponibles para configurar.
      * **Resultados** Se puede manipular correctamente las configuraciones para la creación de páginas.
      * **Conclusión** Las opciones de configuración en la página funcionan tal como se intuye su función.
   2. **Escenario Creación y validación de página**
      * **¿Por qué?** Desde este escenario buscamos validar que podemos realizar la creación de una página y validar que esta quedo relacionada en los borradores de las paginas disponibles.
      * **¿Cómo?** Realizando la creación de la página a través de su flujo.
      * **Resultados** Se crea la página correctamente y guarda su borrador.
      * **Conclusión** La opción de creación de página trabaja y se ejecuta correctamente.
   3. **Escenario Creación y Publicación de una página**
      * **¿Por qué?** A través de este escenario buscamos validar la creación de una página y realizar la publicación de ella, para que pueda ser invocada dese de cualquier post.
      * **¿Cómo?** Realizando la creación de la página a través de su flujo y posterior publicación.
      * **Resultados** Se crea la página correctamente y realiza su publicación correctamente
      * **Conclusión** La publicación y creación de una página se realiza correctamente.
   4. **Escenario asignación de URL a la página**
      * **¿Por qué?** Cada página contiene la descripción del enlace al cual esta atiende o pertenece.
      * **¿Cómo?** En este escenario se busca validar que el formulario tome en cuenta el formato de diferentes tupos de urls asignadas a través del ingreso de esta cadena.
      * **Resultados** La página realiza correctamente la validación del formato de la URL
      * **Conclusión** La asignación de URL básica a la URL expuesta desde la página tiene sus validaciones correctamente realizadas.

