

**Documento de Pruebas del Software**

**Sistema para la Inspección de un Parque Fotovoltaico**

Materia:

**Aseguramiento de la Calidad de Software**

Maestra:

**Dra. Sonia López Ruiz**

Alumno:

**Raúl Rubén Romero López**

Identificador: SIPaF-TCAD-V0.1

Ameca, Jalisco 07 de noviembre de 2022

Control de cambios

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Descripción | Autor | Fecha |
| 0.1 | Versión preliminar | Raúl Romero | 07/11/2022 |

Revisión

|  |  |  |
| --- | --- | --- |
| Versión | Responsable | Fecha |
|  |  |  |

**Tabla de Contenido**

[**1. Análisis del producto de software** 3](#_Toc119966800)

[**1.1 Perspectiva del sistema SIPaF** 3](#_Toc119966801)

[**1.2 Funciones del sistema SIPaF** 3](#_Toc119966802)

[**1.3 Objetivo general del sistema** 4](#_Toc119966803)

[**1.4 Requerimientos generales del sistema** 4](#_Toc119966804)

[**1.5 Módulos del sistema SIPaF** 5](#_Toc119966805)

[**2. Modelo de evaluación de la calidad del software** 7](#_Toc119966806)

[**2.1 Características del modelo de las pruebas** 7](#_Toc119966807)

[**2.2 Subcaracterísticas del modelo de pruebas** 9](#_Toc119966808)

# **1. Análisis del producto de software**

En esta sección se presenta un resumen del sistema, lo que engloba los puntos mas importantes del software y su composición. Esto con la intención de tener en claro lo que se esta pidiendo que haga el sistema para poder diseñar mejores pruebas que puedan asegurar la calidad del sistema.

## **1.1 Perspectiva del sistema SIPaF**

Los parques solares tienen una extensión muy amplia como para basarse únicamente en un monitoreo de los vatios producidos en tiempo real. El monitoreo es requerido por si alguna celda de un panel solar se da ̃no por alguna razón antes mencionada o por otra situación reduciendo la eficiencia. Las mediciones de vatios son un promedio de un sector lleno de paneles solares, por lo que aun hace falta hacer una inspección de paneles solares en esa zona. Hacer una inspección visual de todos los paneles de la zona es costoso y tardado. Es por esto que se están utilizando los vehículos aéreos no tripulados georreferenciados con c ́amaras RGB y térmicas para detectar si alguna celda tiene alguna falla reduciendo el tiempo y costo de inspección lo que beneficia el tiempo de producción efectivo.

El sistema SIPaF busca ser un sistema centralizado y monousuario. Por lo que solo se utilizara en el ordenador en el que este instalado. Dicho sistema propone hacer uso de herramientas tecnológicas para lograr el objetivo planteado por el cliente, en este caso, el director de tesis Dr. Himer Ávila George. Algunas de las tecnologías a utilizar importantes son:

* Imágenes RGB y térmicas
* Visión por computadora
* Redes neuronales convolucionales
* Manejo de funciones del sistema operativo

## **1.2 Funciones del sistema SIPaF**

Para ello es necesario el análisis y el diseño del sistema. De los cuales el sistema ya cuenta con los documentos SIPaF-ERS-V1.0 qué es el documento de requerimientos y el SIPAF-SSD-V0.1 qué es el documento de análisis y diseño. Gracias a esto se puede tener una visión clara de lo que el cliente espera que ejecute el programa.

En base a los documentos el sistema debe cumplir con las siguientes funcionalidades:

1. Administrar espacios de trabajo para realizar proyectos.
2. Especificar para cada proyecto que imágenes van a ser analizadas.
3. Crear un reporte con la información obtenida por el software.
4. Mostrar todos los datos y crear un reporte con las imágenes de los resultados obtenidos.
5. Crear archivos pdf con los reportes.

## **1.3 Objetivo general del sistema**

Desarrollar un sistema no intrusivo para el monitoreo de plantas solares, el cual identifique las celdas que estén fallando, utilizando imágenes térmicas capturadas desde un vehículo aéreo no tripulado procesadas mediante técnicas de procesamiento de imágenes y aprendizaje profundo.

