

**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

|  |
| --- |
| **FACULTAD DE INGENIERIA EN ELECTRICIDAD Y COMPUTACION** |

**PROGRAMACIÓN ORIENTADA A OBJETOS**

**INFORME**

“Proyecto I Primer Parcial”

**INTEGRANTES:**

201812914 – Morán Zavala Justo Bautista

201709003 - Plaza Williams Alberto Joel

-

**CARRERA:**

Ingeniería Mecatrónica

**PROFESOR(A):**

Ángela Isabel Carrera Rivera

**PARALELO:** 1

**TÉRMINO:** 2021 - 1S

**GUAYAQUIL – ECUADOR**

Contenido

[1 Introducción 3](#_Toc76418747)

1.1 Objetivo General……………………………………………………………………………………………………………3

1.2 Objetivos Específico ..…………………………………………………………………………………………………….3

[2 Marco Teórico 3](#_Toc76418748)

[3 Diseño del Sistema 4](#_Toc76418749)

[3.1 Abstracción y Encapsulamiento 4](#_Toc76418750)

[3.2 Herencia](#_Toc76418751) 4

[3.3 Polimorfismo 5](#_Toc76418752)

[4 Implementación y Experimentación (si es que hubiere) 5](#_Toc76418753)

[4.1 Implementación 5](#_Toc76418754)

[4.2 Experimentación (si es que hubiere) 5](#_Toc76418755)

[5 Conclusión 5](#_Toc76418756)

[6 Referencias 5](#_Toc76418757)

[7 Anexos 6](#_Toc76418758)

# Introducción

(En esta parte ira el objetivo del Sistema y un poco de los antecedentes. Una breve descripción de que se desarrolló.)

Hoy en día los sistemas cyber físicos y el internet de las cosas hacen posible que tengamos acceso a una gran cantidad de información relevante a nuestro ambiente físico, información tales cuales, como temperatura, humedad, luz, entre otros. Sin embargo, esta información por sí sola no es de gran ayuda si a los datos no se les da un significado.

En este proyecto crearemos un programa que simule el manejo de notificaciones y que haga el razonamiento de la data obtenida por sensores (Archivo iot\_telemetry\_data\_new.csv), de una de las formas más sencillas utilizando boolean, logic y rule-based reasoning que es de manera general y crear condiciones para procesar la información.

* 1. **Objetivo General**
* Desarrollar una programación que simule el manejo de notificaciones, razonamiento de la data obtenida por sensores y condiciones para procesar la información.
  1. **Objetivos Específicos**
* Realizar paquetes adecuados para agrupar clases y el uso necesario de herencia y asociación.
* Subir toda documentación y programación de proyecto en un repositorio en la web junto con la asesoría del docente del proyecto.
* Verificar el cumplimiento de toda la funcionalidad requerida, así como las correctas validaciones de los datos.

# Marco Teórico

**Interfaz de Usuario**: Es la parte del programa que permite al usuario interaccionar con él.

**Clases:** “Son la base de la Programación Orientada a Objetos. Una clase es una plantilla que define la forma de un objeto; en ella se agrupan datos y métodos que operarán sobre esos datos.” (Ing. Palacios, 2015).

**Diagrama de Clase:** “Es una representación gráfica de una clase, en el que se especifica el nombre de la clase, sus atributos y métodos.” (Fundamentos de la Programación Orientada a Objetos, 2007)

**Objeto:** “Corresponden a cosas que se encuentran en el mundo real.” (Tokio School, 2020)

**Paquetes:** “Son el mecanismo que usa Java para facilitar la modularidad del código. Un paquete puede contener una o más definiciones de interfaces y clases, distribuyéndose habitualmente como un archivo.” (Charte, 2019)

**Herencia:** “Es un mecanismo que permite la definición de una clase a partir de la definición de otra ya existente. La herencia permite compartir automáticamente métodos y datos entre clases, subclases y objetos.” (Carlos, 2016)

**Asociación:** “Expresa una relación (unidireccional o bidireccional) entre las instancias a partir de las clases conectadas. El sentido en que se recorre la asociación se denomina navegabilidad de la asociación.” (Garro, 2015)

**Instancia:** “Instanciar objetos es el proceso de generar un ejemplar de una clase, es decir, la clase es como una declaración de una forma y el objeto es un caso o elemento concreto que responde a esa forma.” (Ángel, 2008)