1. **Post:** esta consideramos que es la función principal de toda la aplicación, ya que es la función sobre la cual giran las demás en su entorno, por lo cual planteamos los siguientes escenarios:
   1. **Escenario validación objetos de tipo botón**
      * **¿Por qué?** Buscamos a través de este escenario validar si el botón creación de post tiene relevancia sobre otras acciones encadenadas.
      * **¿Cómo?** Si existe dependencias mal solventadas a través de las acciones encadenadas de los botones para la opción de generación de post.
      * **Resultados** Se evidencia la jerarquía de las acciones
      * **Conclusión** La generación de acciones esta correctamente ordenada por jerarquía en las acciones ejecutadas.
   2. **Escenario generación de borradores**
      * **¿Por qué?** Sobre este escenario buscamos realizar el flujo correspondiente a la creación de un post y posterior validación de que este se encuentre puesto en los borradores a implementar o listos para asignar al calendario o publicar directamente.
      * **¿Cómo?** Realizando la creación de un post y validando en borradores la creación de este.
      * **Resultados** Se crean los posts correctamente
      * **Conclusión** Se realiza la creación y gestión de borradores correctamente.
   3. **Escenario configuraciones de borradores**
      * **¿Por qué?** Sobre este escenario buscamos realizar la validación de las diferentes opciones de configuraciones disponibles una vez que realizamos la creación del post y este se encuentra en borradores.
      * **¿Cómo?** Realizando la creación de post y posteriormente gestión de sus configuraciones.
      * **Resultados** Las funciones de configuración de post funcionan correctamente.
      * **Conclusión** Se puede realizar la configuración de un post correctamente una vez este haya sido creado.
   4. **Escenario creación y publicación desde los borradores**
      * **¿Por qué?** Este escenario busca completar el flujo de la creación de post hasta su publicación pasando por la creación de post que quedan puestos en borradores para posteriormente ser puestos en el calendario para ser publicados.
      * **¿Cómo?** Realizando la creación de post y posteriormente colocando su publicación.
      * **Resultados** El post se crea correctamente y se publica.
      * **Conclusión** Se puede publicar un post creado.
2. **Staff:** es una función que gestiona la cantidad de colaboradores dentro de la aplicación al igual que creadores de contenido
   1. **Escenario editar usuario**
      * **¿Por qué?** Buscamos validar la edición de un usuario dentro de Staff.
      * **¿Cómo?** Tomando el usuario de administración para realizar su edición.
      * **Resultados** La edición del usuario se realiza correctamente
      * **Conclusión** La edición del usuario guarda los cambios correctamente y el flujo es correcto.
   2. **Escenario invitación errada**
      * **¿Por qué?** Se requiere verificar que valida la estructura del email correctamente para realizar el envío de la invitación.
      * **¿Cómo?** Realizando la inclusión de cadenas de errores aleatorias esperando que se realice la validación correctamente.
      * **Resultados** El campo de ingreso de la invitación a realizar el envío de la invitación contiene las validaciones correspondientes al tipo de String de un correo
      * **Conclusión** Se evidencia que las validaciones están correctamente construidas.
   3. **Escenario revocar invitación**
      * **¿Por qué?** Se requiere hacer la validación de revocación de la invitación.
      * **¿Cómo?** Generando una invitación que posteriormente se revoca y se espera que la lista de invitaciones se actualice.
      * **Resultados** La lista de invitaciones retira la invitación revocada.
      * **Conclusión** Se puede realizar una invitación y ser revocada posteriormente.
   4. **Escenario invitación enviada correctamente**
      * **¿Por qué?** Se espera que una invitación pueda ser enviada correctamente.
      * **¿Cómo?** Realizando el envío de una invitación a un correo valido y posteriormente evidenciando que la lista de invitaciones se actualiza correctamente.
      * **Resultados** La lista de invitaciones se actualiza correctamente con el envío de la invitación.
      * **Conclusión** Se puede realizar el envío de una invitación para ser miembro correctamente.
3. **Tag:** es una función para enlazar un tema específico a un post, para que en su pie de página se pueda relacionar un link a través de un Tag.
   1. **Escenario Utilizar todos los campos de edición de Tag**
      * **¿Por qué?** Es necesario realizar las validaciones específicas sobre cada uno de los campos de ingreso de datos.
      * **¿Cómo?** Realizando una generación de data a priori que permita evaluar los tipos de datos recibidos en todos los campos de entrada.
      * **Resultados** Los campos realizan las validaciones necesarias
      * **Conclusión** Los campos definidos solicitan el tipo de datos definidos según cada una de sus etiquetas.
   2. **Escenario creación y edición de Tags**
      * **¿Por qué?** Es necesario validar la creación de Tags para realizar su asociación interna.
      * **¿Cómo?** Realizando la creación del tag y listando los Tag disponibles para realizar la relación.
      * **Resultados** Se listan los tags creados correctamente.
      * **Conclusión** Se evidencia que es posible crear tags y tenerlos disponibles posteriormente para su relación.
   3. **Escenario guardado de Tags creados**
      * **¿Por qué?** Se requiere validar que los datos creados en los Tags son guardados correctamente.
      * **¿Cómo?** Generando una lista de tags a crear y comparando la lista de Tags creados.
      * **Resultados** La lista de Tags se ve actualizada en el mismo número de su creación.
      * **Conclusión** Se puede realizar la creación de diferentes Tags y tener disponible su uso para la relación.
   4. **Escenario eliminación de Tags**
      * **¿Por qué?** Se requiere validar que se pueda eliminar los Tags creados con anterioridad.
      * **¿Cómo?** Realizando la creación de una lista de Tags que posteriormente serán eliminados.
      * **Resultados** La lista de Tags se ve actualizada según su eliminación.
      * **Conclusión** Se puede crear y eliminar Tags creados anteriormente.

##### **AUTOMATIZACIÓN DE INTEGRACIÓN PRUEBAS E2E**

Para la implementación de pruebas E2E vamos a tomar como estrategias de implementación el uso de Behaviour-Driven Development y Api Testing Development; para el uso de estas dos estrategias requerimos de herramientas específicas, para el caso de BDD utilizaremos Cucumber a través del uso de implementaciones en librerías como Kraken y Webdrawio. A su vez también recurriremos al uso de funciones para realizar generación de datos aleatorios, a priori y dinámicos en definición de escenarios aleatorios.

