MANUAL TÉCNICO

CAMILA BARONA CABRERA

VALERIA SUÁREZ MEJÍA

SOFÍA URIBE CANO

UNIVERSIDAD EAFIT

INGENIERÍA DE SISTEMAS

ESCUELA DE INGENIERÍA

MEDELLÍN

2018

Contenido

[1. INTRODUCCIÓN 3](#_Toc515824661)

[2. REQUERIMIENTOS TÉCNICOS: 3](#_Toc515824662)

[3. Instalación y configuración: 3](#_Toc515824663)

[4. Herramientas utilizadas para el desarrollo 4](#_Toc515824664)

[4.1. Hardware 4](#_Toc515824665)

[4.1.1. Placa 4](#_Toc515824666)

[4.1.2. Sensores 6](#_Toc515824667)

[4.1.2.1. Sensor DHT11 6](#_Toc515824668)

[4.1.2.2. Módulo LDR 7](#_Toc515824669)

[**Especificaciones técnicas** 7](#_Toc515824670)

[4.2. Software 8](#_Toc515824671)

[4.2.1. Java 8](#_Toc515824672)

[4.2.2. Librerías 8](#_Toc515824673)

[4.2.3. C++ 8](#_Toc515824674)

[5. Funcionamiento del módulo de captura 9](#_Toc515824675)

[6. Funcionamiento módulo de analítica 10](#_Toc515824676)

[7. Módulo de visualización 11](#_Toc515824677)

[8. Módulo de persistencia 13](#_Toc515824678)

[9. CASOS DE USO 15](#_Toc515824679)

[10. diagrama de clases 16](#_Toc515824680)

9. Casos de Uso ..……………………………………………………………………….…13

10. Diagrama de Clases……………………………………………………………………..14

# INTRODUCCIÓN

Este manual presenta los pasos necesarios para que cualquier persona pueda instalar la aplicación creada para la graficación y visualización de datos del servidor provisto por la Universidad EAFIT, así como los aspectos técnicos detallados del mismo.

# REQUERIMIENTOS TÉCNICOS:

La aplicación ha sido empaquetada en un archivo.jar, para la ejecución del mismo se necesita:

* Windows Vista\*
* RAM: 128 MB\*
* Navegador Web: Cualquier navegador de preferencia (para confirmar datos del servidor)
* Java Runtime Environment
* Espacio de disco: 124 MB\*
* Procesador: Mínimum Pentium 2 266 MHz\*

\* Mínimos para la instalación del JRE, no específicos al proyecto

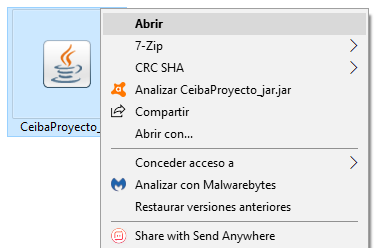
Si se quiere modificar el código, es necesario tener adicionalmente instalado:

* Java Runtime Environment\*
* Java Development Kit\*
* IDE de preferencia\*

\*La aplicación ha sido empaquetada en un archivo.jar, para ejecutar el juego solo se necesita tener el JRE instalado.

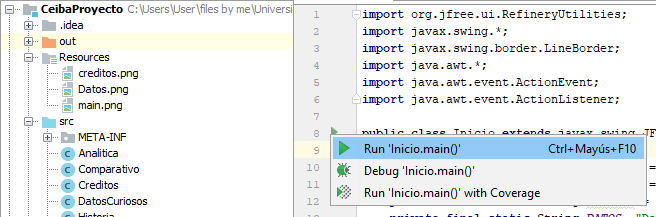
# Instalación y configuración:

Haga doble clic en el archivo .jar del proyecto o clic derecho->abrir



Si desea modificar el código del proyecto descargue el zip del proyecto y ábralo en su IDE preferido.

Para ejecutar el proyecto, ejecute Inicio, la clase principal del proyecto.



# Herramientas utilizadas para el desarrollo

## Hardware

## Placa

Para el proyecto se utilizó una NodeMCU, una placa Open Hardware basada en el chip ESP8266.

