**3124289373.**

**1)Introducción.**

Este plan de pruebas está diseñado para definir los límites, enfoque, recursos y el calendario de las actividades de prueba del proyecto **De todo Para Todos.**

El plan de testing identifica los elementos que se van a probar, las características que se van a probar, los tipos de pruebas que se deben realizar, el personal responsable de las pruebas, los recursos y el calendario que son necesarios para completar las pruebas y evaluar los riesgos asociados con el plan.

**1.1) Ámbito de la aplicación.**

**1.1.1) Alcance de la aplicación**

Por definir(dependiente del equipo de desarrollo)

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Nombre | Testeado como | Descripción |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**1.1.2) Fuera de alcance del testing de la aplicación**

Teniendo en cuenta la proximidad de la fecha límite de entrega del proyecto y el hecho de que los equipos están trabajando en paralelo y es necesario tener entregables de los demás equipos involucrados en el desarrollo para poder desarrollar el sistema de testing y contando con que el equipo de trabajo se ciña a las buenas prácticas se toma la decisión de excluir de testeo las siguientes características :

* Lógica de bases de datos
* Integridad de bases de datos
* interfaces de comunicaciones
* Rendimiento del sitio web

**1.2)Objetivo de calidad**

El objetivo de las pruebas descritas en este documento se centra en la verificación de la óptima funcionalidad de la aplicación web De Todo Para Todos.

**1.3)Funciones y responsabilidades (Equipo)**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Roles** | **Responsabilidades** |
| Oscar Felipe Palacios | Test Manager  Test Team | * Es responsable de supervisar el proceso de prueba, la revisión y la coordinación de la aceptación. * Es responsable de probar o ayudar a probar la aplicación web |
| Juan Pablo Osorio | Test Team | * Es responsable de probar o ayudar a probar la aplicación web |
| William Felipe Carreño | Test Team | * Es responsable de probar o ayudar a probar la aplicación web |

**2)Metodología de prueba**

**2.1) Visión General**

Teniendo en cuenta que el proyecto se está desarrollando implementando una metodología de desarrollo ágil, es necesario que el testing se acople a esta metodología de trabajo para detectar los posibles fallos críticos a tiempo y que estos no retrasen el cronograma de actividades propuesto por el equipo de desarrollo, por ello este equipo se regirá bajo los siguientes principios:

* **El testing no es una fase:** El testing continuo es la única forma de garantizar un avance continuo, por esto, el testing se realiza continuamente junto con el desarrollo de software y demás actividades.
* **Todo el equipo realiza pruebas:** Para poder satisfacer el principio anterior es necesario que cada integrante del equipo de testing esté haciendo constantemente pruebas y redactando los documentos para nutrir el banco de documentos del plan de testing.
* **Reducir el tiempo para recibir retroalimentación:** El equipo de testing deberá retornar en el menor tiempo posible los resultados de las pruebas realizadas al equipo que emitió el elemento que fue testeado.
* **Reducir la documentación de pruebas:** El equipo de testing deberá hacer en lo posible que los documentos agregados al banco de pruebas sean concretos para evitar un desborde de documentos y deben estar diseñados para hacerle mas facil la tarea al equipo de desarrollo.

**2.2) Tipos de pruebas**

De manera general, lo primero que debemos tener en cuenta es que existen pruebas de software manuales y pruebas de software automatizadas.

**Las pruebas manuales** son llevadas a cabo por personas, quienes navegan e interactúan con el software (usando herramientas adecuadas para cada caso).

Estas pruebas resultan costosas, ya que se requiere contar con un profesional encargado de esta labor; para configurar un entorno y así mismo ejecutar las pruebas.

Como es de esperarse, estas pruebas están expuestas a errores humanos: por ejemplo, se pueden cometer errores tipográficos u omitir pasos durante la prueba.

**Las pruebas automatizadas**, por el contrario, son realizadas por máquinas, que ejecutan un "test script" que ya ha sido escrito previamente.

