
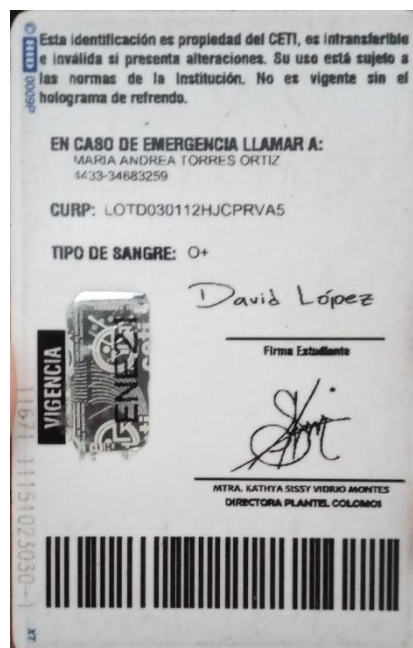
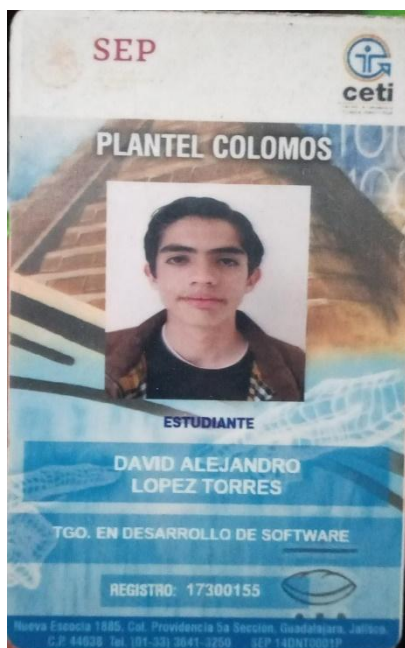


**EVIDENCIA DE ACTIVIDADES EN CLASE**

<b>Nombre del alumno:</b>	David Alejandro López Torres					
<b>Registro</b>	17300155					
<b>Carrera:</b>	Desarrollo de Software					
<b>Materia:</b>	Seguridad en Software					
<b>Clave:</b>	MPF3308DSO	<b>Grupo</b>	8D1	<b>Semestre:</b>	8	
<b>Profesor:</b>	Ing. Luis René Duran Hernández.					

**Foto de credencial:**



## Propósito de la actividad

### Planificación de pruebas

Hacer un análisis de riesgo y elaborar una estimación de pruebas.

## Practica lo que aprendiste I

- I. En la siguiente tabla identifica el nivel de prioridad que tendría cada una de las condiciones y argumenta tu respuesta. (aplicado al proyecto seleccionado)

Descripción	Prioridad			Argumento
	Alto	Medio	Bajo	
No se cuenta con los testers suficientes para realizar las pruebas en el tiempo establecido.				La gran cantidad de ecuaciones que debe resolver la aplicación requiere de la mayor cantidad de pruebas posibles
El equipo de pruebas está conformado en su mayoría por testers con poca experiencia.				No se requiere demasiada experiencia para probar al sistema: solo requiere tomar una foto a un texto y comparar lo que devuelve el sistema con lo esperado
El ambiente de pruebas no está configurado adecuadamente.				Existen 3 maneras de interactuar con el sistema para realizar pruebas, en donde cada uno de ellos es bastante flexible, por lo que aún en condiciones adversas se espera encontrar una ruta para realizar las pruebas
La aplicación tiene problemas de seguridad.				La aplicación en sí no guarda valores muy importantes de los usuarios (solo su cuenta y contraseña, que se utiliza para el historial de consultas de ecuaciones). Sin embargo, la posibilidad de afectar código y

				con ello el funcionamiento podría ser riesgoso.
El presupuesto para la ejecución de las pruebas es menor al que debería ser.				El proyecto está sentado en las bases del desarrollo Open Source, aunque algunas implementaciones de APIs como Google Vision requieren un pequeño aporte (200 pesos mensuales en un mal caso). De esta forma, una salida de la partida presupuestal sería complicada pero solventable por el equipo.
Unos de los inversionistas retira su apoyo al proyecto, por lo cual el equipo de prueba sufre recortes de presupuesto.				Si por alguna razón las tecnologías que se utilizan de manresa Open Source cambian abruptamente a cobrar una cuota por servicio el proyecto se ve en apuros debido a que el proyecto en sí se basa en la interacción de todas estas herramientas: si una se bloquea, la aplicación no puede funcionar

## II. Analiza el siguiente caso.

**La fase de desarrollo de la aplicación que fue encargada a tu compañía ha terminado. El equipo seleccionado para realizar las pruebas de la aplicación consta de 10 personas, 6 de ellas experimentadas y 4 sin experiencia.**

**El líder del proyecto informa que las pruebas deben realizarse en 20 días hábiles y el presupuesto inicial se recortó 60%.**

Argumenta:

**> ¿Está bien organizado el equipo para cumplir con el objetivo?**

Lo está pues se permite que nuevos empleados se integren, estos nuevos empleados trabajan bajo la tutela de los empleados que tienen experiencia, por lo que tienen oportunidad de aprender.

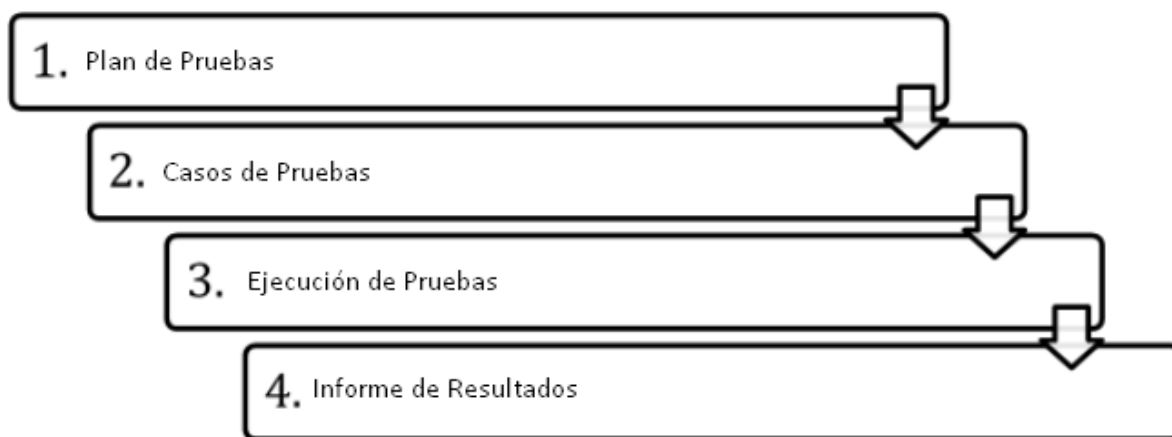
**> ¿Cómo solucionarías el problema del recorte de presupuesto?**

Es necesario cambiar la propuesta, un recorte del 60% representa más de la mitad de lo inicialmente previsto, por ello es necesario replantear el problema y buscar una nueva solución que se adapte a las nuevas circunstancias.

> ¿Tienes los recursos necesarios para cumplir con el objetivo?

Si la meta u objetivos no se vuelven plantear, sería muy difícil para el equipo cumplir con lo acordado pues el límite del tiempo más el cambio en las condiciones iniciales, cambian toda la planeación. Por tanto es muy seguro que no los tengan.

III. **Anota los principales puntos que integran una estimación de pruebas y añade la descripción de cada uno.**



**Plan de pruebas:** Es un producto formal que define los objetivos de la prueba de un sistema, establece y coordina una estrategia de trabajo, y provee del marco adecuado para elaborar una planificación paso a paso de las actividades de prueba.

**Casos de pruebas (Test case):** Son un conjunto de condiciones o variables bajo las cuales un analista determinará si una aplicación, un sistema software (software system), o una característica de éstos es parcial o completamente satisfactoria.