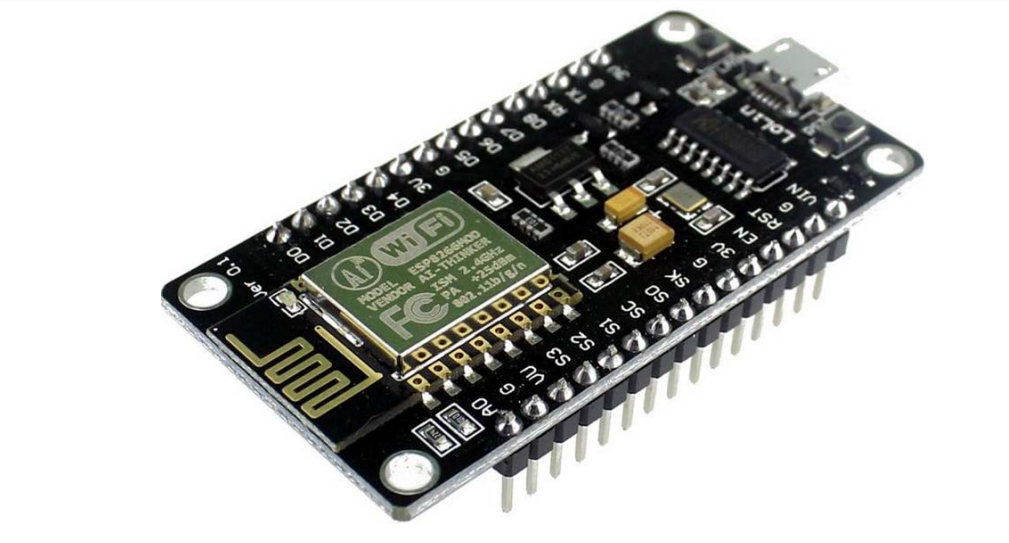
El ESP8266 es el nombre de un microcontrolador diseñado por Espressif Systems. El ESP8266 en sí es una solución de red WiFi autónoma que ofrece un puente desde microcontrolador existente para WiFi y también es capaz de ejecutar aplicaciones autónomas.

Este módulo viene con un conector USB incorporado y una gran variedad de pines.

Con un cable micro USB, puede conectar NodeMCU devkit a su computadora portátil sin ningún problema, al igual que Arduino.

También es inmediatamente compatible con el tablero de pruebas.

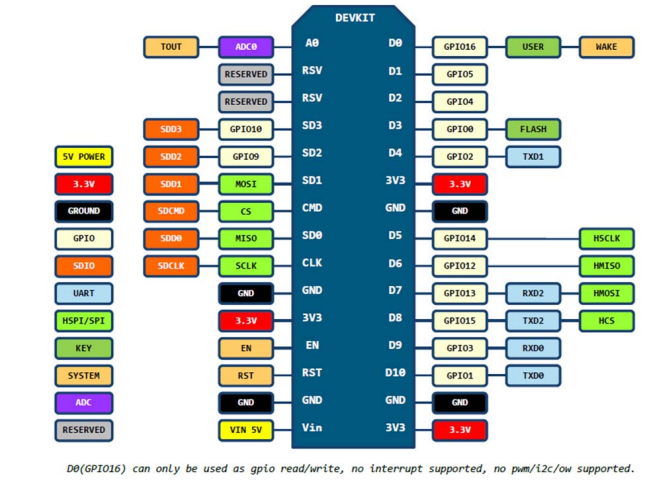
* Desarrollador: ESP8266 Opensource Community
* Tipo: microcontrolador de placa única
* Sistema operativo XTOS
* CPU: ESP8266 (LX106)
* Memoria: 128k Bytes
* Almacenamiento: 4MBytes
* Energía: USB



**Pines del *NodeMCU***

|  |  |  |  |
| --- | --- | --- | --- |
| **Índice IO** | **pin ESP8266** | **Índice IO** | **ESP8266 pin** |
| 0 [\*] | GPIO16 | 7 | GPIO13 |
| 1 | GPIO5 | 8 | GPIO15 |
| 2 | GPIO4 | 9 | GPIO3 |
| 3 | GPIO0 | 10 | GPIO1 |
| 4 | GPIO2 | 11 | GPIO9 |
| 5 | GPIO14 | 12 | GPIO10 |
| 6 | GPIO12 |  |  |

**Estructura de la placa**



## Sensores

## Sensor DHT11

El sensor digital de temperatura y humedad DHT11 es un sensor compuesto que contiene una salida de señal digital calibrada de la temperatura y la humedad.