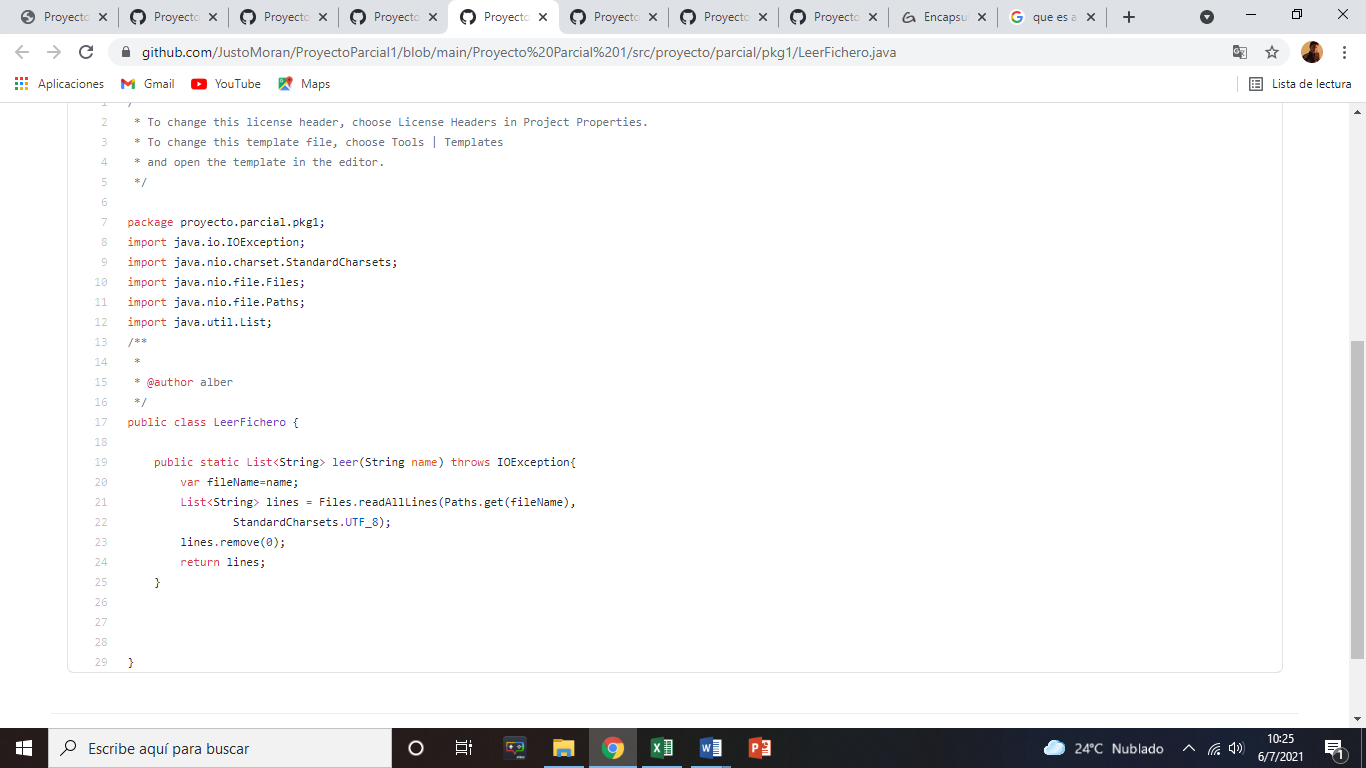
**Javadoc**: “Es una utilidad de Oracle para la generación de documentación de APIs en formato HTML a partir de código fuente Java. Cuando programamos una clase, debemos generar documentación lo suficientemente detallada sobre ella como para que otros programadores sean capaces de usarla sólo con su interfaz.” (Rodríguez, 2017)

