



## Control de documento

Nombre del proyecto	Perfinance
Cierre de iteración	C5
Generador por	Eduardo Ivan Guerrero Hernandez
Aprobado por	Héctor Manuel Chávez De la Vega
Alcance de la distribución del documento	Control interno para todo el proyecto.



## Índice

<b>Sobre este documento .....</b>	<b>3</b>
<b>Resumen de la iteración.....</b>	<b>4</b>
Identificación .....	4
Hitos Especiales .....	5
Artefactos y evaluación .....	7
Riesgos y Problemas.....	8
Notas y Observaciones .....	8
<b>Asignación de recursos .....</b>	<b>9</b>
<b>Anexos.....</b>	<b>10</b>
<b>Referencias a otros documentos.....</b>	<b>20</b>
<b>Glosario de términos .....</b>	<b>21</b>



### **Sobre este documento**

La calidad se logra por medio de la revisión constante de las actividades que conducen desde la idea al producto. Al momento del cierre de una iteración es buen momento para hacer un alto, y evaluar lo logrado, los problemas encontrados y los retos a enfrentar.

El presente documento marca el final de la iteración C5 y contiene una evaluación de los artefactos y actividades realizadas durante la misma.

Se recogen también las impresiones y observaciones hechas durante el desarrollo de la iteración, así como el esfuerzo invertido en cada una de las disciplinas involucradas.



## Resumen de la Iteración

### *Identificación*

Código de iteración: I2, E1, C1, T2...

Se suele usar la siguiente convención: I, E, C, T por la inicial de la fase a la que pertenece la iteración: Inicio, Elaboración, Construcción o Transición.

Se sigue con un número o correlativo que cuenta desde uno.

Fecha de inicio y cierre es auto explicativo. Lo mismo con los comentarios, de haberlos.

<b>Código de la iteración</b>	<b>Fase a la que pertenece</b>	<b>Fecha de inicio</b>	<b>Fecha de cierre</b>	<b>Comentarios</b>
C5	Construcción	10 de Octubre 2022	14 de Octubre 2022	No hay comentarios.



### Hitos especiales

- Evaluación de Calidad utilizando los Factores de McCall.

<b>Factor</b>	<b>Métrica</b>	<b>Calificación</b>	<b>Comentario</b>
Corrección	Consistencia	5	La corrección del cumplimiento de objetivos del usuario, será utilizando la documentación, en donde anotaremos los errores que notamos en las pruebas y se corregirán.
Confiabilidad	Simplicidad	4	El software deberá ser simple y fácil de entender para que el sistema sea utilizado por la mayoría de usuarios que no conocen de los softwares.
Usabilidad	Operatividad	4	La aplicación será fácil de utilizar, con una operación muy eficaz y fácil, no requerirá de mucho esfuerzo para aprender a usarlo.
Integridad o Seguridad	Seguridad	4	El programa contará con elementos de protección, como una base de datos supervisada por administradores y también contraseñas encriptadas en la misma para que no sean vulnerables.
Eficiencia o Performance	Eficiencia de Ejecución	2	La aplicación contará con una eficiencia en sus procesos, es decir, no tardará mucho en realizar las tareas.
Portabilidad	Generalidad	3	La aplicación funcionará en diferentes Sistemas Operativos como Windows, Linux y MacOS
Reusabilidad	Independencia del Sistema	3	La aplicación será independiente de otros programas para proteger los datos de las personas.
Interoperabilidad	Estandarización de datos	2	El software administrará sus datos para cuando existan más tareas no ser tan lento o ralentice los equipos por exceso de datos o tareas.
Facilidad Mantenimiento	Auto documentación	3	Mediante la auto documentación, utilizaremos esto para corregir los errores, y mejorar le mismo código, en caso de que el código está en proceso de mantenimiento o mejora.
			El software se actualizará cada cierto tiempo de una forma simple,



Flexibilidad	Simplicidad	1	la aplicación se pausará temporalmente y mostrará la actualización. (Al interrumpir al usuario, con una actualización no es del todo Flexible).
Facilidad de Prueba	Facilidad de auditoría	4	Las pruebas se realizarán varias veces y se verificará que las pruebas sean fáciles de comprobar y así, facilitando la detección de errores.
Total:		35	



### Artefactos y evaluación

Artefacto	Meta (%)	Comentarios
Desarrollo de la Aplicación I.	10%	La Aplicación no está finalizada, se realizó una interfaz visual, sus botones no estan programados aún. La Aplicación se actualizará en sus respectivos Sprint #6 y Sprint #7.
Implementación de la Base de Datos (BackEnd y FrontEnd).	100%	Base de Datos Terminada, puede que necesite actualizaciones o mejoras, en caso de agregar nuevos datos.
GitHub.	100%	No hay Comentarios.
Documentación de Actividades	100%	No hay Comentarios.