**Ejecución de pruebas:** Implica iniciar con la creación de los datos de prueba necesarios para ejecutar los casos de prueba diseñados. La ejecución de estos casos, puede realizarse de manera manual o automatizada; en cualquiera de los casos, cuando se detecte un fallo en el sistema, este debe ser documentado y registrado en una herramienta que permita gestionar los defectos.

**Informe y resultados:** Representa los resultados obtenidos al probar la aplicación final (después de la integración de los componentes). Este resultado debe coincidir con el objetivo principal propuesto por el aplicativo porque no necesariamente por haber certificado todos los componentes que componen el aplicativo esto se traduce en un correcto funcionamiento después de la integración de componentes.

IV. **Utiliza los valores de la siguiente tabla para realizar la estimación de las pruebas de una aplicación de banca en línea.**

Actividad	Escenario óptimo	Escenario probable	Peor escenario
Creación de un nuevo cliente	1	3	4
Creación de una nueva cuenta	2	3	6
Verificar el acceso correcto a cuenta	1	2	4
El cliente puede revisar su estado de cuenta	2	3	5
El cliente puede realizar transferencias	2	5	7
El cliente puede cambiar su contraseña	0.5	1	2
Eliminación de una cuenta	0.2	1	4
Eliminación de un cliente	0.2	1	3
El cliente puede imprimir su estado de cuenta	0.5	1	3

**Escenario Óptimo: (0.0 - 9.4)** En este escenario las pruebas resultan exitosas y las acciones a probar se concluyen en tiempo de respuesta cortos, de modo que se adquiere la certeza de que los usuarios finales de la aplicación tendrán una experiencia satisfactoria al utilizar la aplicación de banca en línea. Además, la buena respuesta por parte de los ejercicios de prueba da pie a realizar futuras pruebas sin necesidad de detenerse a corregir o modificar la aplicación para ajustarla a las fallas que podrían haberse encontrado.

**Escenario probable: (9.4 - 20.0)** En este escenario la mayoría de las pruebas resultan exitosas, pero algunas de ellas se salen de los parámetros de tiempo establecidos como mínimo de eficiencia. Para resolver estos problemas de eficiencia es posible que se requiere extender la partida presupuestal y el tiempo de desarrollo; no obstante, podemos decir que un porcentaje elevado de la aplicación funciona adecuadamente y por eso se podría plantar el lanzamiento de una versión beta y seguir trabajando en el desarrollo de una versión principal asegurándose en realizar las mismas pruebas para identificar las mejoras realizadas con las nuevas implementaciones.

**Peor escenario: (20.0 - 38.0)** En este escenario muy pocas pruebas resultaron exitosas y aquellas que lo hicieron quedaron fuera de los parámetros de calidad establecidos en los requerimientos de la aplicación. La aplicación no está lista para ser lanzada al mercado y requiere de una reestructuración

en el desarrollo para resolver los problemas expuestos por las pruebas. Se requiere de una nueva planeación de desarrollo y crear una nueva partida presupuestal que considere las modificaciones pertinentes para que el proyecto entre en el rango de escenario probable u óptimo.

## Practica lo que aprendiste II

### I. Elabora un plan de pruebas completo

> Elige una aplicación con la cual estés familiarizado. Si puedes acceder a la información técnica, será mucho mejor.

> Lee el siguiente planteamiento y responde lo que se pide.

**La fase de desarrollo de una aplicación fue encargada a tu compañía. El equipo elegido para realizar las pruebas de la aplicación consta de 8 personas, 6 de ellas experimentadas y 2 sin experiencia.**

**El líder del proyecto informa que las pruebas deben realizarse en 30 días hábiles, que se cuenta con un presupuesto de \$80.000, y que el cliente indicó que deben cubrirse tanto las características funcionales como el rendimiento de la aplicación.**

#### a) Análisis de producto

Con base en la aplicación que hayas elegido, responde lo siguiente:

#### > ¿Quién lo usará?

Durante las etapas de pruebas los teasers, tanto los experimentados (6), como los nuevos (2). En una etapa posterior, la utilizarán estudiantes de universidad de carreras de ingenierías o ciencias exactas; así como docentes e investigadores de las mismas áreas.

#### > ¿Para qué sirve?

Se trata de una PWA para resolver ecuaciones diferenciales capturadas por una imagen, texto en pantalla o por entrada en teclado.

#### > ¿Cómo funciona?

Está distribuida en dos partes: cliente y servidor. El cliente está basado en el modelo de desarrollo vista-controlador implementado para el desarrollo de una PWA por medio de código Python. El servidor estará alojado de manera remota y será por medio de solicitudes HTTP (con cuerpo JSON) que el servidor recibe la solicitud de resolver una determinada ecuación diferencial y retorna la respuesta a la aplicación cliente que solicitó resolver dicha ecuación. Para resolver la ecuación, el servidor cuenta con una serie de métodos predefinidos dependiendo del tipo de ecuación diferencial y posee un soporte simbólico por el módulo SumPy de Python.

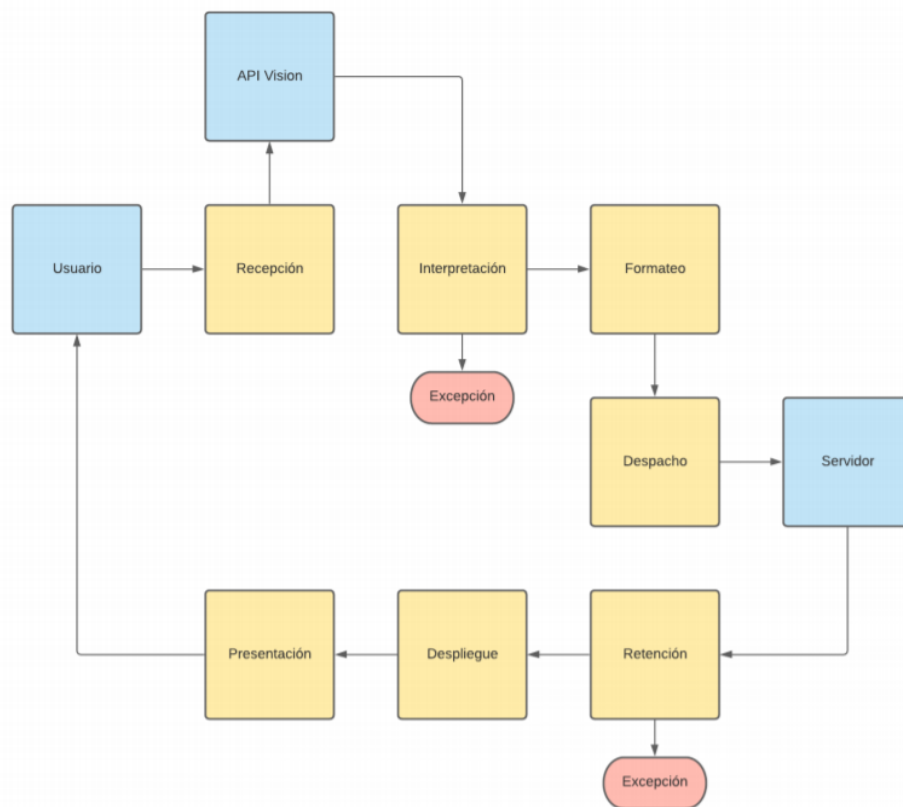
#### > ¿Qué requisitos tiene para su funcionamiento?

Requiere de una buena conexión a internet por parte del usuario que utilice la aplicación cliente para lanzar solicitudes hacia el servidor. Además, el servidor requiere estar siempre activo y tener una respuesta rápida para gestionar múltiples solicitudes en tiempos que le permitan al usuario tener una experiencia cómoda de uso; esto es, se requiere que el hosteador del servidor tenga un desempeño

dentro de lo establecido en la calidad de respuesta. Al ser una PWA se espera alcanzar una compatibilidad muy alta con una gran gama de dispositivos móviles y de escritorio, de modo que no se solicita un Hardware en especial para utilizar la aplicación cliente por parte del usuario. Debido a que la mayoría de las herramientas a utilizar para su desarrollo y posterior implementación se basan en código Open Source, no se tiene una lista larga de requisitos para garantizar la funcionalidad de la aplicación.