# Diseño del Sistema

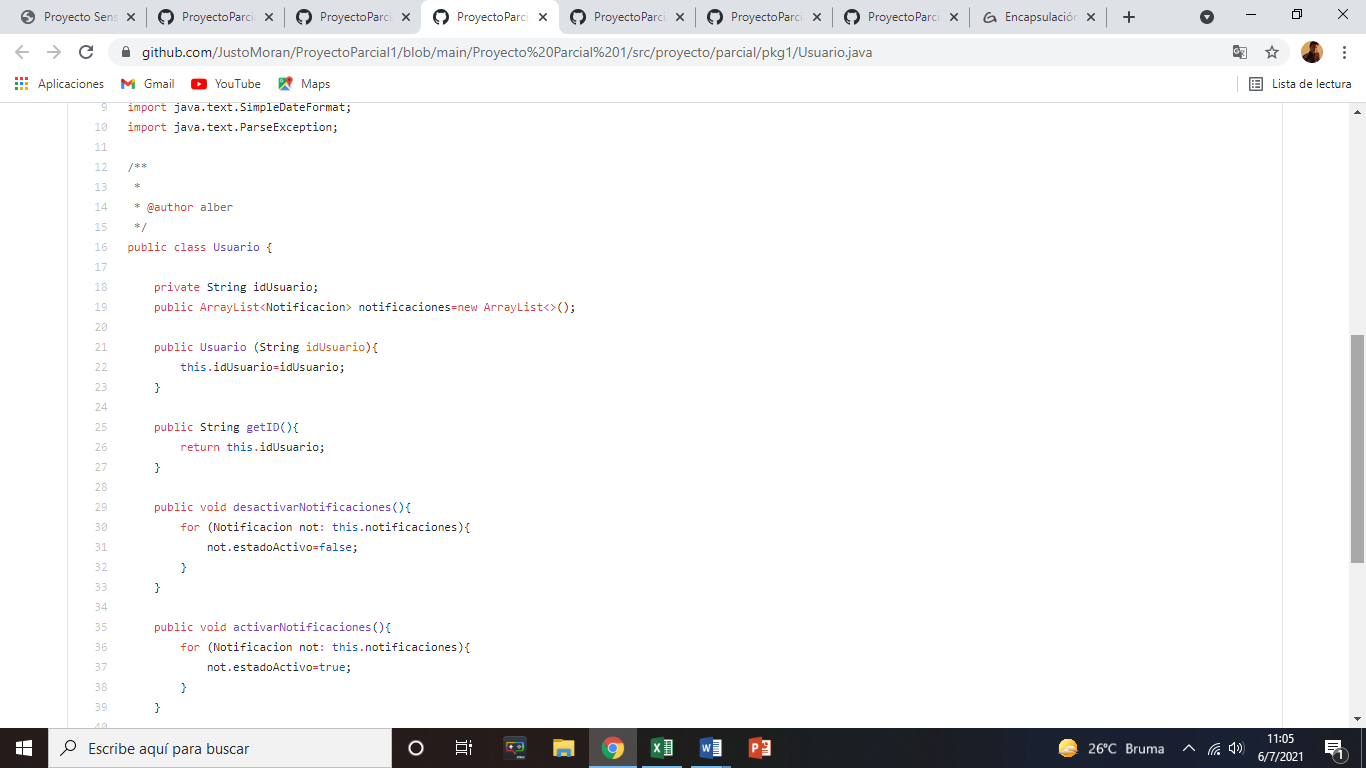
Se desea desarrollar una aplicación donde se muestre datos digitales de las condiciones ambientales de un lugar determinad por defecto y que para ello se necesita acceder con un cuenta de usuario (id) o registrar una nuevo usuario y verificarlo si ya es existente o no; y que por defecto las notificaciones estén activas, ya que por lo menos debe tener el usuario configurado 1 notificación activa de propiedad observable para que el sistema funcione.

## Abstracción y Encapsulamiento

El programa consta de una base de datos o un archivo con sensores ambientales que a su vez estas registran “Propiedades observables”, cada sensor evalúa 7 propiedades observables como “co”, “humidity”, “light”, “Ipg”, “motion”, “smoke”, y “temp”, que a su vez registra el momento de la observación en un formato de fecha (dd/MM/yyyy HH:mm). Todo este procedimiento de abstracción lo realizamos en el método “LeerFichero”.