## **1.4 Requerimientos generales del sistema**

De las necesidad y solicitudes del cliente se deben de generar los requerimientos que el sistema debe cumplir. Los requerimientos funcionales del software que se han obtenido, los cuales se encuentran en el documento SIPaF-ERS, se listan a continuación junto con una descripción de cada uno de ellos:

**RF-01 Crear un espacio de trabajo:** Crear un espacio de trabajo para poder crear proyectos y poder compararlos después, además de ayudar a tener separados por lugar cada proyecto.

**RF-02 Cambiar el espacio de trabajo:** Elegir el espacio de trabajo con los proyectos relacionados a este que se quiera abrir.

**RF-03 Eliminar espacio de trabajo:** Eliminar el espacio de trabajo con los proyectos relacionados a este del sistema.

**RF-04 Crear proyecto:** Crear un proyecto en el que se le pueden especificar algunas variables para el procesamiento de las imágenes.

**RF-05 Eliminar proyecto:** Eliminar un proyecto y los archivos creados por el sistema relacionados a este proyecto.

**RF-06 Modificar proyecto:** Seleccionar un proyecto existente para volver a realizar el análisis modificando parámetros o simplemente volverlo a realizar por si hubo algún error y no se terminó.

**RF-07 Preprocesar las imágenes:** Se debe de hacer un preprocesamiento de las imágenes antes de comenzar el análisis para evitar meter valores erróneos al sistema y poder generalizar los resultados.

**RF-08 Crear ortofoto:** Crear la ortofoto con las imágenes seleccionadas para el análisis.

**RF-09 Buscar archivo:** Buscar un archivo dentro del espacio de trabajo mostrando los archivos que coincidan con la búsqueda en la barra de visualización donde se muestran los archivos del espacio de trabajo.

**RF-10 Crear reporte:** Crear un reporte con la información obtenida del proyecto en formato PDF.

**RF-11 Eliminar reporte:** Eliminar un reporte creado para un proyecto.

**RF-12 Establecer correo electrónico:** Configurar un correo electrónico en el sistema.

**RF-13 Visualizar reporte:** Abrir un reporte creado al seleccionarlo.

**RF-14 Compartir reporte:** El reporte obtenido puede ser compartido a través de correo electrónico.

**RF-15 Imprimir reporte:** Poder mandar imprimir el reporte en alguna impresora ya configurada.

**RF-16 Visualizar datos de varios reportes:** Permite al usuario seleccionar múltiples proyectos para visualizar los datos de ellos y poder hacer comparaciones.

**RF-17 Guardar las gráficas creadas:** Guardar las gráficas que se hayan creado para comparar entre proyectos.

**RF-18 Visualizar las fallas identificadas:** Ver en una ortofoto donde se encuentran las fallas identificadas por el sistema.

**RF-19 Abrir proyecto:** Cargar un archivo de un proyecto ya creado para visualizar la información.

**RF-20 Guardar checkpoints del proceso:** Al momento de empezar a correr el proyecto, que se puedan guardar los puntos importantes por si en algún punto el proceso es interrumpido, que no se tenga que iniciar todo el principio desde el principio si así se desea.

**RF-21 Guardar archivo de proyecto:** Guardar el archivo con toda la información generada para que pueda ser cargada sin la necesidad de realizar de nuevo el proyecto.

Además, se cuentan con requerimientos no funcionales del sistema, que se listan a continuación:

**RNF-01:** Utilizar la tarjeta gráfica del ordenador si es que cuenta con una.

**RNF-02:** Funcionar con las tarjetas gráficas de la marca Nvidia.

**RNF-03:** Los correos que pueden ser ingresados deben de pertenecer únicamente al dominio especificado.

**RNF-04:** Generar resultados aceptables.

**RNF-05:** Advertir si hubo un error, mostrar el error y no continuar con el proceso.

**RNF-06:** Aceptar actualizaciones en un futuro a los módulos del sistema.

**RNF-07:** Permitir actualizar el modelo entrenado para la detección de fallas en los paneles solares.

**RNF-08:** Poder instalarse en Windows 7 en adelante, una arquitectura de 64 bits y un mínimo de 6gb RAM.

**RNF-09:** Contar con un logo.

**RNF-10:** Proporcionar ayudas de tipo texto para las funciones principales.

**RNF-11:** Ser capaz de enviar correos electrónicos.

**RNF-12:** Encriptar las contraseñas del usuario.

**RNF-13:** Tener una propiedad intelectual.

**RNF-14:** Contar con un manual de usuario.

## **1.5 Módulos del sistema SIPaF**

**1. Espacios de trabajo (EDT):** Permite crear espacios de trabajo donde se van a poder proyectos. Los espacios de trabajo son como folders del sistema donde se van a poder crear los proyectos. También es el que se va a encargar de guardar la información del correo electrónico para compartir los reportes. Este módulo es el encargado de mandar crear un proyecto nuevo y de cargar los proyectos relacionados a este espacio de trabajo para ser visualizados. No recibe mucha información mas que el nombre y la dirección del EDT.

**2. Proyectos:** Este módulo se encarga de gestionar la información de los proyectos e iniciar el proceso. Este módulo es el de mayor importancia para el sistema porque es donde se inicia el proyecto y el que contiene la información más relevante para el usuario. Este módulo es el que comienza la interacción para hacer los análisis de las imágenes. Es el encargado también de leer guardar información para empezar el proceso como el nombre, dirección, área de vuelo, el reporte final, la dirección de las imágenes a utilizar, entre otros parámetros. También, debe ser capaz de leer los documentos del sistema creados al final para poder cargar toda la información del proyecto.

**3. Ortofotos:** Este módulo no realiza o tiene tantos métodos, pero lo que hace es esencial para que el proceso termine adecuadamente y no generé resultados erróneos. Como métodos es el encargado de preprocesar las fotos obtenidas antes de generar las ortofotos. Las nuevas imágenes deben se guardadas en el sistema para que puedan usarse en la ortofoto. Lo importante que pasa al siguiente modulo son las ortofotos creadas, tanto para el RGB como para las térmicas.

**4. Modelo entrenado:** Este módulo contiene el método para analizar las imágenes, por lo que es importante verificar que este funcionando. Antes de integrarlo al sistema, se debe de verificar que el proceso elegido genere buenos resultados y se debe validar tanto la entrada que recibe como la información de salida. En este caso, la entrada va a ser la ortofoto térmica y la salida es la información necesaria para generar el reporte. Como información principal que debe de enviar es la ortofoto con los defectos encontrados y los paneles identificados.

**5. Reportes:** Para terminar, el modulo de los reportes es el que debe de tener toda la información generada por el modelo entrenado para después ser mostrada por la interfaz al usuario. De igual manera, debe de poder crear gráficas si así fuera necesario con la información encontrada entre diferentes proyectos. Otra función importante es la de crear el archivo pdf con los datos obtenidos.

# **2. Modelo de evaluación de la calidad del software**

A continuación, se describe el modelo de evaluación de la calidad del software donde se plasman las características que se consideran necesarias para evaluar de acuerdo a los requisitos expresados por el cliente basándose en las normas ISO/IEC 2501n y la norma ISO/IEC 9126 para el modelo de calidad. La tabla 1, mostrada a continuación, muestra el porcentaje de importancia de acuerdo al nivel de importancia, se muestra también la simbología y su significado.

## **2.1 Características del modelo de las pruebas**

En esta sección se muestran como se van a dividir las pruebas, en este caso vamos a tener pruebas para la calidad interna y calidad externa. Para este sistema no se va a lograr aplicar las pruebas en uso pues no se tiene contemplado en el tiempo del proyecto. Lo primero que vamos a hacer es definir el nivel de importancia para cada característica dependiendo de en que parte se encuentre. Luego se especifican las características y su importancia. La tabla 1 explica claramente lo que significan los niveles de importancia en este proyecto.