### > Documentación del producto

Podemos resumir la documentación con base a lo expresado en el Documento de Especificación de Requerimientos (DER) presentado a los asesores de proyecto:



La distribución del diagrama modular de la **aplicación cliente** queda como se muestra en el esquema de arriba. Podemos resumir la operación de cada uno de los módulos como sigue:

**I. Módulo de recepción:** Este módulo se encarga de recibir la información del usuario (por cualquiera de los medios de entrada permitidos) y guardar esta información para su posterior interpretación.

**II. Módulo de interpretación:** Este módulo se encarga de traducir la información contenida en la entrada (que puede ser vía imagen, texto en pantalla o texto en un formato matemático) a una entidad informática que puede ser clasificada como un tipo de ecuación diferencial según sea el caso. Se encarga de validar que la entrada pueda ser interpretada como una ecuación diferencial.

**III. Módulo de formateo:** Este módulo se encarga de generar un formato a la entrada con la intención de que pueda ser interpretada por el servidor de manera adecuada. Junto con el módulo de despliegue representan los “traductores” entre una cadena de texto ordinaria y una expresión matemática. Se pretende que el formato utilizado sea Math LaTeX debido a su extensivo uso, las comodidades de operación que ofrece y la gran cantidad de documentación que existe apoyándose de expresiones en este formato.

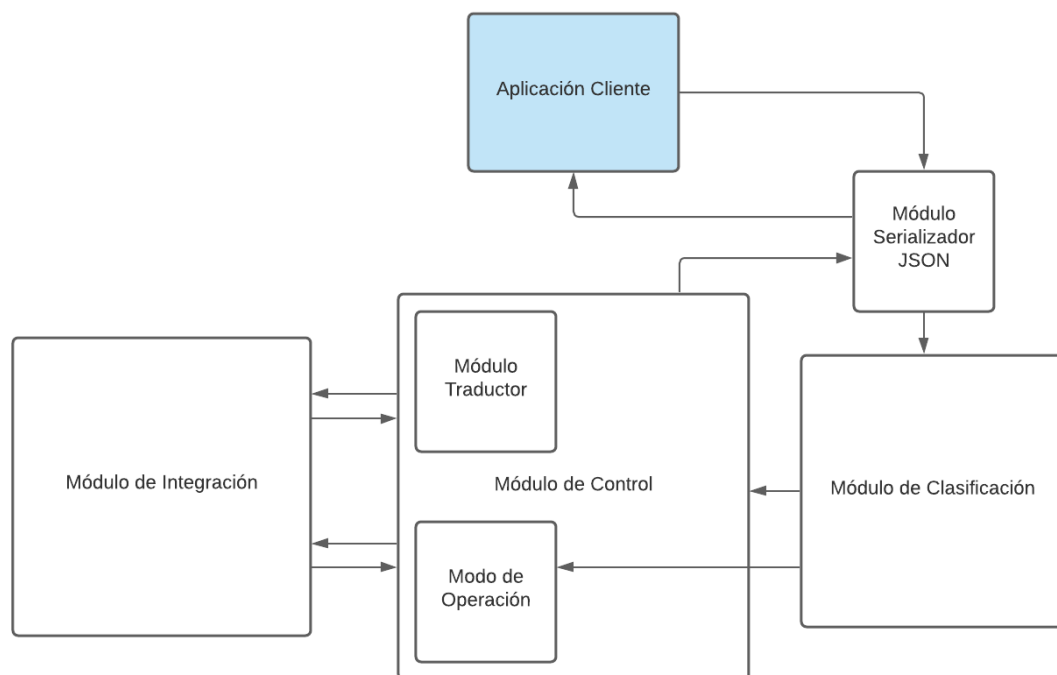
**IV. Módulo de despacho:** Este módulo se encarga de enviar una solicitud al servidor seleccionado para resolver la ecuación diferencial detectada con la información pertinente para que pueda ser interpretado en el servidor. Se encarga también de codificar la ecuación diferencial en un formato que el servidor pueda entender para darle solución.

**V. Módulo de retención:** Este módulo se encarga de retener la respuesta generada por el servidor de la petición, porque es en este módulo donde la respuesta a la ecuación diferencial entra al sistema. Su principal función es mantener la conexión con el servidor mientras genera y envía la solución de la ecuación diferencial hacia el sistema.

**VI. Módulo de despliegue:** Este módulo se encarga de expandir la respuesta obtenida desde el servidor en entidades informáticas que pueden ser manejadas en el sistema para generar una versión menos abstracta de la solución que la recibida por el servidor.

**VII. Módulo de presentación:** Este módulo se encarga de exponer al usuario la solución de la ecuación diferencial en una presentación entendible por él. Su función incluye el interpretar la solución obtenida en el módulo anterior en elementos visuales que formarán la solución mostrada al usuario.

La distribución modular del **servidor** es como se sigue:



**I. Módulo serializador:** Se encarga de procesar la solicitud de la ecuación del cliente, la cual viene en formato JSON. Guarda la información en objetos propios de SymPy para su futuro análisis. También es el encargado de encapsular la solución de la ecuación en formato JSON para ser emitida a la aplicación cliente.

**II. Módulo de clasificación:** Su función es determinar cuál es el tipo de ecuación diferencial que se pretende resolver. Utiliza definiciones matemáticas de cada uno de los tipos para llegar a una conclusión contundente de que se trata de un tipo en concreto.

**III. Módulo de control:** Se encarga de mantener un canal de conexión de datos entre el módulo de operación y el traductor. También es el encargado de regular las intervenciones de cada uno de los



módulos dentro del servidor y del manejo de las situaciones especiales (sobre flujos y truncamientos).

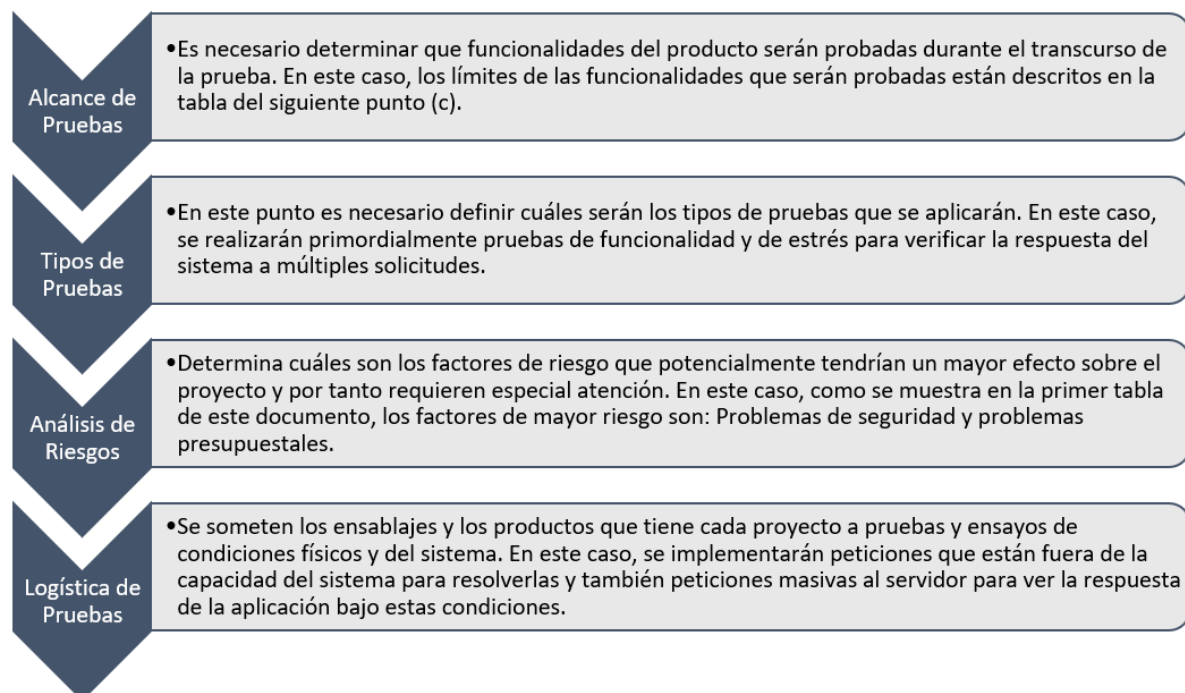
**IV. Módulo de operación:** Su función es determinar cuales son los pasos que siguen para solucionar la ecuación diferencial. Se basa en métodos específicos para atacar a cada tipo de ecuación diferencial en cuestión. Sus solicitudes de pasos 'complejos' (integración o Laplace) son enviadas al módulo de integración. Al terminar un paso, la información recolectada es enviada al módulo traductor.