**Características por evaluar:**

1. **Login:** El objetivo sobre la evaluación de ingreso de la aplicación está en la generación aleatoria de diferentes correos, usuarios y acciones aleatorias.
   1. **Escenario Falla Estructura Email**
      * **¿Por qué?** Es necesario evaluar los limites en caracteres permitidos para el ingreso de un email, y valoración de su estructura con diferentes caracteres a ingresar.
      * **¿Cómo?** Evaluaremos este escenario por medio de la generación de datos aleatorios con la herramienta **Faker** y a su vez validaremos la gestión de los datos construidos a priori dentro de la aplicación.
      * **Resultados** Se obtiene una correcta validación de caracteres especiales y estructura obligatoria de un correo electrónico, al igual que una limitación de caracteres bien definida.
      * **Conclusión** El campo de ingreso de email cuenta con una gestión correcta de validación de estructura, puntuación.
   2. **Escenario Contraseña Invalida**
      * **¿Por qué?** Es necesario validar si la generación de diferentes cadenas aleatorias como a priori permiten el ingreso a la aplicación, al igual que la guía especifica que brinda la aplicación para guiar la solución del error presentado.
      * **¿Cómo?** Se realizará la creación usuarios de forma aleatoria y se validará su acceso a través de la generación de datos a priori, y también se validará aleatoriamente diferentes tipos de contraseñas con caracteres especiales.
      * **Resultados** Las pruebas se ejecutan de manera exitosa tal que se realiza un registro correcto a la vez que su ingreso también.
      * **Conclusión** Se evidencia claramente la restricción en el acceso de claves erradas y brinda la información suficiente para dar solución al no poder ingresar.
   3. **Escenario Usuario Errado**
      * **¿Por qué?** Es necesario validar que los usuarios no registrados no tengan acceso a la aplicación.
      * **¿Cómo?** Realizando una generación aleatoria de usuarios que no van a estar registrados como usuarios válidos.
      * **Resultados** Se evidencia una correcta validación de usuarios que no están registrados en la aplicación.
      * **Conclusión** El ingreso sobre la aplicación es seguro y valida la existencia de los usuarios.
   4. **Escenario Login Ok**
      * **¿Por qué?** Necesitamos validar que los usuarios creados en la aplicación tienen acceso a ella.
      * **¿Cómo?** Por medio de la generación de una lista de usuarios específicos que posteriormente será ejecutada y validaremos si todos cuentan con el acceso necesario a la aplicación.
      * **Resultados** Se evidencia que la creación de los usuarios que colaboraran en la aplicación se crea correctamente y tiene acceso a la aplicación con sus credenciales definidas.
      * **Conclusión** La aplicación guarda correctamente las credenciales definidas para cada uno de los usuarios.
2. **Paginas:** por medio de la estrategia de pruebas E2E buscamos validar los flujos completos de la creación de páginas, validaciones dentro de sus formularios y validaciones en su creación.
   1. **Escenario Configuraciones sobre la opción páginas**
      * **¿Por qué?** Es necesario verificar si el uso de las diferentes opciones de configuraciones de páginas está disponible para el componente de páginas.
      * **¿Cómo?** Por medio de la creación de diferentes páginas y asociación de sus configuraciones específicas.
      * **Resultados** Se evidencia que las configuraciones son funcionales para la gestión de asociación de urls autores y demás.
      * **Conclusión** La creación de páginas y su posterior asociación de configuraciones funciona correctamente.
   2. **Escenario Creación y validación de página**
      * **¿Por qué?** Es necesario ejecutar el flujo completo de creación y validación de existencia de la página creada con anterioridad.
      * **¿Cómo?** A través de las pruebas e2e podemos ejecutar el flujo de creación y validación de existencia encadenando la generación de diferentes tipos de títulos y contenidos en las páginas.
      * **Resultados** Se logra realizar la validación de existencia de las paginas creadas correctamente estas hacen uso de la opción de borradores como locación de su pre-publicación.
      * **Conclusión** La creación de las páginas se realiza correctamente y se puede validar.
   3. **Escenario Creación y Publicación de una página**
      * **¿Por qué?** Se requiere realizar las pruebas de extremo a extremo, para validar la creación de páginas y su publicación para ser utilizadas en los Post.
      * **¿Cómo?** A través de la generación de datos a priori y aleatoria además del planteamiento de los escenarios dinámicos que permiten la creación de diferentes páginas con un contenido variado.
      * **Resultados** Se verifica la creación de diferentes páginas y se evidencia la publicación de las mismas.
      * **Conclusión** El escenario se ejecuta correctamente y se puede evidenciar que las diferentes páginas son creadas y publicadas adecuadamente.
   4. **Escenario asignación de URL a la página**
      * **¿Por qué?** Es necesario realizar las pruebas de extremo a extremo para validar la asociación de sitios a las páginas que serán enlazadas desde los Posts.
      * **¿Cómo?** Por medio de la ejecución de las pruebas E2E podemos realizar la validación de asignación de una URL específica para el sitio de la página definido.
      * **Resultados** La asignación de la URL específica sobre la que se va a tomar como recurso principal en la página queda correctamente asignada.
      * **Conclusión** Las asignaciones realizadas por medio de las pruebas de extremo a extremo, se guardan correctamente y son listadas en las configuraciones de las páginas.