Artefacto	Aspecto a evaluar	Evaluación	Comentarios
Desarrollo de la Aplicación I.	Interfaz Visual, de la pantalla de Inicio.	80%	La Interfaz Visual del Inicio puede sufrir mejoras y actualizaciones, falta dedicarle más.
Implementación de la Base de Datos (BackEnd y FrontEnd).	Diagramas, Tablas y Programación.	100%	No hay Comentarios.
GitHub.	Capturas de Pantalla del Github hecho.	100%	No hay Comentarios.
Documentación de Actividades	Documentación.	100%	No hay Comentarios.



### Riesgos y problemas

ID	Riesgo	Descripción	Ejemplos
RSK-01	Desastres Naturales.	Los desastres naturales pueden dificultar las reuniones presenciales.	Las inundaciones pueden hacer que sea difícil reunirnos.
RSK-02	Falla del Suministro Eléctrico.	El equipo no puede trabajar si no hay luz.	Puede surgir un apagón en la ciudad o colonia que no permita al equipo trabajar.
RSK-03	Falla del Suministro de Internet.	Si no hay internet no hay comunicación ni investigaciones para la programación.	La falta de internet hace que se complique la comunicación si hay cambios.
RSK-04	Ausencia del Personal.	La ausencia de algún personal, complicaría nuestros proyectos.	El encargado de programar puede ausentarse indefinidamente retrasando el proyecto.
RSK-10	Documentación Ambigua.	La documentación del código no sea lo suficientemente clara.	La documentación puede ser difícil de entender si no se le toma su tiempo para escribirla.
RSK-17	Caducidad de Licencias.	Vencimiento de las licencias de nuestros programas software utilizados.	Las licencias que utilizamos sean vencidas y no poder continuar con nuestro trabajo.

### Notas y observaciones

Nota 01: Aún no se incluirá el 'Estudio de Viabilidad' ni 'Documentación de la Página Web', son catalogadas como tareas secundarias.

Nota 02: RSK-17 Caducidad de Licencias – Asana ya caduco, por suerte aún no hemos necesitado ningún de los beneficios de la versión premium, pero ahora la aplicación esta más limitada.

Nota 03: Los errores del Github han sido corregidos y ahora se implementó en los anexos capturas de esto, muestra pocas carpetas debido a que sólo muestra las carpetas que tienen al menos un documento.





### Asignación de recursos

Rol	Horas-Hombre	Desempeñado por	Observaciones
Desarrollo de la Aplicación I.	12 Horas.	Alberto Daniel Mireles Soto.	No hay observaciones.
Implementación de la Base de Datos (BackEnd y FrontEnd).	8 Horas.	Pedro López Ramírez.	No hay observaciones.
GitHub.	1 Horas.	Héctor Manuel Chávez de la Vega.	No hay observaciones.
Documentación	3 Horas	Eduardo Iván Guerrero Hernández	No hay observaciones.