**Tabla 1.** Especificación de importancia para las características a probar.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nivel de importancia** | **Simbología** | **Porcentaje del nivel de importancia** | **Significado** |
| Alto | **A** | 75% | El grado de importancia de la característica y subcaracterística es alto por lo tanto el esfuerzo y las mediciones destinadas a esas características y subcaracterísticas consumirá un alto porcentaje de recursos hasta donde se considere necesario. |
| Medio | **M** | 20% | El grado de importancia de la característica y subcaracterística es medio, por lo que puede o no ser medida. |
| Bajo | **B** | 5% | El grado de importancia de la característica o sub característica es bajo porque no son tan relevantes y difícilmente se llegarán a medir. |
| No Aplica | **NA** | 0% | El grado de importancia de la característica o sub característica no se puede medir por algún motivo. |

La calidad interna es la que se busca que tenga el software durante el proceso de desarrollo, cuando se encuentra en desarrollo. Aquí es donde entran las pruebas unitarias para revisar las funciones que se han hecho dentro de cada clase. Es de gran importancia pues es donde si llegan a encontrar errores es donde serán más fáciles de detectar que los causa y corregirlos. La tabla 2, mostrada a continuación, muestra las características de calidad interna que se proponen para evaluar el sistema a lo largo de las distintas etapas del desarrollo y codificación del mismo.

**Tabla 2.** Características de la calidad interna del sistema.

|  |  |  |
| --- | --- | --- |
| **Característica** | **Nivel de importancia** | **Motivo de selección** |
| Adecuación funcional | A | Se califica con valor de importancia M porque es necesario evaluar que las funciones codificadas estén de acuerdo a los requerimientos especificados en el sistema. |
| Fiabilidad | M | Se califica con M porque se considera necesario evaluar que el sistema cumpla con la función especificada en los requerimientos aun siendo sometido a ciertas condiciones durante un tiempo determinado. |
| Eficiencia en el desempeño | M | Se califica con valor de importancia M porque es necesario evaluar el rendimiento del sistema tomando en cuenta los recursos que serán utilizados. |
| Usabilidad | B | Se califica con valor de importancia B porque es necesario que las funciones del programa sean fácilmente identificadas. |
| Seguridad | B | Se califica con valor de importancia B porque es necesario asegurar la protección de los datos del correo electrónico y la contraseña únicamente. |
| Compatibilidad | B | Se considera un valor B porque es necesario que el sistema cumpla con sus funciones sin afectar otros programas. |
| Mantenibilidad | B | Se califica con valor de importancia B porque es necesario que el software este realizado de cierta manera para que pueda ser modificado en un futuro si fuera necesario. |
| Portabilidad | NA | Para las características internas del sistema no se tiene pensado que se prueba la parte de la portabilidad aún. |

La calidad externa es la indicada para mostrar la calidad del producto antes de pasarla al usuario final en un ambiente cercano al de trabajo. En esta parte se llevan a cabo pruebas de integración, que verifican que todos los módulos creados funcionen adecuadamente en una máquina con las características para las que el software fue desarrollado probando un dispositivo diferente al donde fue desarrollado. La tabla 3, presentada a continuación, muestra las características de calidad externa propuestas y su nivel de importancia para evaluar al sistema una vez que pasa a la etapa de producción para ser probado.

**Tabla 3.** Características de la calidad externa del sistema.

|  |  |  |
| --- | --- | --- |
| **Característica** | **Nivel de importancia** | **Motivo de selección** |
| Adecuación funcional | A | Se califica con valor de importancia A porque es necesario evaluar que los módulos trabajando en conjunto sigan cumpliendo con los requerimientos especificados del sistema. |
| Fiabilidad | A | Se califica con A porque se considera necesario evaluar que el sistema cumpla con la función especificada en los requerimientos aun siendo sometido a ciertas condiciones ya en un ambiente más cercano al de trabajo. |
| Eficiencia en el desempeño | B | Se califica con valor de importancia B porque es necesario evaluar el rendimiento del sistema tomando en cuenta los recursos que serán utilizados ya en un ambiente cercano al real. |
| Usabilidad | M | Se califica con valor de importancia M porque es necesario que las funciones del programa se puedan entender de manera intuitiva o que leyendo una vez el manual uno se pueda dar una idea de cómo usarlo. |
| Seguridad | B | Se califica con valor de importancia B porque es necesario asegurar la protección de los datos del correo electrónico y la contraseña únicamente. |
| Compatibilidad | M | Se considera un valor M porque es necesario que el sistema cumpla con sus funciones sin afectar otros programas. |
| Mantenibilidad | A | Se califica con valor de importancia A porque es necesario que el código sea modificado o actualizado en caso de que existan fallas con el sistema. |
| Portabilidad | B | Se debe probar que se pueda usar en otros sistemas operativos de Windows y que utilice las tarjetas gráficas si es que cuentan con una, pero no es muy importante |