**Humedad relativa**

* Resolución: 16 bits
* Repetibilidad: ± 1% de HR
* Precisión: a 25 ℃ ± 5% de HR
* Intercambiabilidad: completamente intercambiable
* Tiempo de respuesta: 1 / e (63%) de 25 ℃ 6s
* 1 m / s de aire 6s
* Histéresis: <± 0.3% RH
* Estabilidad a largo plazo: <± 0.5% HR / año

**Temperatura**

* Resolución: 16 bits
* Repetibilidad: ± 0.2 ℃
* Rango: a 25 ℃ ± 2 ℃
* Tiempo de respuesta: 1 / e (63%) 10S
* Características eléctricas
* Fuente de alimentación: DC 3.5 ~ 5.5V
* Corriente de suministro: medida 0.3mA en espera 60μ A
* Período de muestra: más de 2 segundos

**Descripción de pines**

* 1, la fuente de alimentación VDD 3.5 ~ 5.5V DC
* datos en serie de datos, un solo bus
* 3, NC, pin vacío
* 4, suelo GND, la potencia negativa

## Módulo LDR

Un LDR es un dispositivo cuya resistencia varía de acuerdo con la cantidad de luz que reciba. Son muy útiles para proyectos de control de iluminación, seguidores solares, interruptores crepusculares, etc.

Este módulo posee 2 salidas, una analógica que debes conectar a una entrada analógica y así utilizar el conversor ADC. La salida digital posee solo 2 estados: activo/apagado, el cambio de un estado a otro depende del umbral que se fije con el potenciómetro del módulo. La salida digital puede utilizarse para controlar un relay y así realizar una acción dependiente de la intensidad de luz.

**Especificaciones técnicas**

* Voltaje de Operación: 5V DC
* Conexión de 4 cables: VCC, GND, DO, AO
* Salida analógica y digital(comparador)
* Opamp en modo comparador: LM393
* Potenciómetro para ajuste de comparador
* Led rojo de encendido y verde de salida digital

## Software

## Java

Java es un lenguaje de programación creado por Sun Microsystems, (empresa que posteriormente fue comprada por Oracle) para poder funcionar en distintos tipos de procesadores. Su sintaxis es muy parecida a la de C o C++, e incorpora como propias algunas características que en otros lenguajes son extensiones: gestión de hilos, ejecución remota, etc. El código Java, una vez compilado, puede llevarse sin modificación alguna sobre cualquier máquina, y ejecutarlo. Esto se debe a que el código se ejecuta sobre una máquina hipotética o virtual, la Java Virtual Machine, que se encarga de interpretar el código (ficheros compilados .class) y convertirlo a código particular de la CPU que se esté utilizando (siempre que se soporte dicha máquina virtual).

## Librerías

**JFreeChart**

Una biblioteca libre de gráficos de Java. JFreeChart admite gráficos circulares (2D y 3D), gráficos de barras (horizontales y verticales, regulares y apilados), gráficos de líneas, diagramas de dispersión, diagramas de series de tiempo, diagramas alto-bajo-abierto-cerrado, diagramas de velas, diagramas combinados, diagramas de Gantt, termómetros, diales y más. JFreeChart se puede usar en aplicaciones del lado del cliente y del lado del servidor.

Este proyecto es mantenido por David Gilbert.

Nota: esta librería fue utilizada en los gráficos históricos de los datos más no en el gráfico del resumen.