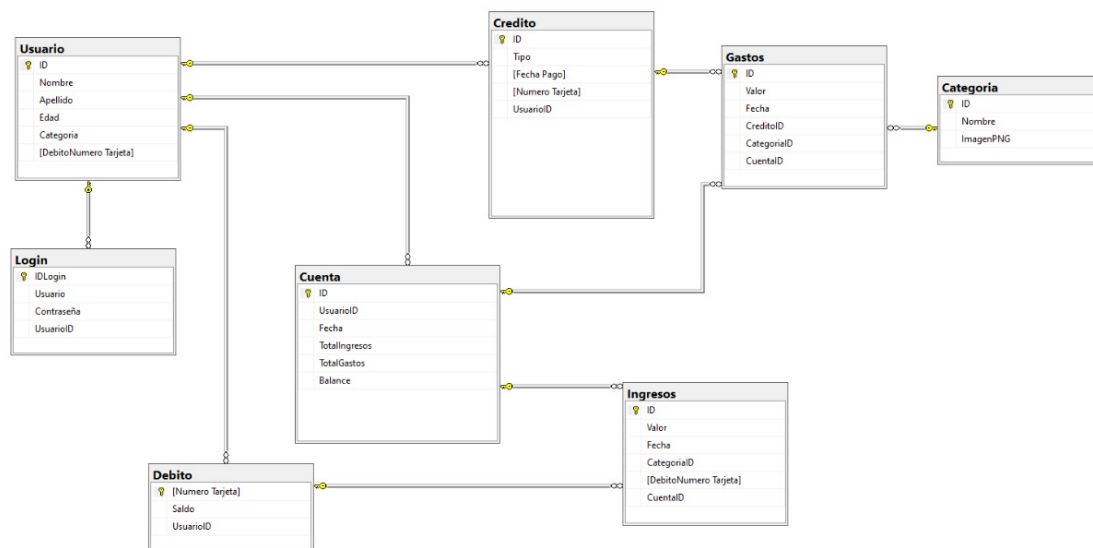
## Anexos

### Anexo A: Desarrollo de la Aplicación I.





## Anexo B: Implementación de la Base de Datos (BackEnd y FrontEnd).







```
CREATE DATABASE PERFINANCE

CREATE TABLE Categoria (
  ID          int IDENTITY NOT NULL,
  Nombre      int NOT NULL,
  ImagenPNG   int NULL,
  PRIMARY KEY (ID));

CREATE TABLE Credito (
  ID          int IDENTITY NOT NULL,
  Tipo        int NOT NULL,
  [Fecha Pago] date NOT NULL,
  [Numero Tarjeta] int NULL,
  UsuarioID   int NOT NULL,
  PRIMARY KEY (ID));

CREATE TABLE Cuenta (
  ID          int IDENTITY NOT NULL,
  UsuarioID   int NOT NULL,
  Fecha       date NOT NULL,
  TotalIngresos int NOT NULL,
  TotalGastos  int NOT NULL,
  Balance     int NOT NULL,
  PRIMARY KEY (ID));

CREATE TABLE Debito (
  [Numero Tarjeta] int IDENTITY NOT NULL,
  Saldo           int NOT NULL,
  UsuarioID       int NOT NULL,
  PRIMARY KEY ([Numero Tarjeta]));

CREATE TABLE Gastos (
  ID          int IDENTITY NOT NULL,
  Valor       int NOT NULL,
  Fecha       date NOT NULL,
  CreditoID   int NOT NULL,
  CategoriaID int NOT NULL,
  CuentaID    int NOT NULL,
  PRIMARY KEY (ID));
```



```
CREATE TABLE Ingresos (
    ID int IDENTITY NOT NULL,
    Valor int NOT NULL,
    Fecha date NOT NULL,
    CategoriaID int NOT NULL,
    [DebitoNumero Tarjeta] int NOT NULL,
    CuentaID int NOT NULL,
    PRIMARY KEY (ID));

CREATE TABLE Login (
    IDLogin int IDENTITY NOT NULL,
    Usuario varchar(10) NOT NULL,
    Contraseña varchar(10) NOT NULL,
    UsuarioID int NOT NULL,
    PRIMARY KEY (IDLogin));

CREATE TABLE Usuario (
    ID int IDENTITY NOT NULL,
    Nombre varchar(255) NOT NULL,
    Apellido varchar(10) NOT NULL,
    Edad int NOT NULL,
    Categoria int NOT NULL,
    [DebitoNumero Tarjeta] int NOT NULL,
    PRIMARY KEY (ID));

ALTER TABLE Ingresos ADD CONSTRAINT FKIngresos537142 FOREIGN KEY (CuentaID) REFERENCES Cuenta (ID);

ALTER TABLE Ingresos ADD CONSTRAINT FKIngresos67708 FOREIGN KEY ([DebitoNumero Tarjeta]) REFERENCES Debito ([Numero Tarjeta]);

ALTER TABLE Gastos ADD CONSTRAINT FKGastos757377 FOREIGN KEY (CuentaID) REFERENCES Cuenta (ID);

ALTER TABLE Gastos ADD CONSTRAINT FKGastos845649 FOREIGN KEY (CategoriaID) REFERENCES Categoria (ID);

ALTER TABLE Gastos ADD CONSTRAINT FKGastos328099 FOREIGN KEY (CreditoID) REFERENCES Credito (ID);

ALTER TABLE Credito ADD CONSTRAINT FKCredito657295 FOREIGN KEY (UsuarioID) REFERENCES Usuario (ID);

ALTER TABLE Login ADD CONSTRAINT FKLogin621390 FOREIGN KEY (UsuarioID) REFERENCES Usuario (ID);

ALTER TABLE Debito ADD CONSTRAINT FKDebito862956 FOREIGN KEY (UsuarioID) REFERENCES Usuario (ID);

ALTER TABLE Cuenta ADD CONSTRAINT FKCuenta621621 FOREIGN KEY (UsuarioID) REFERENCES Usuario (ID);
```





## Anexo C: GitHub.

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).