**V. Módulo traductor:** Se encarga de representar los pasos realizados en estructuras de texto en formato LaTeX para su posterior serialización.

**VI. Módulo de integración:** Se encarga de resolver integrales indefinidas dentro del servidor y aplicar la transformada de Laplace así como su inversa. Posee una memoria propia que le permite evaluar la dificultad de los pasos que está desarrollando. También cuenta con una serie de definiciones matemáticas que le permiten resolver integrales que se componen de varios pasos.

### b) Estrategia de pruebas

Completa el siguiente esquema con la información que se debe incluir en cada fase.



### c) Definición de Objetivos de Prueba

Elabora una lista de todas las funciones de la aplicación a la que se le deben realizar las pruebas y las metas.

Funciones	Metas
1. Obtención de información del usuario por medio de un form.	Obtener la información del usuario, sin perder la integridad de los datos y guardarlos en una base de datos. Esta información constituye la cuenta del usuario.
2. Conversión del texto de una imagen a texto plano.	Obtener el texto de una imagen, idealmente el texto debe ser el mismo que de la imagen. De forma que podamos modificarlo y trabajar con él como texto plano.
3. Interpretación del texto plano a una ecuación definida.	El texto plano llevará una cierta sintaxis definida con la que reconstruiremos una ecuación matemática.
4. Conversión del texto plano a latex	Utilizamos el texto plano y la sintaxis definida para reconstruir la ecuación que interpretó el sistema en formato LaTeX.
5. Identificación de la respectiva ecuación diferencial	Una vez hemos obtenido la ecuación el sistema deberá matemáticamente identificar correctamente el tipo de ecuación diferencial que ha sido insertada.
6. Resolución de la ecuación diferencial	Tras determinarse el tipo de ecuación diferencial, el sistema deberá ser capaz de resolver la ecuación diferencial de forma algebraica siguiendo los pasos usuales como si fuera contestada "a lapiz". Así obtener una solución.
7. Obtención de la respuesta del módulo de resolución de la ecuación diferencial	La respuesta es regresar del módulo de solución a la aplicación, la comunicación se da de forma correcta entre ambos módulos.
8. La respuesta se imprime en la plataforma en formato LaTeX.	La respuesta regresada se desempaqueta y es tratada de forma adecuada para la presentación de los pasos y de la solución de la ecuación diferencial. Dejándola en el formato LaTeX correspondiente.

9. Se agrega dicha ecuación al historial del usuario	Actualizar la base de datos de forma correcta de forma que el historial del usuario incluya la ecuación diferencial que recién pidió.
--	---

**d) Criterios de prueba**

En la siguiente tabla anota los criterios que seleccionaste para las pruebas.

<b>Criterio de suspensión</b>	<b>Argumento</b>	<b>Criterio de salida (reanudación)</b>	<b>Argumento</b>
Se perdió la conexión a Internet por parte del equipo de testers	Sin una conexión a internet es imposible desarrollar las pruebas de funcionalidad del proyecto	Regresa la conexión de Internet al equipo de testers	Una vez restablecida la conexión a Internet del equipo de testers no existe impedimento para continuar con las pruebas de funcionalidad
Se perdió la conexión con el servidor (o fue rechazada).	Sin una comunicación con el servidor por parte de la aplicación cliente es imposible desarrollar las pruebas de funcionalidad del proyecto.	Se restablece la conexión con el servidor por medio de una serie de modificaciones en el servidor	Una vez restablecida la conexión con el servidor por parte del equipo de testers no existe impedimento para continuar con las pruebas de funcionalidad.
Se registró una cantidad de 5 respuestas erróneas por parte del servidor para diferentes ecuaciones (diferentes tipos)	Tener un registro de malas respuestas por parte del servidor de esa magnitud es un indicador de que hay errores de codificación contundentes desde el lado del servidor que deben ser corregidos antes de	Una vez corregidos los errores del servidor, se realiza una prueba rápida con las mismas 5 ecuaciones que fallaron al poner en suspensión las pruebas. Si es favorable en las 5, se	Obteniendo una correcta respuesta correcta para 5 tipos de ecuaciones diferentes es un indicador de que el servidor está realizando las operaciones correspondientes de manera adecuada y

	perder más tiempo en pruebas	continúa con las pruebas	está listo para continuar con las pruebas de funcionalidad
Se detectaron fallas en la seguridad de la aplicación	No se puede continuar con la ejecución de pruebas si es posible que el código no esté asegurado así como la integridad de los datos de los usuarios, aún en la etapa de pruebas de funcionalidad	La falla de seguridad que fue reportada fue correctamente atendida	Una vez restablecida la seguridad del código y de la información de la aplicación es posible continuar con las pruebas de funcionalidad
Se tiene un grupo muy limitado de testers disponibles en algún momento	Las pruebas de funcionalidad de la aplicación requieren de un equipo grande de testers para abordar las diferentes funcionalidades y factores que se deben probar antes del lanzamiento de la aplicación	Se reunió un grupo con una cantidad de testers adecuada (6)	Con un grupo de ese tamaño es posible emular diferentes circunstancias para el desarrollo de las pruebas de estrés y de funcionalidad en casos específicos.
Se realizó un recorte en el tiempo de realización del proyecto.	Debido a la amplia cantidad de pruebas que se requieren para la validación de la aplicación y servidor, sería necesario dedicar más esfuerzos al desarrollo porque el tiempo para pruebas puede reducir el volumen de desarrollo (para este caso, más vale cantidad que calidad)	Gracias al esfuerzo de todos, se consiguió terminar el desarrollo antes de la fecha límite.	Con el tiempo a favor, es posible realizar pruebas a las diferentes funcionalidades de la aplicación sin afectar al tiempo de desarrollo.

Se realizó un corte abrupto en el presupuesto del proyecto	Debido al costo que requieren algunos Softwares para algunas de las pruebas, es necesario suspender estas pruebas para quedar dentro de los límites de la nueva partida presupuestal	Se encontraron alternativas que pueden entrar en la nueva partida presupuestal	Teniendo el Software necesario para realizar las pruebas sin afectar el presupuesto del proyecto indica que no hay ningún impedimento para continuar con el desarrollo de las pruebas
--	--	--	---

#### e) Planeación de recursos

Elabora la planeación de recursos según las condiciones que se tienen para el proyecto. (con base a la situación planteada al inicio de este apartado)

Recursos humanos	Recursos económicos	Recursos materiales	Otros
Equipo de 8 integrantes para el desarrollo de la aplicación; siendo 6 de ellos expertos y 2 novatos.	Una partida presupuestal de \$80.000	Se cuenta con el equipo de cómputo (hardware) tanto para el desarrollo como las pruebas en sí (ver el cuadro de F para las especificaciones)	Se cuenta con un tiempo máximo de 30 días hábiles para el desarrollo y realización de pruebas de la aplicación.

#### f) Ambiente de pruebas

Anota las configuraciones necesarias para realizar las pruebas tanto de hardware como de software.