## C++

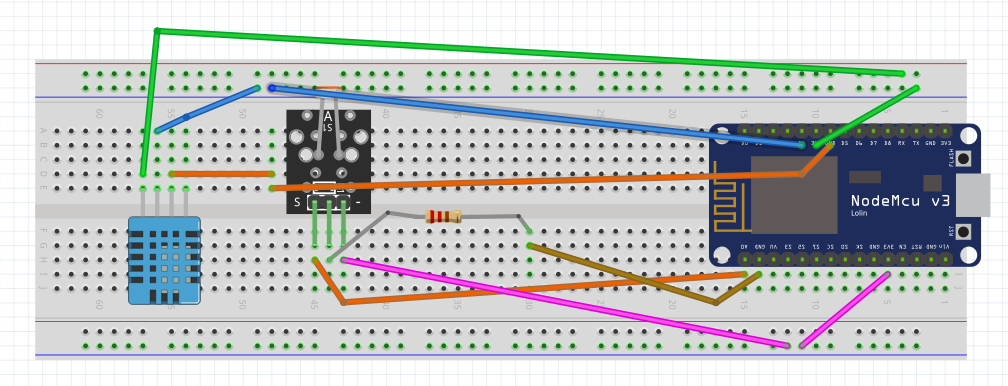
C++ es un lenguaje de programación, creado a mediados de 1980 por Bjarne Strousstrup, como extensión del lenguaje C. Este lenguaje abarca tres paradigmas de la programación:

1. Programación Estructurada
2. Programación Genérica
3. Programación Orientada a Objetos

En la actualidad, C++ es un lenguaje versátil, potente y general. Su éxito entre los programadores le ha llevado a ocupar el primer puesto como herramienta de desarrollo de aplicaciones, ya sea en Windows o GNU Linux, que es el sistema operativo en el cual basaremos este tutorial.