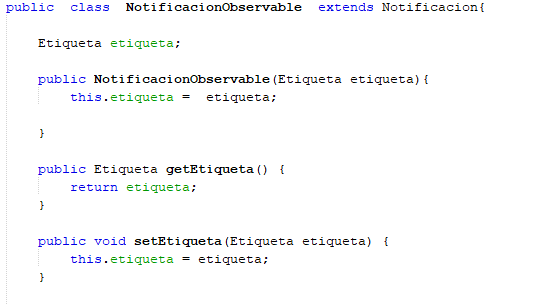


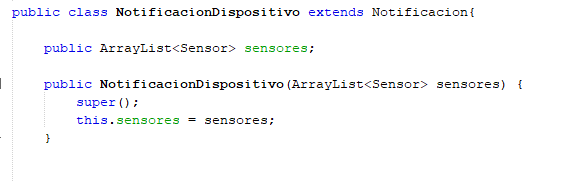
*Imagen n°1: “Bloque de código para obtención de datos”*

Es importante encapsular ciertos datos como por ejemplo el “id” del usuario, ya que si un miembro de datos es privado, significa que solo se puede acceder dentro de la misma clase. Ninguna clase externa puede acceder a miembros de datos privados o variables de otra clase.

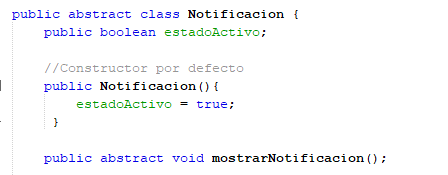
*Imagen n°2: “Encapsulamiento del dato id del usuario”*