1. **Post:** Para evaluar la funcionalidad de creación de diferentes Post, es necesario hacer uso de las pruebas de extremo a extremo para validar la totalidad de los flujos y las configuraciones que ellas conllevan:
   1. **Escenario validación objetos de tipo botón**
      * **¿Por qué?** Es necesario dentro del flujo de creación de Post, realizar la validación de jerarquía de los eventos desencadenados por los botones disponibles en la simple page.
      * **¿Cómo?** Por medio de la generación de eventos con introducción de concurrencia, podemos validar la jerarquía definida en los botones disponibles en la simple page de Post, por medio de las pruebas de extremo a extremo hacemos generación aleatoria de eventos click en los botones disponibles para validar la respuesta del componente.
      * **Resultados** Se evidencia claramente que los botones se encuentran jerarquizados y están a la espera de la habilitación del botón de creación de post.
      * **Conclusión** La construcción de la aplicación sobre el motor de inyección de vistas está bien definida y guarda la jerarquía necesaria.
   2. **Escenario generación de borradores**
      * **¿Por qué?** Por medio del uso de pruebas de extremo a extremo, esperamos realizar la generación de una cantidad de post importante, para realizar la validación necesaria de su creación a nivel de borradores.
      * **¿Cómo?** Con la generación de datos aleatoriamente en el encabezado y cuerpo del post, y también con el planteamiento de datos generados a priori esperamos realizar la acumulación de post generados sobre la pila de datos.
      * **Resultados** Los posts se crean correctamente y quedan definidos en borradores según el número de datos contenidos en la pila.
      * **Conclusión** Los post tienen una latencia correcta con respecto a la pila de la generación de datos.
   3. **Escenario configuraciones de borradores**
      * **¿Por qué?** Por medio de las pruebas E2E podemos validar la configuración de varios escenarios sobre el borrador de un Post.
      * **¿Cómo?** Una vez creados los posts definidos en la pila de datos podemos ejecutar los flujos correspondientes a configuraciones de post puestos en borradores.
      * **Resultados** La pila de datos se ve creada correctamente en los borradores de Post y actualizada por las configuraciones definidas.
      * **Conclusión** Las configuraciones disponibles en los borradores de post realizan la actualización correcta sobre la característica a cambiar de cada uno de los campos disponibles.
   4. **Escenario creación y publicación desde los borradores**
      * **¿Por qué?** Este es el flujo de extremo a extremo más largo que encontramos y es la definición final que esperamos encontrar creada correctamente con la pila de datos.
      * **¿Cómo?** Al configurar una pila de datos definida en la ejecución de la prueba podemos obtener un rango alto de prueba de datos para la generación de Post y su publicación.
      * **Resultados** La pila de datos con la cantidad de posts esperados se crea y se publica correctamente.
      * **Conclusión** Se puede asegurar que la funcionalidad de Posts y su posterior publicación es funcional.
2. **Staff:** validaremos a través de las pruebas de extremo a extremo la gestión de colaboradores sobre la aplicación, así como los autores de materiales o generadores de contenido variado en los posts.
   1. **Escenario editar usuario**
      * **¿Por qué?** Una vez creado el usuario dentro de la aplicación la generación de los test unitarios busca realizar la actualización de los datos del usuario anteriormente creado.
      * **¿Cómo?** A través de la pila de datos disponible en la creación de usuarios, podemos realizar la validación para hacer la actualización de datos del usuario que pretendemos crear.
      * **Resultados** Se puede realizar la actualización de usuarios correctamente, tomando como base la pila de datos generada anteriormente.
      * **Conclusión** La opción para hacer actualización de datos funciona correctamente.
   2. **Escenario invitación errada**
      * **¿Por qué?** Por medio de la generación de pruebas de extremo a extremo podemos probar las validaciones necesarias que se esperan en el formulario de la aplicación.