## **2.2 Subcaracterísticas del modelo de pruebas**

En esta subsección se muestran las subcaracterísticas escogidas para cada característica a evaluar. A continuación, se van a ver las subcaracterísticas definidas junto con las métricas a evaluar para cada una de ellas. La importancia que se esta contemplando para cada una de ellas como la importancia, su ponderación, las métricas y su significado. Primero se muestra en la tabla 4 las subcaracteristicas y métricas de la calidad interna del software. Después la tabla 5 muestra la subcaracterísticas y métricas de la calidad externa. Luego de las tablas se muestran los requerimientos para poder aplicarle las pruebas de calidad al software y en que parte.

**Tabla 4.** Subcaracterísticas de la calidad interna del sistema.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SUBCARACTERÍSTICAS Y ATRIBUTOS DE CALIDAD INTERNA | | | | | | |
| Características | Nivel de Importancia | Ponderación | Subcaracterísticas | Nivel de Importancia | Métricas | Significado |
| Adecuación funcional | M | 15% | Completitud funcional | A | Completitud de la implementación funcional | Qué tan completa es la implementación de acuerdo con la  especificación de  requerimientos funcionales y no funcionales |
| Exactitud funcional | M | Presión computacional | La frecuencia con que ocurren  los resultados inexactos |
| Fiabilidad | M | 15% | Madurez | A | Eliminación de errores | Cuántos posibles errores han sido corregidos |
| Cobertura de pruebas | Cuántos casos de prueba han  sido ejecutados durante la  etapa de pruebas |
| Disponibilidad | M | Tiempo de servicio | Disponibilidad del sistema para ser usado en cualquier momento |
| Tolerancia a fallos | M | Redundancia | Capacidad que tiene el sistema de seguir funcionando en caso de que uno de sus componentes falle (Base de datos) |
| Recuperabilidad | B | N/A | N/A |
| Eficiencia del desempeño | M | 15% | Comportamiento temporal | M | Tiempo de respuesta | El tiempo estimado para  completar una tarea |
| Tiempo de espera | El tiempo en completar un  trabajo completo con el  sistema |
| Rendimiento | La cantidad de tareas que  pueden ser procesadas |
| Utilización de recursos | B | N/A | N/A |
| Capacidad | B | N/A | N/A |
| Facilidad de uso | M | 15% | Capacidad de reconocer su adecuación | B | N/A | N/A |
| Capacidad de ser entendido | A | Efectividad de la  documentación del  usuario o ayuda del  sistema | Funciones que son descritas en  la documentación del usuario o  ayuda del sistema |
| Operatividad | M | Claridad de mensajes | Que tan entendibles son los  mensajes del sistema que se  muestran al usuario |
| Protección frente a errores de usuarios | B | N/A | N/A |
| Estética de la interfaz de usuario | B | N/A | N/A |
| Seguridad | M | 15% | Confidencialidad | B | N/A | N/A |
| Integridad | B | N/A | N/A |
| No repudio | A | Utilización de firma digital | El usuario y la contraseña cuentan con un sistema de cifrado |
| Responsabilidad | B | N/A | N/A |
| Autenticidad | B | N/A | N/A |
| Compatibilidad | A | 20% | Co-Existencia | B | N/A | N/A |
| Interoperabilidad | A | Capacidad de interactuar con otro sistema | El sistema interactúa con la base de datos correctamente |
| Mantenibilidad | A | 20% | Modularidad | A | Seccionamiento del sistema en módulos | El sistema se encuentra organizado en módulos comprensibles |
| Reusabilidad | B | Reusabilidad de los módulos del sistema | Los módulos del sistema pueden ser reusables |
| Capacidad de ser analizado | A | Estructura y orden del sistema | El sistema se encuentra ordenado y es entendible para cualquier programador |
| Capacidad de modificación | A | Módulos independientes | La organización del sistema permite modificar un módulo sin afectar negativamente a otros |
| Capacidad de ser probado | A | Se pueden establecer criterios de prueba | Se pueden aplicar las pruebas necesarias al sistema para verificar su calidad |
| Portabilidad | NA | NA | Adaptabilidad | NA | N/A | N/A |
| Facilidad de instalación | NA | N/A | N/A |
| Capacidad de ser reemplazado | NA | N/A | N/A |