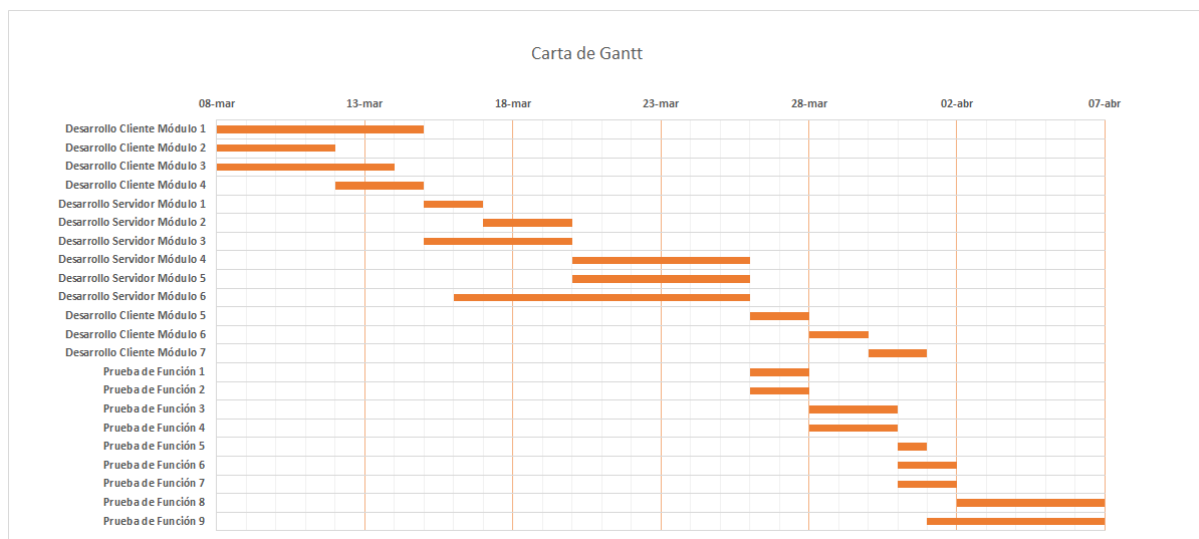
Hardware	Software
<b>CPU-Z:</b> Es un programa de detección de hardware gratuito para el sistema operativo Windows de Microsoft. Se trata de una aplicación que ayuda a recopilar información del sistema, y luego muestra los detalles en una sola pantalla. CPU-Z es	<b>Postman:</b> Es una herramienta que principalmente nos permite crear peticiones sobre APIs de una forma muy sencilla y poder, de esta manera, probar las APIs. Todo basado en una extensión de Google Chrome. El usuario de Postman

desarrollado por CPUID y actualizado regularmente, gracias a lo cual es compatible con la mayoría de procesadores y chipsets, incluso los más nuevos.	puede ser un desarrollador que esté comprobando el funcionamiento de una API para desarrollar sobre ella o un operador el cual esté realizando tareas de monitorización sobre un API.
<b>Motadata:</b> Es un integrado monitoreo de servidor software y herramienta que ofrece detalles perspicaces de todos los parámetros críticos de rendimiento del servidor, como la capacidad del disco duro, la utilización de la CPU, la utilización de la memoria y la utilización del ancho de banda desde una consola web intuitiva. Nuestro monitoreo del rendimiento del servidor permite que un administrador de sistemas esté al tanto del tiempo de inactividad del servidor y los problemas de rendimiento.	<b>JMeter:</b> es una herramienta que facilita la gestión integral de los procesos de pruebas de rendimiento, no obstante no es la única puesto que tenemos otras como: Micro Focus LoadRunner, IBM RPT, SilkPerformer, WebLoad, NeoLoad, OpenSTA, otras.
<b>CPkali:</b> Es un nuevo software que nos permite medir el ancho de banda máximo que conseguimos localmente, y también nos permitiría simular una conexión a Internet de un router neutro si conectamos un ordenador en la WAN de Internet y otro en la LAN, de esta manera comprobaremos si realmente proporciona buen rendimiento.	<b>Apache JMeter:</b> es una aplicación de código abierto, 100% basada en Java con una interfaz gráfica de usuario. Está diseñada para analizar, medir el rendimiento y cargar el comportamiento funcional de la aplicación web y la variedad de servicios
<b>LogicMonitor:</b> Es un Software de Application Performance Management Software creado por la empresa <b>LogicMonitor</b> (United States). Es la plataforma de monitorización del rendimiento de TI automatizado basado en SaaS para las infraestructuras en las instalaciones, híbridos, y en la nube.	<b>Selenium:</b> es un entorno de pruebas que se utiliza para comprobar si el software que se está desarrollando funciona correctamente. Esta herramienta permite: grabar, editar y depurar casos de pruebas que se pueden automatizar.
<b>Nagios:</b> es un sistema de monitorización de redes ampliamente utilizado de código abierto, que vigila los equipos (hardware) y servicios (software) que se especifiquen, alertando cuando el comportamiento de los mismos no sea el deseado. Ofrece	<b>LoadRunner:</b> Es un framework de Microfocus para realizar pruebas de rendimiento. Es un framework que está compuesto de varias herramientas. Es multiprotocolo pues nos permite grabar y ejecutar scripts en varios protocolos, cómo puede ser HTTP, SAP, etcétera. Es

<p>monitorización de servicios de red (SMTP, POP3, HTTP, SNMP...), la monitorización de los recursos de sistemas hardware, independencia de sistemas operativos, posibilidad de monitorización remota mediante túneles SSL cifrados o SSH, y la posibilidad de programar plugins específicos para nuevos sistemas.</p>	<p>multilenguaje de forma nativa.. No es Open Source, sino que es un framework propietario. Es gratuito hasta 50 usuarios concurrentes, y a partir de ahí se necesita más de una licencia.</p>
<p><b>Icinga:</b> Es un sistema de monitoreo de software libre, utiliza la licencia GNU GPL. es un sistema de monitorización que que vigila las redes y cualquier recurso de red concebible, notifica a usuario los errores y recuperaciones y genera datos de rendimiento para los informes</p>	<p><b>NeoLoad:</b> Es un software de prueba de carga y rendimiento para aplicaciones web y móviles que simula la actividad del usuario y monitoriza el funcionamiento del servidor. Facilita el despliegue de tus aplicaciones nativas de web, intranet o móvil con total confianza, independientemente de las tecnologías utilizadas, incluso las más recientes como GWT, Silverlight, Flex y Ajax Push.</p>

#### g) Calendarización

Realiza la estimación de tiempo de cada tarea y elabora un programa de trabajo con esa información, tomando en cuenta el número de personas disponibles y el tiempo de entrega.



En formato de tabla se tiene la siguiente distribución de actividades por tiempo e integrantes (EX es el experto número X, NX es el novato número X).

Actividad	Fecha Inicio	Fecha Fin	Duración	Responsable	% Completado	Días Completados
Desarrollo Cliente Módulo 1	08-mar	15-mar	7	E1, E2, E3	0.00%	0.00
Desarrollo Cliente Módulo 2	08-mar	12-mar	4	E4, N1, N2	0.00%	0.00
Desarrollo Cliente Módulo 3	08-mar	14-mar	6	E5, E6	0.00%	0.00
Desarrollo Cliente Módulo 4	12-mar	15-mar	3	E4, N1, N2	0.00%	0.00
Desarrollo Servidor Módulo 1	15-mar	17-mar	2	E5, E6	0.00%	0.00
Desarrollo Servidor Módulo 2	17-mar	20-mar	3	E5, E6	0.00%	0.00
Desarrollo Servidor Módulo 3	15-mar	20-mar	5	E4, N1, N2	0.00%	0.00
Desarrollo Servidor Módulo 4	20-mar	26-mar	6	E4, N1, N2	0.00%	0.00
Desarrollo Servidor Módulo 5	20-mar	26-mar	6	E5, E6	0.00%	0.00
Desarrollo Servidor Módulo 6	16-mar	26-mar	10	E1, E2, E3	0.00%	0.00
Desarrollo Cliente Módulo 5	26-mar	28-mar	2	E1, E2	0.00%	0.00
Desarrollo Cliente Módulo 6	28-mar	30-mar	2	E1, E2	0.00%	0.00
Desarrollo Cliente Módulo 7	30-mar	01-abr	2	E1, E2	0.00%	0.00
Prueba de Función 1	26-mar	28-mar	2	E5, E6, N2	0.00%	0.00
Prueba de Función 2	26-mar	28-mar	2	E3, E4, N1	0.00%	0.00
Prueba de Función 3	28-mar	31-mar	3	E5, E6, N2	0.00%	0.00
Prueba de Función 4	28-mar	31-mar	3	E3, E4, N1	0.00%	0.00
Prueba de Función 5	31-mar	01-abr	1	E3, E4	0.00%	0.00
Prueba de Función 6	31-mar	02-abr	2	E5, N1	0.00%	0.00
Prueba de Función 7	31-mar	02-abr	2	E6, N2	0.00%	0.00
Prueba de Función 8	02-abr	07-abr	5	E3, E4, E5, E6, N1, N2	0.00%	0.00
Prueba de Función 9	01-abr	07-abr	6	E1, E2	0.00%	0.00