Estos tests pueden variar mucho en cuanto a complejidad:

desde verificar que el método de una clase específica funcione correctamente,

hasta asegurar que una secuencia de acciones complejas en la UI se lleve a cabo correctamente y devuelvan los resultados esperados.

Estas pruebas son más rápidas y confiables que las que se llevan a cabo manualmente – pero la calidad de estas pruebas automatizadas depende de qué tan bien escritos se encuentren los "tests scripts".

Los siguientes tipos de testing son los que pueden ser aplicados a la aplicación web De Todo Para Todos.

* **User Research** (investigación de usuarios) para comprender en profundidad la problemática a solucionar desde la aplicación web.
* **UX Testing** (testeo de la experiencia de los usuarios) para evaluar si la experiencia del usuario es la óptima.
* **Unit Tests**(pruebas unitarias ) para probar de forma individual las funciones y métodos.
* **Integration Tests**(pruebas de integración) para verificar que los diferentes módulos y/o servicios usados por nuestra aplicación funcionen en armonía cuando trabajan en conjunto.
* **End-to-end tests**(pruebas de punta a punta) para replicar el comportamiento de los usuarios con el software, en un entorno de aplicación completo.
* **Smoke testing(**pruebas de humo**)** para verificar la funcionalidad básica de una aplicación.
* **Acceptance testing**(pruebas de aceptación) Estas pruebas requieren que el software se encuentre en funcionamiento, y se centran en replicar el comportamiento de los usuarios, a fin de rechazar cambios si no se cumplen los objetivos. Estos objetivos pueden ir más allá de obtener una respuesta específica, y medir el rendimiento del sistema.
* **SQL injection**(Inyección SQL)para verificar que los posibles vectores ejecutables esten bloqueados
* **Penetration Testing**(Pruebas de penetración) Para encontrar posibles fallos de seguridad que puedan comprometer la información de los usuarios o el funcionamiento de la aplicación web

**2.3) Triage de errores**

Los errores encontrados eran evaluados para determinar la prioridad de solución bajo los siguientes criterios a los cuales se les dara un numero del 1 al 5.

* **Validez:** ¿Es realmente un error? o **¿**es solamente un comportamiento inesperado?
* **Gravedad:** cuando ocurre este error, ¿qué tan negativo es el impacto?
* **Frecuencia:** ¿con qué frecuencia ocurre este error o bug?
* **Costo** : ¿cuánto esfuerzo se requeriría para solucionar este bug?
* **Riesgo** : ¿Cuál es el riesgo de corregir este error?”

**2.4)Criterios de suspensión y requisitos de reanudación**

Si los miembros del equipo informan que hay un 50% de los casos de prueba fallidos, se suspenderán las pruebas hasta que el equipo de desarrollo arregle todos los casos fallidos.

**2.5)Integridad de las pruebas**

La tasa de aprobación (80%) es obligatoria para mantener la integridad de las pruebas, pero por motivos de seguridad de la aplicación web los elementos calificados en el triage de errores como un error de gravedad alta deben tener una aprobación del 100% al momento de evaluar el parche aplicado.

**2.6) Tareas y estimación del proyecto y calendario**

Por definir (dependiente de los equipos de desarrollo y diseño)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tarea** | **Ejecutada por** | **Tiempo estimado de ejecución** | **Validada por** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**3)Entregables**

Los entregables se proporcionarán de la siguiente manera:

**Antes de la fase del testing:**

* Documento de planes de prueba.
* Documentos de casos de prueba
* Especificaciones de diseño de prueba.

**Durante el testing:**

* Simuladores de herramientas de prueba.
* Datos de prueba
* Test Trace-ability Matrix - Registros de errores y registros de ejecución.