      * **¿Cómo?** Al tener una pila de datos podemos realizar la generación de inyección de datos con caracteres especiales al igual que probar los límites que se esperan encontrar en los formularios de ingreso de la invitación.
      * **Resultados** Se evidencia que el formulario de envío tiene correctamente configurado las validaciones necesarias que se esperan en la estructura de un email.
      * **Conclusión** El proceso de invitación de colaboradores funciona correctamente en la validación del formulario.
   3. **Escenario revocar invitación**
      * **¿Por qué?** Una vez enviada la invitación para colaborar en el proyecto, es necesario validar si esta invitación puede ser revocada o declinada.
      * **¿Cómo?** Una vez hemos ingresado en la aplicación una cantidad de invitaciones considerables, podemos proceder a realizar la revocación de todas estas a través de la opción que nos brinda la aplicación.
      * **Resultados** La lista de invitaciones se ve actualizada con respecto a la cantidad de revocaciones realizadas.
      * **Conclusión** Se evidencia que se pueden realizar las revocaciones de todas las invitaciones enviadas con anterioridad.
   4. **Escenario invitación enviada correctamente**
      * **¿Por qué?** Es necesario validar que se relaciona en la aplicación un correcto envió de invitaciones a colaborar en la creación de contenidos.
      * **¿Cómo?** Por medio de la generación de data a priori y aleatoria disponible en la pila de datos podemos evidenciar si logramos realizar el envío correcto de la invitación a colaborar con en la aplicación.
      * **Resultados** Las invitaciones se envían correctamente y se evidencia el registro correcto del envío.
      * **Conclusión** Se puede realizar la invitación masiva a colaboradores.
3. **Tag:** es una herramienta para relacionar temas específicos a los posts publicados dentro de la aplicación por lo tanto las pruebas e2e nos brindan la posibilidad de probar los flujos completos e iterados.
   1. **Escenario Utilizar todos los campos de edición de Tag**
      * **¿Por qué?** Porque se requiere validar que los campos de edición de tag están disponibles en el formato que sugiere en los marcadores de los campos.
      * **¿Cómo?** A través de la generación sobre la pila de datos de forma aleatoria o a priori y la inyección de estos datos en el formulario.
      * **Resultados** Se pueden utilizar adecuadamente todos los campos de ingreso del formulario
      * **Conclusión** Podemos gestionar las pruebas a través del uso de herramientas de extremo a extremo y generación de data sintética.
   2. **Escenario creación y edición de Tags**
      * **¿Por qué?** Porque debemos probar que un tag creado previamente puede ser editado en cualquiera de sus componentes o características.
      * **¿Cómo?** Realizando la creación de Tags utilizando la data puesta en las tuplas de data.
      * **Resultados** Se evidencia que se pueden crear y editar los tags creados correctamente.
      * **Conclusión** Se concluye que los Tags son creados y disponibles para ser editados.
   3. **Escenario guardado de Tags creados**
      * **¿Por qué?** Porque es necesario validar que la generación de Tags como disponibles para utilizar en los diferentes Posts.
      * **¿Cómo?** Una vez generada la cantidad de tags planeada en la tupla de datos de las pruebas e2e se busca la relación de estos tags en los diferentes posts disponibles.
      * **Resultados** Se evidencia que los tags creados pueden ser utilizados desde las relaciones de configuraciones de la relación de los posts.
      * **Conclusión** Se pueden utilizar los tags creados en las pruebas de extremo a extremo para ser relacionados en las configuraciones de los posts.
   4. **Escenario eliminación de Tags**
      * **¿Por qué?** Porque se requiere verificar que los tags creados a través de las pruebas de extremo a extremo pueden estar disponibles para ser eliminados de la aplicación.
      * **¿Cómo?** Haciendo uso de los datos ingresados y sobre la lista de los Tags creados.
      * **Resultados** Se pueden eliminar los tags creados a través de la tupla de datos generados.
      * **Conclusión** Los datos sobre Tags se crean correctamente, se pueden actualizar y también están disponibles para ser eliminados.