## h) Entregables

Realiza un reporte completo del plan de pruebas



# Reporte de Plan de Pruebas

## 1. Introducción y Análisis de la aplicación

El presente plan de pruebas contiene la estrategia que se seguirá para realizar de manera adecuada la ejecución de pruebas sobre la aplicación desarrollada por un equipo conformado por 6 expertos y 2 novatos. Algunos de los requisitos generales del proyecto que fueron determinados por el cliente fueron: Una partida presupuestal de \$80.000 destinados al desarrollo y pruebas, así como un límite de tiempo de 30 días hábiles para llevar a cabo todo el desarrollo y las pruebas pertinentes para garantizar la funcionalidad de la aplicación.

La aplicación a desarrollar es una PWA para resolver ecuaciones diferenciales capturadas por una imagen, texto en pantalla o por entrada en teclado. Está distribuida en dos partes: cliente y servidor. El cliente está basado en el modelo de desarrollo vista-controlador implementado para el desarrollo de una PWA por medio de código Python. El servidor estará alojado de manera remota y será por medio de solicitudes HTTP (con cuerpo JSON) que el servidor recibe la solicitud de resolver una determinada ecuación diferencial y retorna la respuesta a la aplicación cliente que solicitó resolver dicha ecuación. Para resolver la ecuación, el servidor cuenta con una serie de métodos predefinidos dependiendo del tipo de ecuación diferencial y posee un soporte simbólico por el módulo SymPy de Python.

Para funcionar correctamente requiere de una buena conexión a internet por parte del usuario que utilice la aplicación cliente para lanzar solicitudes hacia el servidor. Además, el servidor requiere estar siempre activo y tener una respuesta rápida para gestionar múltiples solicitudes en tiempos que le permitan al usuario tener una experiencia cómoda de uso; esto es, se requiere que el hosteador del servidor tenga un desempeño dentro de lo establecido en la calidad de respuesta. Al ser una PWA se espera alcanzar una compatibilidad muy alta con una gran gama de dispositivos móviles y de escritorio, de modo que no se solicita un Hardware en especial para utilizar la aplicación cliente por parte del usuario. Debido a que la mayoría de las herramientas a utilizar para su desarrollo y posterior implementación se basan en código Open Source, no se tiene una lista larga de requisitos para garantizar la funcionalidad de la aplicación.

## 2. Funcionalidades

Las siguientes son las funciones del sistema que serán sometidas a pruebas con sus respectivas metas.

- ❖ **Obtención de información del usuario por medio de una forma:** Obtener la información del usuario, sin perder la integridad de los datos y guardarlos en una base de datos. Esta información constituye la cuenta del usuario.
- ❖ **Conversión del texto de una imagen a texto plano:** Obtener el texto de una imagen, idealmente el texto debe ser el mismo que de la imagen. De forma que podamos modificarlo y trabajar con él como texto plano.
- ❖ **Interpretación del texto plano a una ecuación definida:** El texto plano llevará una cierta sintaxis definida con la que reconstruiremos una ecuación matemática:
- ❖ **Conversión del texto plano a latex:** Utilizamos el texto plano y la sintaxis definida para reconstruir la ecuación que interpretó el sistema en formato LaTeX.

- ❖ **Identificación de la respectiva ecuación diferencial:** Una vez hemos obtenido la ecuación el sistema deberá matemáticamente identificar correctamente el tipo de ecuación diferencial que ha sido insertada.
- ❖ **Resolución de la ecuación diferencial:** Tras determinarse el tipo de ecuación diferencial, el sistema deberá ser capaz de resolver la ecuación diferencial de forma algebraica siguiendo los pasos usuales como si fuera contestada “a lapiz”. Así obtener una solución.
- ❖ **Obtención de la respuesta del módulo de resolución de la ecuación diferencial:** La respuesta es regresar del módulo de solución a la aplicación, la comunicación se da de forma correcta entre ambos módulos.
- ❖ **La respuesta se imprime en la plataforma en formato LaTeX:** La respuesta regresada se desempaqueta y es tratada de forma adecuada para la presentación de los pasos y de la solución de la ecuación diferencial. Dejándola en el formato LaTeX correspondiente.
- ❖ **Se agrega dicha ecuación al historial del usuario:** Actualizar la base de datos de forma correcta de forma que el historial del usuario incluya la ecuación diferencial que recién pidió.

### 3. Riesgos

- ❖ **No se cuenta con los testers suficientes para realizar las pruebas en el tiempo establecido (Medio):** La gran cantidad de ecuaciones que debe resolver la aplicación requiere de la mayor cantidad de pruebas posibles
- ❖ **El equipo de pruebas está conformado en su mayoría por testers con poca experiencia (Medio):** No se requiere demasiada experiencia para probar al sistema: solo requiere tomar una foto a un texto y comparar lo que devuelve el sistema con lo esperado
- ❖ **El ambiente de pruebas no está configurado adecuadamente (Medio):** Existen 3 maneras de interactuar con el sistema para realizar pruebas, en donde cada uno de ellos es bastante flexible, por lo que aún en condiciones adversas se espera encontrar una ruta para realizar las pruebas
- ❖ **La aplicación tiene problemas de seguridad (Medio):** La aplicación en sí no guarda valores muy importantes de los usuarios (solo su cuenta y contraseña, que se utiliza para el historial de consultas de ecuaciones). Sin embargo, la posibilidad de afectar código y con ello el funcionamiento podría ser riesgoso.
- ❖ **El presupuesto para la ejecución de las pruebas es menor al que debería ser (Medio):** El proyecto está sentado en las bases del desarrollo Open Source, aunque algunas implementaciones de APIs como Google Vision requieren un pequeño aporte (200 pesos mensuales en un mal caso). De esta forma, una salida de la partida presupuestal sería complicada pero solventable por el equipo.
- ❖ **Unos de los inversionistas retira su apoyo al proyecto, por lo cual el equipo de prueba sufre recortes de presupuesto (Alto):** Si por alguna razón las tecnologías que se utilizan de manera Open Source cambian abruptamente a cobrar una cuota por servicio el proyecto se ve en apuros debido a que el proyecto en sí se basa en la interacción de todas estas herramientas: si una se bloquea, la aplicación no puede funcionar.