**Después del testing:**

* Resultados/informes de la prueba
* Informe de defectos
* Directrices de procedimientos de instalación /

**4)Herramientas y entornos de prueba**

4.1)Herramientas de prueba(Físicas)

|  |  |
| --- | --- |
| **Recurso** | **Descripción** |
| Servidor | Computador UNIX configurado con LAMP |
| Celulares | 4 dispositivos móviles con diferente configuración de recursos |
| Computadores | 4 computadores para probar el comportamiento en los sistemas operativos más utilizados (MacOS,Linux,Windows) |

**4.2) Herramientas de prueba (Software)**

Las herramientas aquí descritas pueden mutar de acuerdo con las necesidades del proceso

|  |  |  |
| --- | --- | --- |
| **Herramienta** | **Descripción** | **Versión** |
| Maze | Maze es una plataforma de user research que le permite recopilar datos de usabilidad cualitativos y cuantitativos, todo en un solo lugar. Maze se integra directamente con Figma, InVision, Marvel o Sketch, y te permite importar un prototipo existente desde la herramienta de diseño que utilices. | N/A |
| PHPUnit | PHPUnit es un framework para PHP que facilita la creación de clases de pruebas sobre aplicaciones basadas en PHP. PHP es un lenguaje que posibilita la creación de páginas web complejas, lo que provoca la necesidad de controlar el correcto funcionamiento de las mismas. |  |
| Codeception | es una herramienta que cubre los tres tipos de test, unitarios, funcionales y de aceptación. Está desarrollada sobre PHPUnit, por lo que mantiene todas las funcionalidades y compatibilidades con él. Al estar desarrollado sobre PHPUnit, Codeception añade algunos procedimientos a problemas comunes en los distintos tipos de test unitarios |  |
| Storyplayer | Es un framework realizar pruebas End to end para PHP es de código abierto que se utiliza para realizar pruebas de extremo a extremo. Es el más adecuado para las pruebas de automatización de la API y las pruebas de las aplicaciones web. |  |
| SeleniumHQ | Es un entorno de pruebas de software para aplicaciones basadas en la web. Selenium provee una herramienta de grabar/reproducir para crear pruebas sin usar un lenguaje de scripting para pruebas (**Selenium IDE**). |  |
| Phpspec | Es un framework que toma el enfoque de desarrollo de pruebas por comportamiento. Usando PHPSpec puedes describir cómo se debe comportar el código de la aplicación antes de realizar las pruebas. |  |
| Phpstorm | PhpStorm es el IDE PHP «Lightning Smart» creado por JetBrains y el cual está disponible para desarrolladores en Linux, MacOS y Windows. |  |
| BeEF | BeEF es la abreviatura de The Browser Exploitation Framework. Es una herramienta de prueba de penetración que se centra en el navegador web. |  |
| W3af | w3af (Web Application Attack and Audit Framework) es una herramienta open source de auditoría que permite detectar vulnerabilidades web y explotarlas |  |
| PhantomJS | es un navegador que se utiliza para automatizar las interacciones de la página con fines de prueba. Ayuda a los usuarios a habilitar la navegación y el comportamiento del usuario en una página sin cargar la interfaz gráfica. PhantomJS imita y manipula una página web para llevar a cabo la automatización de pruebas que en última instancia, ahorra una tremenda cantidad de tiempo para los probadores. |  |
| Postman | es una gran herramienta para probar APIs. Los probadores y desarrolladores pueden utilizar esta herramienta gratuita como una extensión de Chrome o un producto de colaboración en la nube para desarrollar, probar y documentar las API más rápidamente. Permite a los usuarios comprobar el historial de las solicitudes HTTP enviadas, personalizar secuencias de comandos, autocompletar URL, previsualizar imágenes, realizar pruebas de producción, organizar o configuraciones locales con una amplia gama de características y funciones. |  |
| DataGrip | Es un entorno de gestión de bases de datos para desarrolladores. Está diseñado para consultar, crear y gestionar bases de datos. Las bases de datos pueden funcionar localmente, en un servidor o en la nube. Es compatible con MySQL, PostgreSQL, Microsoft SQL Server, Oracle y otros. |  |