**Tabla 5.** Subcaracterísticas de la calidad externa del sistema.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SUBCARACTERÍSTICAS Y ATRIBUTOS DE CALIDAD EXTERNA | | | | | | |
| Características | Nivel de Importancia | Ponderación | Subcaracterísticas | Nivel de Importancia | Métricas | Significado |
| Adecuación funcional | A | 35% | Completitud funcional | A | Completitud de la implementación funcional | Qué tan completa es la implementación de acuerdo con la  especificación de  requerimientos funcionales y no funcionales |
| Exactitud funcional | M | Presión computacional | La frecuencia con que ocurren  los resultados inexactos |
| Fiabilidad | A | 15% | Madurez | A | Eliminación de errores | Cuántos posibles errores han sido corregidos |
| Cobertura de pruebas | Cuántos casos de prueba han  sido ejecutados durante la  etapa de pruebas |
| Disponibilidad | M | Tiempo de servicio | Disponibilidad del sistema para ser usado en cualquier momento |
| Tolerancia a fallos | M | Redundancia | Capacidad que tiene el sistema de seguir funcionando en caso de que uno de sus componentes falle (Base de datos) |
| Recuperabilidad | B | N/A | N/A |
| Eficiencia del desempeño | M | 10% | Comportamiento temporal | M | Tiempo de respuesta | El tiempo estimado para  completar una tarea |
| Tiempo de espera | El tiempo en completar un  trabajo completo con el  sistema |
| Rendimiento | La cantidad de tareas que  pueden ser procesadas |
| Utilización de recursos | B | N/A | N/A |
| Capacidad | B | N/A | N/A |
| Facilidad de uso | M | 10% | Capacidad de reconocer su adecuación | B | N/A | N/A |
| Capacidad de ser entendido | A | Efectividad de la  documentación del  usuario o ayuda del  sistema | Funciones que son descritas en  la documentación del usuario o  ayuda del sistema |
| Operatividad | M | Claridad de mensajes | Que tan entendibles son los  mensajes del sistema que se  muestran al usuario |
| Protección frente a errores de usuarios | B | N/A | N/A |
| Estética de la interfaz de usuario | B | N/A | N/A |
|  |  |  |  |
| Seguridad | M | 10% | Confidencialidad | B | N/A | N/A |
| Integridad | B | N/A | N/A |
| No repudio | A | Utilización de firma digital | El usuario y la contraseña cuentan con un sistema de cifrado |
| Responsabilidad | B | N/A | N/A |
| Autenticidad | B | N/A | N/A |
| Mantenibilidad | A | 20% | Modularidad | A | Seccionamiento del sistema en módulos | El sistema se encuentra organizado en módulos comprensibles |
| Reusabilidad | B | Reusabilidad de los módulos del sistema | Los módulos del sistema pueden ser reusables |
| Capacidad de ser analizado | A | Estructura y orden del sistema | El sistema se encuentra ordenado y es entendible para cualquier programador |
| Capacidad de modificación | A | Módulos independientes | La organización del sistema permite modificar un módulo sin afectar negativamente a otros |
| Capacidad de ser probado | A | Se pueden establecer criterios de prueba | Se pueden aplicar las pruebas necesarias al sistema para verificar su calidad |
| Portabilidad | NA | NA | Adaptabilidad | NA | N/A | N/A |
| Facilidad de instalación | NA | N/A | N/A |
|  |  |  |  |