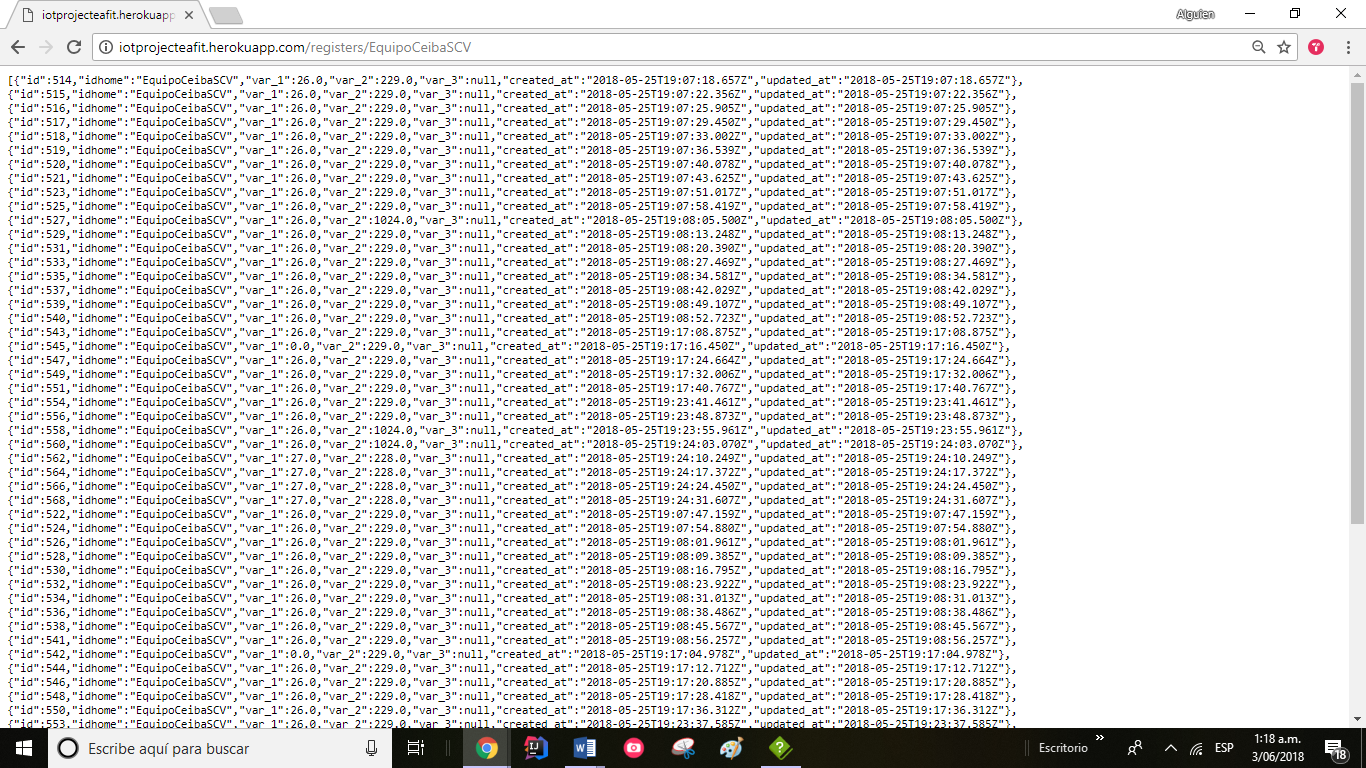
# Funcionamiento del módulo de captura

Primeramente, se conectó la placa de la siguiente manera y después a un computador con un cable USB.



Para la captura de datos se ha implementado un programa en el Arduino IDE, el cual se tiene que configurar para el uso de una placa como la nodeMCU. El programa recolecta datos de temperatura y fotorresistencia cada hora, y posteriormente los envía al servidor.

En el servidor se guarda el valor de la primera variable Temperatura y el de la variable Fotorresistencia, así como la hora en que se guarda, esto se realiza cada hora. Como se muestra en la siguiente imagen.



# Funcionamiento módulo de analítica

Este módulo fue desarrollado en el IDE IntelliJ Idea en el lenguaje JAVA. En el proyecto corresponde a la clase Analítica.

A continuación, se explica brevemente los diferentes procesos que realiza este módulo. Para más información refiérase al [JavaDoc](https://eafit-my.sharepoint.com/:f:/g/personal/msuribec_eafit_edu_co/Em4iA3ctcCxEmumUaaKyHDEB6zbUtEXBedDYKqzuOPgkbg?e=yGad9Q) realizado como expansión de este manual.

|  |  |  |  |
| --- | --- | --- | --- |
| Proceso | Cálculo | Variables para las cuales se realiza | Dato que se lanza |
| Promedio | Calcula el promedio de todos los valores de la variable, actualmente almacenados en el servidor | Temperatura y fotorresistencia | Media aritmética |
| Promedio Recientes | Calcula el promedio de los últimos 24 valores de la variable, actualmente almacenados en el servidor |
| Mínimo | Calcula el mínimo de todos los valores de la variable, actualmente almacenados en el servidor y | Valor mínimo y la hora en que se da |
| Mínimo Recientes | Calcula el mínimo de los últimos 24 valores de la variable, actualmente almacenados en el servidor y |
| Máximo | Calcula el máximo de todos los valores de la variable, actualmente almacenados en el servidor | Valor máximo y la hora en que se da |
| Máximo Recientes | Calcula el máximo de los últimos 24 valores de la variable, actualmente almacenados en el servidor |

# Módulo de visualización

Este módulo fue desarrollado en el IDE IntelliJ Idea en el lenguaje JAVA. En el proyecto corresponde al paquete Visualización.

A continuación, se explica brevemente los diferentes gráficos y ventanas que realiza este módulo. Para más información refiérase al [JavaDoc](https://eafit-my.sharepoint.com/:f:/g/personal/msuribec_eafit_edu_co/Em4iA3ctcCxEmumUaaKyHDEB6zbUtEXBedDYKqzuOPgkbg?e=yGad9Q) realizado como expansión de este manual.

Al abrir la aplicación se observa un menú con los botones de "comparación de las variables", "histórico de las variables", "resumen y análisis", "datos curiosos" y "créditos".



Al presionar cada uno de estos botones se visualiza una ventana, en la siguiente tabla se especifica la función de cada gráfico y su funcionamiento.