##### **AUTOMATIZACIÓN DE PRUEBAS DE REGRESION VRT**

Para la implementación de pruebas de regresión VRT vamos a usar el resultado de las pruebas E2E de la versión anterior y con ayuda de Backstop revisaremos de manera automatizada las diferencias entre las versiones a probar.

1. **Login:**
   * **¿Por qué?** Es necesario evaluar que no haya cambios inesperados en la interfaz gráfica, inputs de email, password
   * **¿Cómo?** Comparación por VRT de las capturas de pantalla generadas.
   * **Resultados** Se obtiene una correcta validación y comparación de la GUI. Reporte de la herramienta utilizada.
   * **Conclusión** La pantalla de login no sufrió cambios inesperados ni afectación funcional.
2. **Paginas:**
   * **¿Por qué?** Es necesario evaluar que no haya cambios inesperados en la interfaz gráfica, campos de captura de información, botones de publicación
   * **¿Cómo?** Comparación por VRT de las capturas de pantalla generadas **Resultados** Se obtiene una correcta validación y comparación de la GUI. Reporte de la herramienta utilizada.
   * **Conclusión** La pantalla de login no sufrió cambios inesperados ni afectación funcional.

1. **Post:**
   * **¿Por qué?** Es necesario evaluar que no haya cambios inesperados en la interfaz gráfica, campos de captura de información, botones de publicación.
   * **¿Cómo?** Comparación por VRT de las capturas de pantalla generadas**.**
   * **Resultados** Se obtiene una correcta validación y comparación de la GUI. Reporte de la herramienta utilizada.
   * **Conclusión** La pantalla de login no sufrió cambios inesperados ni afectación funcional.
2. **Staff:**
   * **¿Por qué?** Es necesario evaluar que no haya cambios inesperados en la interfaz gráfica, campos de captura de información
   * **¿Cómo?** Comparación por VRT de las capturas de pantalla generadas.
   * **Resultados** Se obtiene una correcta validación y comparación de la GUI. Reporte de la herramienta utilizada.
   * **Conclusión** La pantalla de login no sufrió cambios inesperados ni afectación funcional.
3. **Tags:**
   * **¿Por qué?** Es necesario evaluar que no haya cambios inesperados en la interfaz gráfica, campos de captura de información
   * **¿Cómo?** Comparación por VRT de las capturas de pantalla generadas **Resultados** Se obtiene una correcta validación y comparación de la GUI. Reporte de la herramienta utilizada.
   * **Conclusión** La pantalla de login no sufrió cambios inesperados ni afectación funcional.

## RECURSOS HUMANOS

### PRUEBAS MANUALES

* **Ingeniero Senior** con conocimientos de automatización con conocimientos intermedios de procesos de calidad contará una distribución diaria de **8** horas para realizar las pruebas manuales dentro del alcance, todo para un acumulado de **40** horas dentro de plan de ejecución.

### PRUEBAS POR EXPLORACIÓN

* **Ingeniero Junior** con habilidades en automatización contará una distribución diaria de **8** horas para realizar las pruebas y ajuste en scripts de pruebas aleatorias dentro del alcance, todo para un acumulado de **40** horas dentro de plan de ejecución.

### AUTOMATIZACIÓN DE COMPONENTES

* **Ingeniero Junior** que tenga conocimientos solidos en el lenguaje de programación Java Script.
* **Ingeniero Senior** automatizador que tenga conocimientos solidos en construcción de casos de uso y escenario de la validación sobre GUI, al igual que conocimiento en herramientas construidas sobre sintaxis Gherkin y librerías utilizadas para este fin.

### INTEGRACIÓN DE PRUEBAS DE EXTREMO A EXTREMO

* **Ingeniero Senior** con conocimientos en infraestructura en la nube en Amazon para colocar los elementos de generación de datos aleatorios, a priori y dinámicos, así como conocimientos solidos en BDD y generación de automatización por medio de Apis.

### AUTOMATIZACIÓN DE PRUEBAS DE REGRESIÓN

* **Ingeniero Senior** con conocimientos en automatización y nube para ejecutar pruebas VRT contará una distribución diaria de **8** horas para realizar las pruebas manuales dentro del alcance, todo para un acumulado de **40** horas dentro de plan de ejecución.

## **RECUSOS COMPUTACIONALES**

### **PRUEBAS MANUALES**

* Equipo Core i3. 8gb de RAM

### **AUTOMATIZACIÓN DE COMPONENTES**

* Cuenta de AWS para realizar la colocación de componentes pequeños para evaluar funciones específicas en el caso de esta sección, por lo cual requerimos elementos de Amazon tales como:
  + Lambdas
  + Ecs
  + Ecrs
  + Cloudwatch

### **INTEGRACION DE PRUEBAS DE EXTREMO A EXTREMO**

* Cuenta de AWS (Amazon Linux 64-bit x86) ***Para colocar los componentes de generación de data y la ejecución de las pruebas e2e sobre la data sintética generada.*** 
  + Aws Kubernetes
  + Ecs - Ecrs
  + Gateway
  + CloudWatch
  + Lambdas
  + NLB – ALB

### **PRUEBAS DE REGRESION AUTOMATIZADAS**

* Cuenta de AWS ***Para ejecutar scripts de automatización de pruebas e2e y resemblejs.*** 
  + Amazon Linux 64-bit x86
  + Aws Kubernetes
  + Ecs - Ecrs
  + Gateway
  + CloudWatch
  + Lambdas
  + NLB - ALB

## **PRESUPUESTO**

#### **PRUEBAS MANUALES**

**600** USD para la contratación de un ingeniero Senior para la ejecución de las pruebas manuales siguiendo los escenarios de pruebas planteados con anterioridad.