### **2.2.1 Preparación de los requerimientos de evaluación**

En el diagrama anterior se muestran los requerimientos necesarios y donde se aplica la calidad interna y externa. Actualmente el proyecto ya tiene el 100% del documento de requerimientos. El diseño ya se encuentra en un 100%. El problema en estos momentos es la cantidad de código que lleva desarrollado el proyecto. El código se encuentra como en un 10% y las pruebas apenas en un 5%, por lo que esta complicado terminar de aplicar la calidad interna. La idea es tener todo establecido para efectuar todo el plan de pruebas a todos los documentos del proyecto. También las métricas del modelo deben ser establecidas.

# **3. Ambiente de pruebas**

En esta sección se define como es que se van a hacer las pruebas. Para las pruebas se debe especificar con que se van a hacer las pruebas, todas las herramientas a usar, incluyendo los softwares y librerías, así como las características del dispositivo en donde se llevaron a cabo y en la manera en que se llevaron a cabo.

## **3.1 Software para la ejecución de pruebas**

En este caso el proyecto se esta desarrollando dentro del IDE PyCharm, por lo que aquí mismo es donde se van a desarrollar las pruebas. Afortunadamente existen librerías dentro de Python que ayudan a realizar las pruebas y poder hacer pruebas programadas para poder realizarlas una vez que se haya avanzado más en el proyecto para comprobar que no se agregaron errores al sistema después de haber avanzado mas en el desarrollo. Una de las ventajas de PyCharm es que se pueden tener entornos virtuales en los que uno elige que librerías tener, por lo que es fácil encontrar que versiones de las librerías a utilizar no son compatibles entre sí. Aquí mismo podremos ir guardando los resultados de las pruebas e ir haciendo las correcciones en el código.

## **3.2 Librerías utilizadas para la ejecución de pruebas**

Ahora que ya se sabe donde se va a trabajar es necesario especificar como se va a hacer. Para la calidad interna del software es necesario hacer las pruebas unitarias o de caja blanca. Como se ha mencionado anteriormente son importantísimas y se deben ir diseñando y ejecutando en la medida que el software se va desarrollando para no llegar a un punto muy avanzado y que el software presente algún error y encontrar la causa sería muy complicado para ese punto. De igual manera cabe resaltar que van a haber pruebas de integración o de caja negra necesarias para asegurar que los módulos desde el punto de vista del usuario si estén trabajando en conjunto una vez que esté terminado. Aunque para las pruebas de la calidad externa serán mas de manera manual, estas son igual de importantes para lograr encontrar alguna falla en el software. A continuación, se detallan las librerías mas importantes para la realización de las pruebas en la calidad de software, que en este caso serán por medio del framework Pytest, pero tomando en cuenta también el estándar PEP 8 establecido para darle una estructura al código por la librería pycodestyle.

* **Pycodestyle 2.7.0:** Esta librería se usa en python para revisar que el código se encuentre dentro de las convenciones del PEP 8, establecidas para el lenguaje Python. Las convenciones son una guía que sirven para estructurar de una mejor manera la programación en Python que ha venido cambiando desde que se empezaron a implementar. Esta librería esta más apegada a la estructura del código, como los espacios libres entre líneas de código, como se deben de poner las líneas que no alcancen a escribirse dentro del límite de caracteres por línea, entre otros. Afortunadamente esta librería ya viene con las nuevas versiones de Python por lo que no se tiene que instalar y PyCharm viene con la función para formatear el código escrito bajo este estándar.
* **Pytest 7.1.1:** Un framework muy popular para la creación de pruebas unitarias dentro de la comunidad de Python la cual cuenta con gran apoyo. Provee una información detallada de las fallas, es fácil de implementar, tiene funciones para manejar tanto pequeñas cómo pruebas largas. Sobre todo esto, lo importante es que se pueden crear documentos para realizar las pruebas de manera automáticamente y obtener un reporte.

## **3.3 Software para el control de versiones**

Para todo proyecto es importante ir llevando un control de las versiones que se vayan generando, tanto para la documentación general del proyecto como para el código desarrollado.

# **Bibliografía**