Owner \*      Repository name \*


 Chahemon / PerfinanceProject 

Great repository names are short and memorable. Need inspiration? How about [improved-fiesta?](#)

Description (optional)

Gestion de Proyectos de Software

☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.



☐  **Private**  
You choose who can see and commit to this repository.

#### Initialize this repository with:

Skip this step if you're importing an existing repository.

- ☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more](#).

**Quick setup — if you've done this kind of thing before**

 Set up in Desktop   or   **HTTPS**   **SSH**   <https://github.com/Chahemon/PerfinanceProject.git> 

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

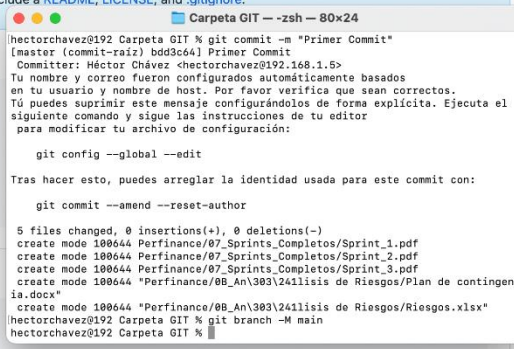
```
echo "# PerfinanceProject" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Chahemon/PerfinanceProject.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/Chahemon/PerfinanceProject.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.



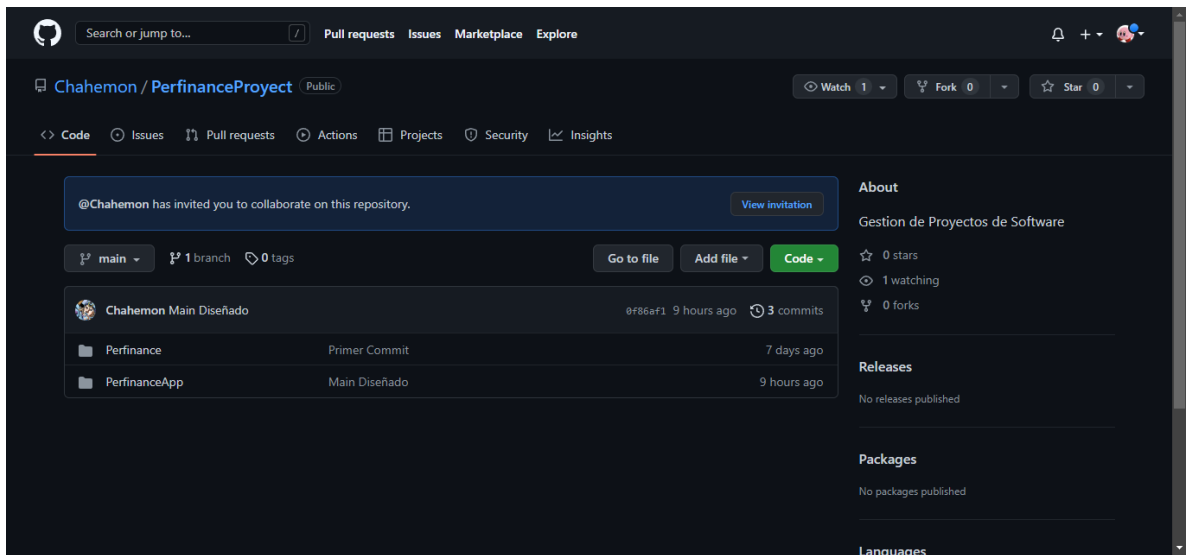
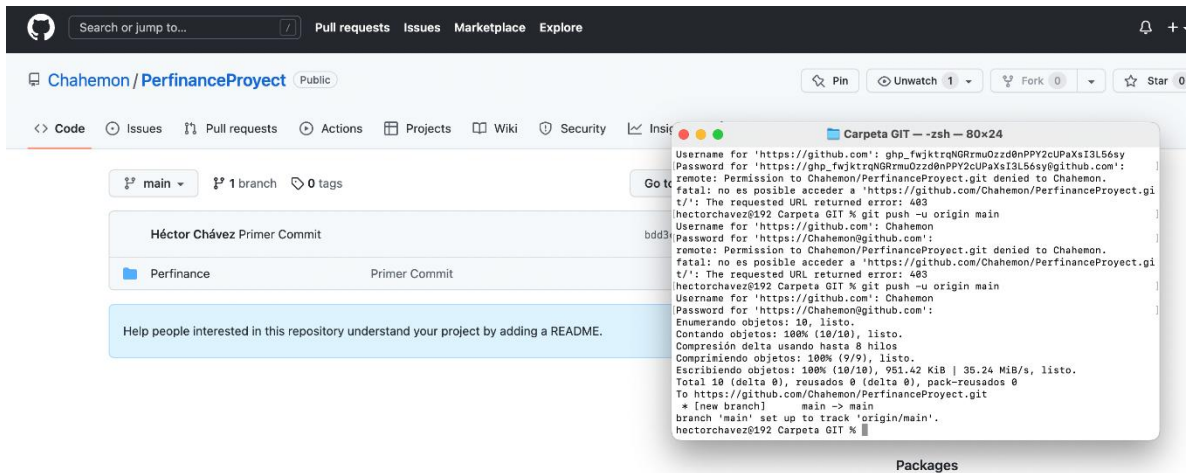
```
hectorchavez@192 Carpeta GIT % git commit -m "Primer Commit"
[master (commit-raiz) bdd3c64] Primer Commit
Committer: Héctor Chávez <hectorchavez@192.168.1.5>
Tu nombre y correo fueron configurados automáticamente basados
en tu usuario y nombre de host. Por favor verifica que sean correctos.
Tú puedes suprimir este mensaje configurándolos de forma explícita. Ejecuta el
siguiente comando y sigue las instrucciones de tu editor
para modificar tu archivo de configuración:

git config --global --edit

Tras hacer esto, puedes arreglar la identidad usada para este commit con:

git commit --amend --reset-author

5 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Perfinance/07_Sprints_Completos/Sprint_1.pdf
create mode 100644 Perfinance/07_Sprints_Completos/Sprint_2.pdf
create mode 100644 Perfinance/07_Sprints_Completos/Sprint_3.pdf
create mode 100644 "Perfinance/08_An\383\241lisis de Riesgos/Plan de contingencia.docx"
create mode 100644 "Perfinance/08_An\383\241lisis de Riesgos/Riesgos.xlsx"
hectorchavez@192 Carpeta GIT % git branch -M main
hectorchavez@192 Carpeta GIT %
```







## Anexo G: Documentación Sprint #5.

### Lunes:

Se introdujeron las ISO para tener un mejor manejo en la calidad de nuestros productos o softwares en este caso, donde tenemos que definir muy bien nuestros factores y administración para que el producto este bien organizado y sea concreto, utilizando estándares, en este día comenzamos a planear sobre cómo hacer los diagramas de la base de datos. [1]

### Martes:

Se introdujeron las métricas y requisitos de un proceso de evaluación, definiendo las principales características del proceso de evaluación y sus medidas concretas. Otro ejemplo que vimos fue Square, donde su principal objetivo es la coordinación y armonización del contenido, para planificar y evaluar la medición que se divide en gestión de calidad, división del modelo de calidad, división de mediciones de calidad, división de requisitos de calidad, división de evaluación, y los estándares.

CCMI es utilizado para mejorar las actividades de un proyecto, IEEE es utilizado para mejorar el trabajo en equipo, PSP se centra en las prácticas de trabajo, TSP busca producir productos de calidad y de bajo costo, MoProSoft se enfoca en estandarizar las operaciones y prácticas en la gestión de la ingeniería de software. [1]

### Miércoles:

Tendríamos una presentación de los estándares en la documentación, los cuales se dividen en diferentes tipos de estándares:

- Como el proceso donde se documenta las etapas y actividades.
- Conocer nuestro documento, organizarlo, estructurarlo, presentarlo y actualizarlos.
- El conjunto de macros que delimitan el uso de letras, colores, tamaño de imágenes y usar un contenido sólido sin cambiarlo.

En esto utilizaremos las métricas de calidad de software como la medida, medición, métrica e indicador, para saber los atributos de nuestro producto, medirlo, conocer los errores y también el código, por último; la métrica o combinación de nuestras métricas.

GQM ayuda a la definición de metas y objetivos de una organización, conociendo nuestro objetivo, preguntas medición e implementación. [1]

### Jueves:

Aprenderíamos sobre las Métricas de Complejidad, la cual permite ver las opciones que tenemos utilizando grafos, mientras que la Métrica del Código Fuente busca saber que tan largo es el código; conociendo sus líneas, longitud y comentarios.



Las Métricas para UML son utilizadas para conocer sobre nuestros diagramas, los Puntos de Función nos ayudan a reconocer el esfuerzo y costo de nuestra aplicación para darle un valor a esta, sobre que tan compleja y eficiente es nuestra aplicación, reconocer los demás atributos que no tomamos en cuenta, como software, hardware, archivos, etc. [1]

Viernes:

No hubo reunión en este día por un evento, pero después tuvimos una reunión fuera de las instalaciones para finalizar los pendientes del sprint, aclarando dudas, tomando capturas de nuestros avances y también el desarrollo del Sprint respectivo (Sprint #5).



## **Referencias a otros documentos**

- [1] L. H. Medina, «Gestión de proyectos de,» 15 Septiembre 2020. [En línea]. Available:  
[https://catedig.itlalaguna.edu.mx/pluginfile.php/810/mod\\_resource/content/5/GPS\\_13.pdf](https://catedig.itlalaguna.edu.mx/pluginfile.php/810/mod_resource/content/5/GPS_13.pdf).  
[Último acceso: 10 Octubre 2022].



## Glosario de términos

**Base de Datos:** Es una recopilación de información donde se almacena en un servidor, siendo controlada por un Sistema de Gestión de Bases de Datos, permite conectar una aplicación a la información que este contiene, para validar o consultar.

**BackEnd:** Es un Sistema Corporativo utilizado por una aplicación donde se puede utilizar el sistema de gestión de pedidos, inventario y procesamiento, recogiendo la información, es conocido como el código y también lo que permite la conexión a las bases de datos.

**FrontEnd:** Es un Sistema Corporativo, el cual un usuario interactúa con la aplicación mediante la interfaz visual, es todo aquello que el usuario puede ver y también accionar, así por decirlo, es la capa exterior de una aplicación.

**GitHub:** Es una Plataforma en la cual los usuarios pueden subir los proyectos y aplicaciones desarrolladas por ellos mismos, permite la colaboración entre los usuarios para que puedan mejorar un código, los proyectos pueden ser descargados y también ser compartidos a otros usuarios.

**ISO (International Organization for Standardization / Organización Internacional de Normalización):** Son normas técnicas internacionales, que buscan contribuir el desarrollo, producción y suministro de los productos, mejorando los productos y también el proceso de estos para dar un mejor producto a un usuario.

**Métricas de Calidad Software:** Son medidas de calificar o verificar el nivel de calidad de un producto, lo importante de estas medidas es que pueden darnos más soluciones y también más información de estas últimas.

**SQuaRE (ISO/IEC 25000):** Son una familia de normas que tienen el objetivo de la creación de un marco de trabajo para evaluar la calidad del producto software.

**CMMI (Capabilty Maturity Model Integration):** Es un conjunto de buenas prácticas organizadas para mejorar el rendimiento de un producto, asegurando la calidad, diseñando productos, entregando servicios, seleccionar y planificar la gestión de trabajo entre otros.

**IEEE:** Es un método de establecimiento para la mejora del trabajo en equipo para procesos software dedicada a la estandarización.

**PSP:** Es un método de autoconocimiento que permite estimar la duración de un software en desarrollo, caracterizándose por su uso personal y su aplicación a los programas pequeños.

**TSP:** Es un método de establecimiento y mejora del trabajo en equipo para los procesos software.

**MoProSoft:** Norma Mexicana que sirve para estandarizar operaciones y prácticas en gestión de ingeniería software elevando la capacidad de las organizaciones de ofrecer mejores productos con calidad.



Código Fuente: Son las instrucciones necesarias realizadas en un lenguaje de programación para compilar y desarrollar una aplicación, es el código utilizado para que una aplicación tenga sus eventos y funciones.