#### **PRUEBAS AUTOMATIZADAS EXPLORATORIAS**

**800** USD para la contratación de un ingeniero junior para la configuración y ejecución en ambiente nube de las pruebas aleatorias automatizadas con las herramientas y scripts planteados con anterioridad.

#### **AUTOMATIZACIÓN DE COMPONENTES**

**1250** USD para la contratación de un ingeniero Junior y un ingeniero Senior para la elaboración de las características técnicas de las Apis para evaluar los componentes y disponibilidad de una cuenta Amazon para poder desplegar las funcionalidades en dicha cuenta y lanzar la ejecución de las pruebas sobre componentes en total para un uso de (40 horas /persona) por cada sprint que se requiera.

#### **INTEGRACIÓN DE PRUEBAS DE EXTREMO A EXTREMO**

**675** USD para la contratación de un ingeniero Senior que construya las funciones que realicen la generación de data sintética, y posteriormente construir las funciones que realicen la ejecución de las pruebas e2e basadas en la data generada, el presupuesto dado se plantea por cada sprint que se requiera en (40 horas/persona) por semana.

#### **PRUEBAS AUTOMATIZADAS DE REGRESION VRT**

**1000** USD para la contratación de un ingeniero senior para la configuración y ejecución en ambiente nube de las pruebas de regresión visuales automatizadas con las herramientas y scripts planteados con anterioridad.

## **TNT**

### **PRUEBAS MANUALES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nivel** | **Tipo** | **Técnica** | **Objetivo** |
| Sistema | Positivas | Manuales – Definidas | Evidenciar el comportamiento del sistema en pruebas de QA en escenario exitoso |
| Sistema | Negativas | Manuales - Definidas | Identificar puntos de mejora en el comportamiento del sistema en pruebas de QA en escenario negativos |
| Unidad | Funcional | Manuales - Definidas | Documentar las pruebas unitarias que posee el sistema. |

### **AUTOMATIZACIÓN DE PRUEBAS EXPLORATORIAS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nivel** | **Tipo** | **Técnica** | **Objetivo** |
| Integración | Funcionales | Automatizadas | Lograr desarrollar y ejecutar los scripts con pruebas aleatorias automatizadas y de exploración sistémica. |
| Integración | No Funcionales | Automatizadas | Lograr desarrollar y ejecutar los scripts con pruebas aleatorias automatizadas y de exploración sistémica. |

### **AUTOMATIZACIÓN DE COMPONENTES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nivel** | **Tipo** | **Técnica** | **Objetivo** |
| Integración | Negativos | Automatizadas | Lograr desarrollar y ejecutar los escenarios planteados, automatizando casos erróneos. |
| Integración | Positivos | Automatizadas | Identificar y desarrollar los casos de escenarios positivos para cada uno de los componentes segregados de la aplicación. |

### **INTEGRACIÓN DE PRUEBAS DE EXTREMO A EXTREMO**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nivel** | **Tipo** | **Técnica** | **Objetivo** |
| Aceptación | Funcionales | Automáticas | Lograr implementar pruebas e2e para la ejecución completa de los flujos definidos basados en la data generada. |
| Aceptación | No Funcionales | Automáticas | Identificar la falta de validaciones en la aplicación por medio de las pruebas e2e y la generación de data aleatoria. |

### **INTEGRACIÓN DE AUTOMATIZACIÓN DE PRUEBAS DE REGRESION**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nivel** | **Tipo** | **Técnica** | **Objetivo** |
| Aceptación | Funcionales | Automáticas | Implementar la automatización de pruebas regresión e2e a través de VRT. |
| Aceptación | No Funcionales | Automáticas | Implementar la automatización de pruebas regresión e2e a través de VRT. |