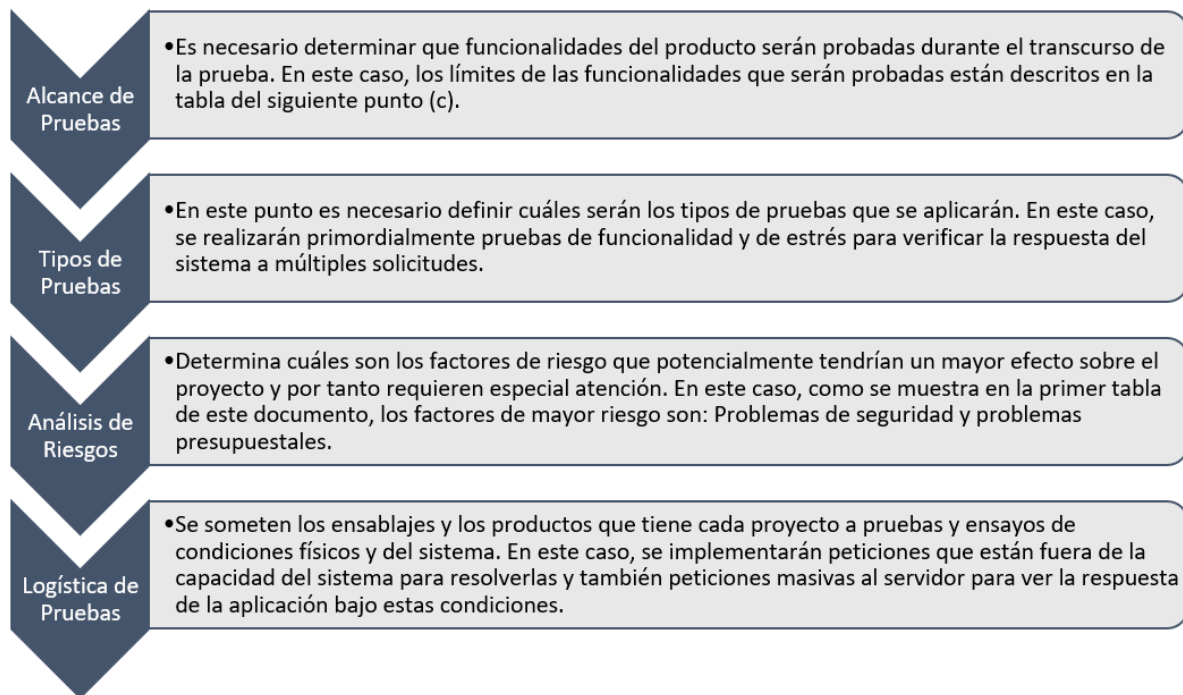
#### 4. Criterios de suspensión y reanudación

Las siguientes son los criterios de suspensión de realización de pruebas argumentados con su respectivo criterio para reanudar con la aplicación de las pruebas.

- ❖ **Se perdió la conexión a Internet por parte del equipo de testers:** Sin una conexión a internet es imposible desarrollar las pruebas de funcionalidad del proyecto. Regresa la conexión de Internet al equipo de testers: Una vez restablecida la conexión a Internet del equipo de testers no existe impedimento para continuar con las pruebas de funcionalidad
- ❖ **Se perdió la conexión con el servidor (o fue rechazada):** Sin una comunicación con el servidor por parte de la aplicación cliente es imposible desarrollar las pruebas de funcionalidad del proyecto. Se restablece la conexión con el servidor por medio de una serie de modificaciones en el servidor: Una vez restablecida la conexión con el servidor por parte del equipo de testers no existe impedimento para continuar con las pruebas de funcionalidad.
- ❖ **Se registró una cantidad de 5 respuestas erróneas por parte del servidor para diferentes ecuaciones (diferentes tipos):** Tener un registro de malas respuestas por parte del servidor de esa magnitud es un indicador de que hay errores de codificación contundentes desde el lado del servidor que deben ser corregidos antes de perder más tiempo en pruebas. Una vez corregidos los errores del servidor, se realiza una prueba rápida con las mismas 5 ecuaciones que fallaron al poner en suspensión las pruebas: Si es favorable en las 5, se continúa con las pruebas Obteniendo una correcta respuesta correcta para 5 tipos de ecuaciones diferentes es un indicador de que el servidor está realizando las operaciones correspondientes de manera adecuada y está listo para continuar con las pruebas de funcionalidad
- ❖ **Se detectaron fallas en la seguridad de la aplicación:** No se puede continuar con la ejecución de pruebas si es posible que el código no esté asegurado así como la integridad de los datos de los usuarios, aún en la etapa de pruebas de funcionalidad. La falla de seguridad que fue reportada fue correctamente atendida: Una vez restablecida la seguridad del código y de la información de la aplicación es posible continuar con las pruebas de funcionalidad
- ❖ **Se tiene un grupo muy limitado de testers disponibles en algún momento:** Las pruebas de funcionalidad de la aplicación requieren de un equipo grande de testers para abordar las diferentes funcionalidades y factores que se deben probar antes del lanzamiento de la aplicación. Se reunió un grupo con una cantidad de testers adecuada (6): Con un grupo de ese tamaño es posible emular diferentes circunstancias para el desarrollo de las pruebas de estrés y de funcionalidad en casos específicos.
- ❖ **Se realizó un recorte en el tiempo de realización del proyecto:** Debido a la amplia cantidad de pruebas que se requieren para la validación de la aplicación y servidor, sería necesario dedicar más esfuerzos al desarrollo porque el tiempo para pruebas puede reducir el volumen de desarrollo (para este caso, más vale cantidad que calidad). Gracias al esfuerzo de todos, se consiguió terminar el desarrollo antes de la fecha límite: Con el tiempo a favor, es posible realizar pruebas a las diferentes funcionalidades de la aplicación sin afectar al tiempo de desarrollo.
- ❖ **Se realizó un corte abrupto en el presupuesto del proyecto:** Debido al costo que requieren algunos Softwares para algunas de las pruebas, es necesario suspender estas pruebas para quedar dentro de los límites de la nueva partida presupuestal . Se encontraron alternativas que pueden entrar en la nueva partida presupuestal. Teniendo el Software

necesario para realizar las pruebas sin afectar el presupuesto del proyecto indica que no hay ningún impedimento para continuar con el desarrollo de las pruebas

## 5. Estrategia de Pruebas



## 6. Planificación, componentes del proyecto y recursos (software y hardware)

Los recursos humanos que constituyen al equipo de desarrollo y pruebas se compone por 8 personas, de entre las cuales 6 son expertos y 2 son novatos. Los recursos financieros ascienden a lo establecido en una partida presupuestal de \$80.000 proporcionados por el cliente para pruebas y desarrollo de la aplicación. Se cuenta además con un tiempo establecido de 30 días hábiles destinados a desarrollo y pruebas de la aplicación.

Los recursos materiales que se requieren para realizar las pruebas se dividen en dos:

a) Recursos para pruebas en hardware:

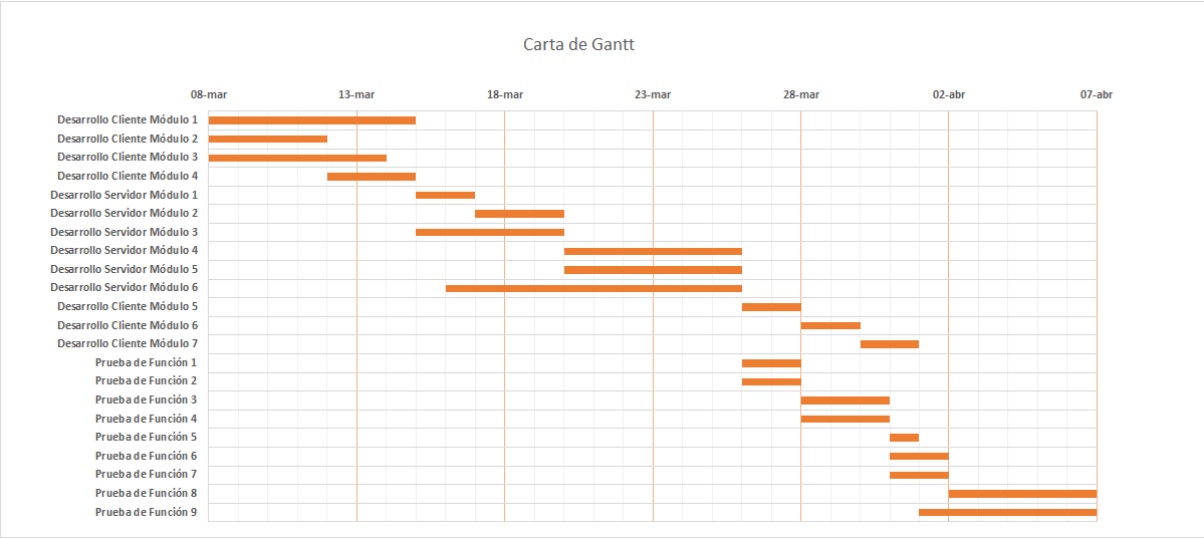
- ❖ **CPU-Z:** Es un programa de detección de hardware gratuito para el sistema operativo Windows de Microsoft. Se trata de una aplicación que ayuda a recopilar información del sistema, y luego muestra los detalles en una sola pantalla. CPU-Z es desarrollado por CPUID y actualizado regularmente, gracias a lo cual es compatible con la mayoría de procesadores y chipsets, incluso los más nuevos.
- ❖ **Motadata:** Es un integrado monitoreo de servidor software y herramienta que ofrece detalles perspicaces de todos los parámetros críticos de rendimiento del servidor, como la capacidad del disco duro, la utilización de la CPU, la utilización de la memoria y la utilización del ancho de banda desde una consola web intuitiva. Nuestro monitoreo del rendimiento del servidor permite que un administrador de sistemas esté al tanto del tiempo de inactividad del servidor y los problemas de rendimiento.