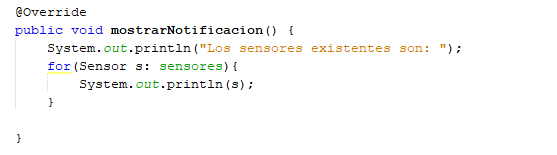
## Herencia

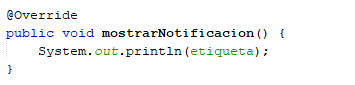




## Polimorfismo



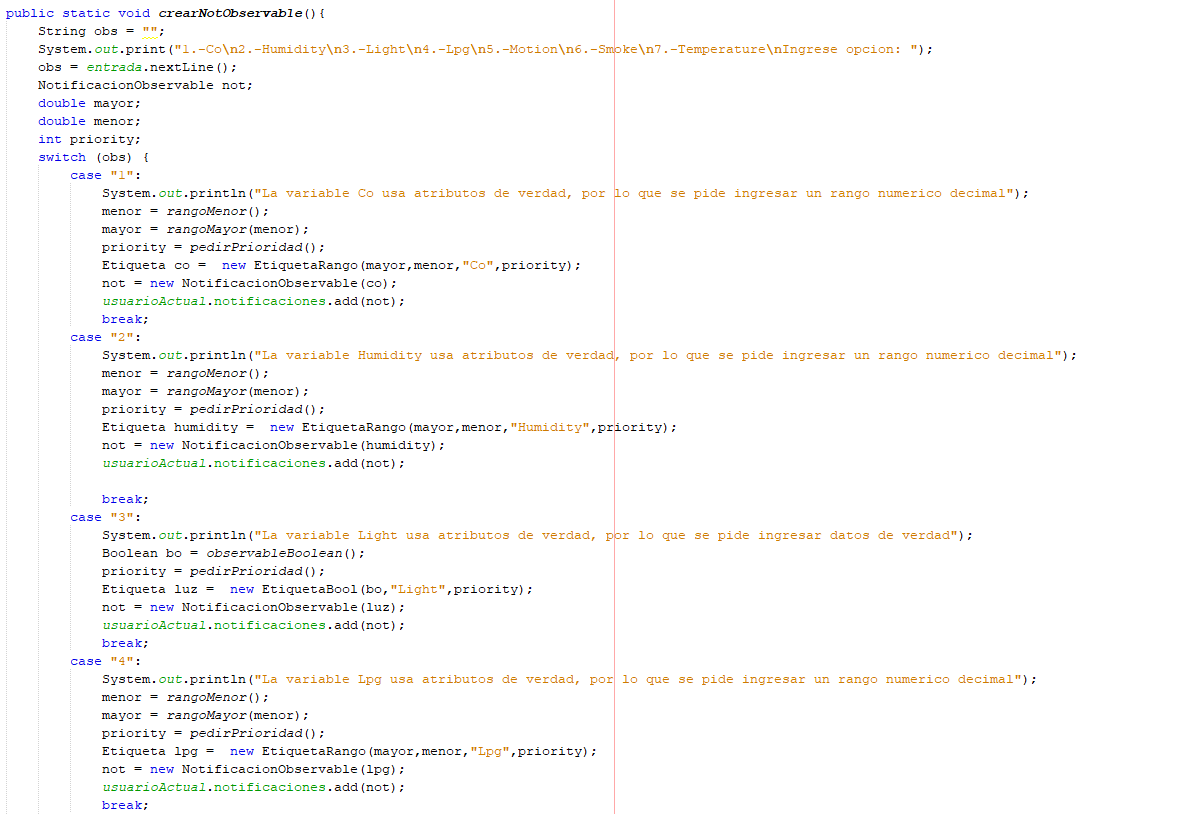




# Implementación y Experimentación (si es que hubiere)

## Implementación

Con respecto a la implementación primero se realizó un análisis de todos los objetos detallados en la problemática, luego de esto se realizo un UML con el fin de tener un diagrama mas claro antes de empezar a programar. Una vez programando se empezó a crear las clases definidas anteriormente, pero se notó que algunas partes realizadas contenían determinados problemas con respecto a lo asumido, entre estos problemas se encuentra que la clase etiqueta recibía parámetros diferentes en algunos casos, por lo que se decidió utilizar herencia para segmentar y afrontar mejor estas clases además de detallar la estructura de control switch para un mejor manejo.



## Experimentación (si es que hubiere)

# Conclusión

Se pudo realizar en el menú principal de la programación la validación del “id” del usuario si está registrado o no en la base de datos, para así proceder a utilizar dicha variable del id para obtener la información de su sensor en la base de datos.

Entre las dificultades al momento de realizar los bloques de código se puede resaltar el entender que la clase Etiqueta debía ser partida en dos debido a sus parámetros, también esta el uso de separar cada caso de las notificaciones y determinar la manera más optima de hacerla.

Como sugerencias a tomar se recalca el uso de herencia en la clase notificación y etiqueta para tener un mejor manejo de los parámetros y no utilizar atributos innecesarios en estas clases, otra sugerencia adicional es el uso de varios métodos para segmentar los diversos casos de notificaciones y como sugerencia final se recomienda tener en cuenta la forma en que se lee los archivos debido a que una mala lectura puede dar problemas cuando el usuario ingrese un sensor pero al momento de leerlo este diferente en el objeto definido.

# Referencias

Ing. Palacios, A.L. (2015) Introducción a la Programación en Java, dirección obtenida en: <http://fcqi.tij.uabc.mx/usuarios/palacios/java4.pdf>

Fundamentos de la Programación Orientada a Objetos (2007) Diagrama de clase, dirección obtenida en: <http://www.utn.edu.ec/reduca/programacion/poo/diagrama_de_clase.html#:~:text=Un%20Diagrama%20de%20Clase%20es,las%20relaciones%20entre%20las%20clases>.

Tokio School (2020) Objeto en Java, dirección obtenida en: <https://www.tokioschool.com/noticias/que-es-objeto-en-java-aprende-programar-java/#:~:text=Pero%2C%20%C2%BFqu%C3%A9%20es%20un%20objeto,de%20compras%2C%20cliente%20y%20producto>.

Charte, Francisco (2019) Paquetes en Java, dirección obtenida en: <https://www.campusmvp.es/recursos/post/paquetes-en-java-que-son-para-que-se-utilizan-y-como-se-usan.aspx>

Rodríguez, Alex (2017) Documentar proyectos Java con Javadoc, dirección obtenida en: <https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=646:documentar-proyectos-java-con-javadoc-comentarios-simbolos-tags-deprecated-param-etc-cu00680b&catid=68&Itemid=188>

Carlos (2016) Programación en Java – Capítulo 3, dirección obtenida en: http://profesores.fi-b.unam.mx/carlos/java/java\_basico3\_4.html#:~:text=La%20herencia%20es%20un%20mecanismo,entre%20clases%2C%20subclases%20y%20objetos.&text=Java%20s%C3%B3lo%20permite%20herencia%20simple.