## **DISTRIBUCION DEL ESFUERZO**

#### **PRUEBAS MANUALES**

**Semana 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Funcionalidad Home | Funcionalidad Post | Funcionalidad Pages | Funcionalidad Tags | Funcionalidad Staff |
| **Nocturno** | N/A | N/A | N/A | N/A | N/A |

**Semana 2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Funcionalidad Configuraciones Generales | Funcionalidad identidad de publicaciones | Funcionalidad Diseño | Funcionalidad Plugins | Funcionalidad Características experimentales |
| **Nocturno** | N/A | N/A | N/A | N/A | N/A |

### **AUTOMATIZACIÓN DE PRUEBAS EXPLORATORIAS**

**Semana 1 -3**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Ejecución de scripts explorando funcionalidad Home | Ejecución de scripts explorando funcionalidad Post | Ejecución de scripts explorando funcionalidad Pages | Ejecución de scripts explorando funcionalidad Tags | Ejecución de scripts explorando funcionalidad Staff |
| **Nocturno** | N/A | N/A | N/A | N/A | N/A |

**Semana 3 - 8**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Ejecución de scripts explorando funcionalidad Configuraciones Generales | Ejecución de scripts explorando funcionalidad identidad de publicaciones | Ejecución de scripts explorando funcionalidad Diseño | Ejecución de scripts explorando funcionalidad Plugins | Ejecución de scripts explorando funcionalidad Características experimentales |
| **Nocturno** | N/A | N/A | N/A | N/A | N/A |

### **AUTOMATIZACIÓN DE COMPONENTES**

**Semana 1 -3**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Creación de pruebas Login | Creación de pruebas Staff | Creación de pruebas Post | Creación de pruebas Pages | Creación de pruebas Tags |
| **Nocturno** | Ejecución de pruebas Login | Ejecución de pruebas Staff | Ejecución de pruebas Post | Ejecución de pruebas Pages | Ejecución de pruebas Tags |

### **INTEGRACIÓN DE PRUEBAS DE EXTREMO A EXTREMO**

**Semana 3 - 5**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Planeación de las funcionalidades | Desarrollo de los escenarios planeados con las API | Construcción y refactorización del código entregado | Seguimiento de las ejecuciones y construcción de nuevos casos | Seguimiento de las ejecuciones y construcción de nuevos casos |
| **Nocturno** |  |  | Ejecución de pruebas e2e y generación de data\* | Ejecución de pruebas e2e y generación de data\* | Ejecución de pruebas e2e y generación de data\* |

**Semana 5 - 8**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Planeación de las funcionalidades | Desarrollo de los escenarios planeados con las API | Construcción y refactorización del código entregado | Seguimiento de las ejecuciones y construcción de nuevos casos | Seguimiento de las ejecuciones y construcción de nuevos casos |
| **Nocturno** |  |  | Ejecución de pruebas e2e y generación de data\* | Ejecución de pruebas e2e y generación de data\* | Ejecución de pruebas e2e y generación de data\* |

### **AUTOMATIZACIÓN DE PRUEBAS DE REGRESIÓN VTR**

**Semana 5 - 8**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Ejecución de scripts explorando funcionalidad Home | Ejecución de scripts explorando f uncionalidad Post | Ejecución de scripts explorando f uncionalidad Pages | Ejecución de scripts explorando f uncionalidad Tags | Ejecución de scripts explorando f uncionalidad Staff |
| **Nocturno** | N/A | N/A | N/A | N/A | N/A |

**Semana 5-8**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Horario** | **Día 1** | **Día 2** | **Día 3** | **Día 4** | **Día 5** |
| **Diurno** | Ejecución de scripts explorando funcionalidad Configuraciones Generales | Ejecución de scripts explorando funcionalidad identidad de publicaciones | Ejecución de scripts explorando funcionalidad Diseño | Ejecución de scripts explorando funcionalidad Plugins | Ejecución de scripts explorando funcionalidad Características experimentales |
| **Nocturno** | N/A | N/A | N/A | N/A | N/A |

1. La estrategia esta definida en el readme del [proyecto](https://github.com/jandresboyaca/TestingAutomation#estrategia-de-datos) [↑](#footnote-ref-2)
2. Formato de pruebas suministrado en el readme del [proyecto](https://github.com/jandresboyaca/TestingAutomation/blob/main/exploratory-inventory-testing.xlsx) [↑](#footnote-ref-3)