- ❖ **CPkali:** Es un nuevo software que nos permite medir el ancho de banda máximo que conseguimos localmente, y también nos permitiría simular una conexión a Internet de un router neutro si conectamos un ordenador en la WAN de Internet y otro en la LAN, de esta manera comprobaremos si realmente proporciona buen rendimiento.
- ❖ **LogicMonitor:** Es un Software de Application Performance Management Software creado por la empresa LogicMonitor (United States). Es la plataforma de monitorización del rendimiento de TI automatizado basado en SaaS para las infraestructuras en las instalaciones, híbridos, y en la nube.
- ❖ **Nagios:** es un sistema de monitorización de redes ampliamente utilizado de código abierto, que vigila los equipos (hardware) y servicios (software) que se especifiquen, alertando cuando el comportamiento de los mismos no sea el deseado. Ofrece monitorización de servicios de red (SMTP, POP3, HTTP, SNMP...), la monitorización de los recursos de sistemas hardware, independencia de sistemas operativos, posibilidad de monitorización remota mediante túneles SSL cifrados o SSH, y la posibilidad de programar plugins específicos para nuevos sistemas.
- ❖ **Icinga:** Es un sistema de monitoreo de software libre, utiliza la licencia GNU GPL. es un sistema de monitorización que que vigila las redes y cualquier recurso de red concebible, notifica al usuario los errores y recuperaciones y genera datos de rendimiento para los informes.

b) Recursos para pruebas en software:

- ❖ **Postman:** Es una herramienta que principalmente nos permite crear peticiones sobre APIs de una forma muy sencilla y poder, de esta manera, probar las APIs. Todo basado en una extensión de Google Chrome. El usuario de Postman puede ser un desarrollador que esté comprobando el funcionamiento de una API para desarrollar sobre ella o un operador el cual esté realizando tareas de monitorización sobre un API.
- ❖ **JMeter:** es una herramienta que facilita la gestión integral de los procesos de pruebas de rendimiento, no obstante no es la única puesto que tenemos otras como: Micro Focus LoadRunner, IBM RPT, SilkPerformer, WebLoad, NeoLoad, OpenSTA, otras.
- ❖ **Apache JMeter:** es una aplicación de código abierto, 100% basada en Java con una interfaz gráfica de usuario. Está diseñada para analizar, medir el rendimiento y cargar el comportamiento funcional de la aplicación web y la variedad de servicios
- ❖ **Selenium:** es un entorno de pruebas que se utiliza para comprobar si el software que se está desarrollando funciona correctamente. Esta herramienta permite: grabar, editar y depurar casos de pruebas que se pueden automatizar.
- ❖ **LoadRunner:** Es un framework de Microfocus para realizar pruebas de rendimiento. Es un framework que está compuesto de varias herramientas. Es multiprotocolo pues nos permite grabar y ejecutar scripts en varios protocolos, cómo puede ser HTTP, SAP, etcétera. Es multilenguaje de forma nativa.. No es Open Source, sino que es un framework propietario. Es gratuito hasta 50 usuarios concurrentes, y a partir de ahí se necesita más de una licencia.
- ❖ **NeoLoad:** Es un software de prueba de carga y rendimiento para aplicaciones web y móviles que simula la actividad del usuario y monitoriza el funcionamiento del servidor. Facilita el despliegue de tus aplicaciones nativas de web, intranet o móvil con total confianza, independientemente de las tecnologías utilizadas, incluso las más recientes como GWT, Silverlight, Flex y Ajax Push.

La distribución de las actividades para el desarrollo y pruebas tanto en tiempo y recursos humanos se estableció a manera de Carta de Gantt, en donde se indican las diferentes actividades a realizar:



**Carta de Gantt del Desarrollo y Pruebas de la Aplicación**

Actividad	Fecha Inicio	Fecha Fin	Duración	Responsable	% Completado	Días Completados
Desarrollo Cliente Módulo 1	08-mar	15-mar	7	E1, E2, E3	0.00%	0.00
Desarrollo Cliente Módulo 2	08-mar	12-mar	4	E4, N1, N2	0.00%	0.00
Desarrollo Cliente Módulo 3	08-mar	14-mar	6	E5, E6	0.00%	0.00
Desarrollo Cliente Módulo 4	12-mar	15-mar	3	E4, N1, N2	0.00%	0.00
Desarrollo Servidor Módulo 1	15-mar	17-mar	2	E5, E6	0.00%	0.00
Desarrollo Servidor Módulo 2	17-mar	20-mar	3	E5, E6	0.00%	0.00
Desarrollo Servidor Módulo 3	15-mar	20-mar	5	E4, N1, N2	0.00%	0.00
Desarrollo Servidor Módulo 4	20-mar	26-mar	6	E4, N1, N2	0.00%	0.00
Desarrollo Servidor Módulo 5	20-mar	26-mar	6	E5, E6	0.00%	0.00
Desarrollo Servidor Módulo 6	16-mar	26-mar	10	E1, E2, E3	0.00%	0.00
Desarrollo Cliente Módulo 5	26-mar	28-mar	2	E1, E2	0.00%	0.00
Desarrollo Cliente Módulo 6	28-mar	30-mar	2	E1, E2	0.00%	0.00
Desarrollo Cliente Módulo 7	30-mar	01-abr	2	E1, E2	0.00%	0.00
Prueba de Función 1	26-mar	28-mar	2	E5, E6, N2	0.00%	0.00
Prueba de Función 2	26-mar	28-mar	2	E3, E4, N1	0.00%	0.00
Prueba de Función 3	28-mar	31-mar	3	E5, E6, N2	0.00%	0.00
Prueba de Función 4	28-mar	31-mar	3	E3, E4, N1	0.00%	0.00
Prueba de Función 5	31-mar	01-abr	1	E3, E4	0.00%	0.00
Prueba de Función 6	31-mar	02-abr	2	E5, N1	0.00%	0.00
Prueba de Función 7	31-mar	02-abr	2	E6, N2	0.00%	0.00
Prueba de Función 8	02-abr	07-abr	5	E3, E4, E5, E6, N1, N2	0.00%	0.00
Prueba de Función 9	01-abr	07-abr	6	E1, E2	0.00%	0.00

**Tabla de Distribución de Actividades de Desarrollo y Pruebas**

## Referencias

Cillero. M. Especificación del plan de pruebas. Recuperado el 05/03/2021 de:

<https://manuel.cillero.es/doc/metodologia/metrica-3/procesos-principales/asi/actividad-10/#:~:text=El%20plan%20de%20pruebas%20es,de%20las%20actividades%20de%20prueba.>

Carricay. Que son los casos de pruebas. Recuperado el 05/03/2021 de:

<https://medium.com/grupo-carricay/qu%C3%A9-son-los-casos-de-pruebas-4893799b5b84>