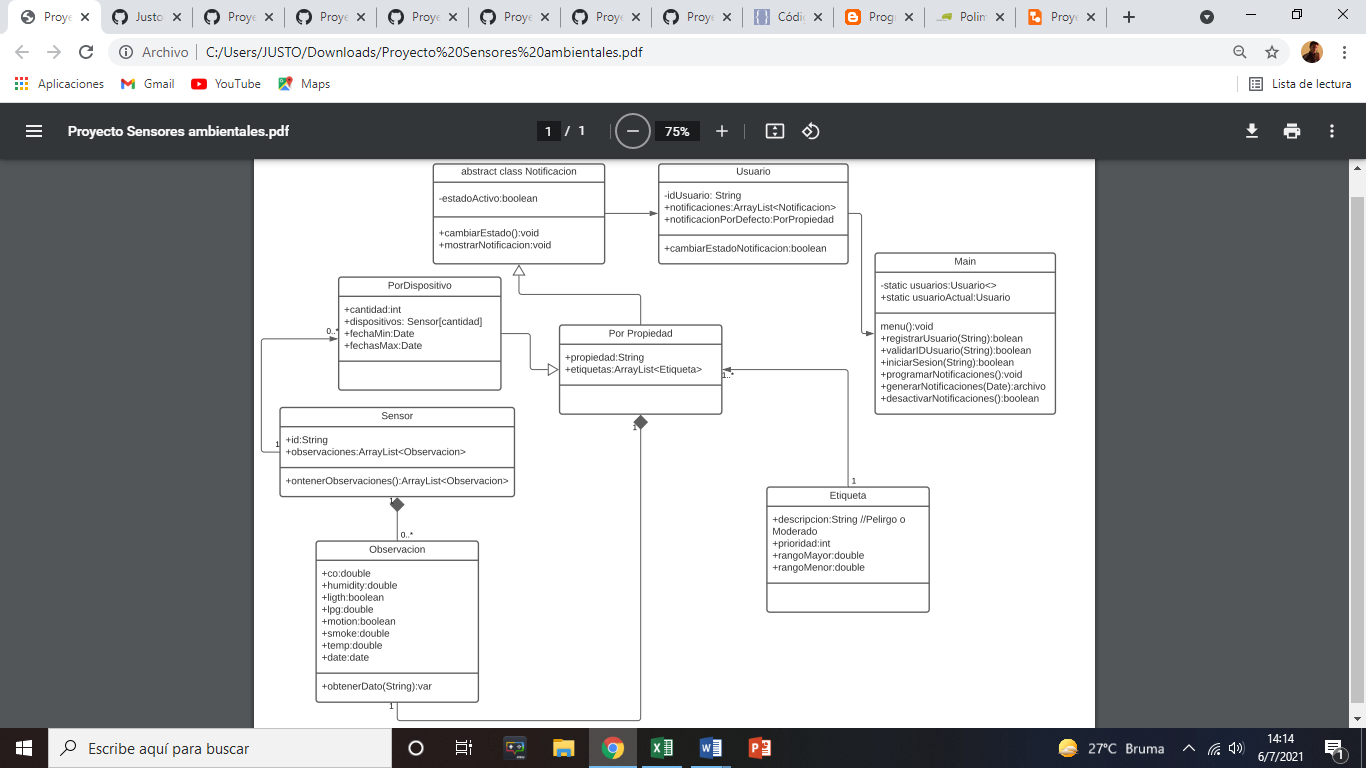
Garro, Arkaitz (2015) Capítulo 17 - Otras relaciones entre objetos, dirección obtenida en: <https://www.arkaitzgarro.com/java/capitulo-17.html#:~:text=En%20una%20asociaci%C3%B3n%2C%20dos%20instancias,s%C3%AD%20existen%20de%20forma%20independiente.&text=La%20relaci%C3%B3n%20de%20asociaci%C3%B3n%20expresa,denomina%20navegabilidad%20de%20la%20asociaci%C3%B3n>.

Ángel Álvarez, Miguel (2008) Instanciar y trabajar con objetos de clases creadas con Mootools, dirección obtenida en: <https://desarrolloweb.com/articulos/instanciar-trabajar-con-objetos-clases-mootools.html#:~:text=Instanciar%20objetos%20es%20el%20proceso,que%20responde%20a%20esa%20forma.&text=Podemos%20crear%20infinitos%20objetos%20a,un%20ejemplar%20de%20esa%20clase>.

# Anexos

* Diagrama de Clases

[https://lucid.app/lucidchart/3c38e86d-b64a-472a-a381-20bf9da480e2/edit?shared=true&page=0\_0#](https://lucid.app/lucidchart/3c38e86d-b64a-472a-a381-20bf9da480e2/edit?shared=true&page=0_0)



* Reporte del Repositorio.

<https://github.com/JustoMoran/ProyectoParcial1.git>