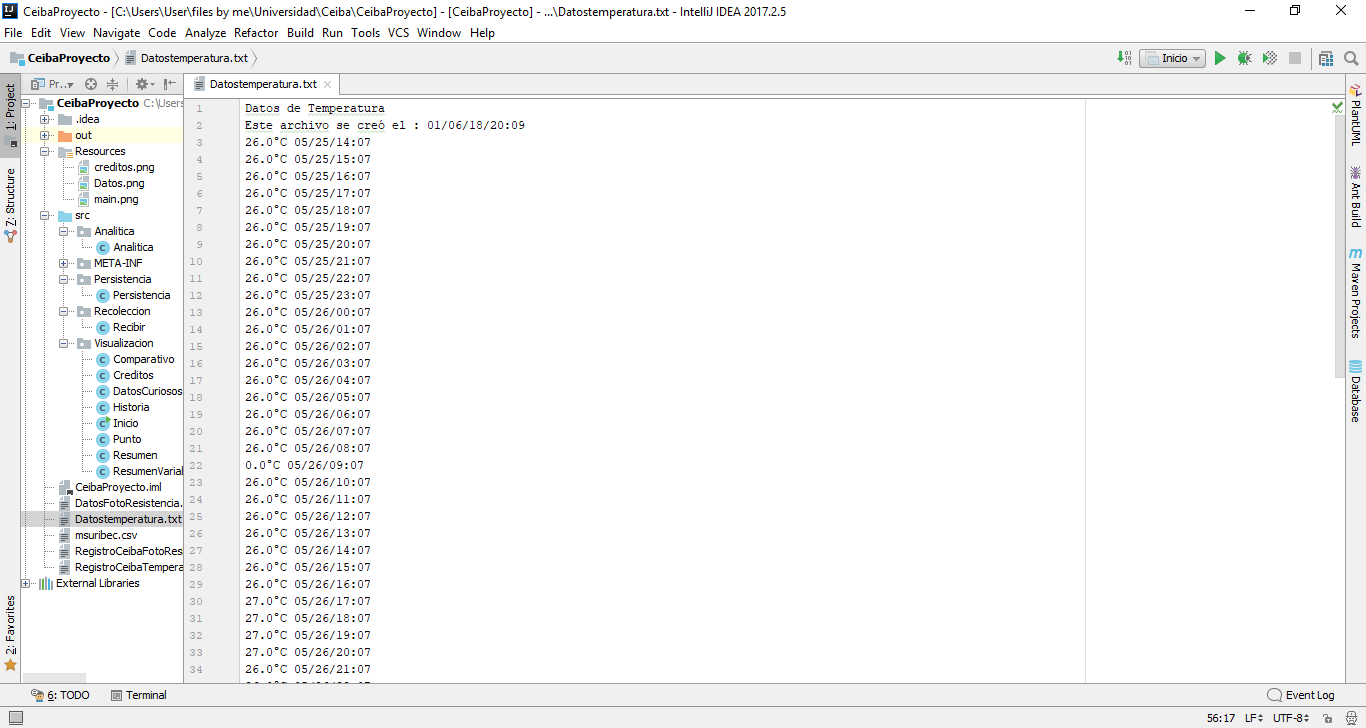
|  |  |  |
| --- | --- | --- |
| Nombre del gráfico | Descripción | Interacción |
| Comparativo | Vista general de ambos datos (Temperatura y fotorresistencia) por hora. | Encerrando en un rectángulo la zona que se quiere ver se puede acercar para ver cada hora y dato con más precisión. |
| Ventana:  Imagen que contiene captura de pantalla  Descripción generada con confianza muy alta | | |
| Histórico | Vista de cada dato de ambas variables que se encuentran en el servidor. | Se debe hacer clic en el botón de “siguiente hora” para avanzar en los datos |
| Ventana: | | |
| Resumen y análisis | Graficas de las últimas 24 horas de cada variable de manera individual, además se muestra un panel con el análisis del comportamiento de la variable. | Se puede seleccionar en el menú desplegable la variable que se desea ver |
| Imagen que contiene mapa, texto  Descripción generada con confianza muy alta | | |
| Datos Curiosos | Ventana que muestra una imagen que contiene información pertinente al porqué se decidió medir estas variables y como esto puede ayudar a mejorar a la ceiba solar. | |
| Imagen que contiene captura de pantalla  Descripción generada con confianza muy alta | | |
| Créditos | Ventana que muestra la información de derechos de autor de las imágenes que aparecen dentro del proyecto y a las librerías utilizadas dentro de la codificación. | |

# Módulo de persistencia

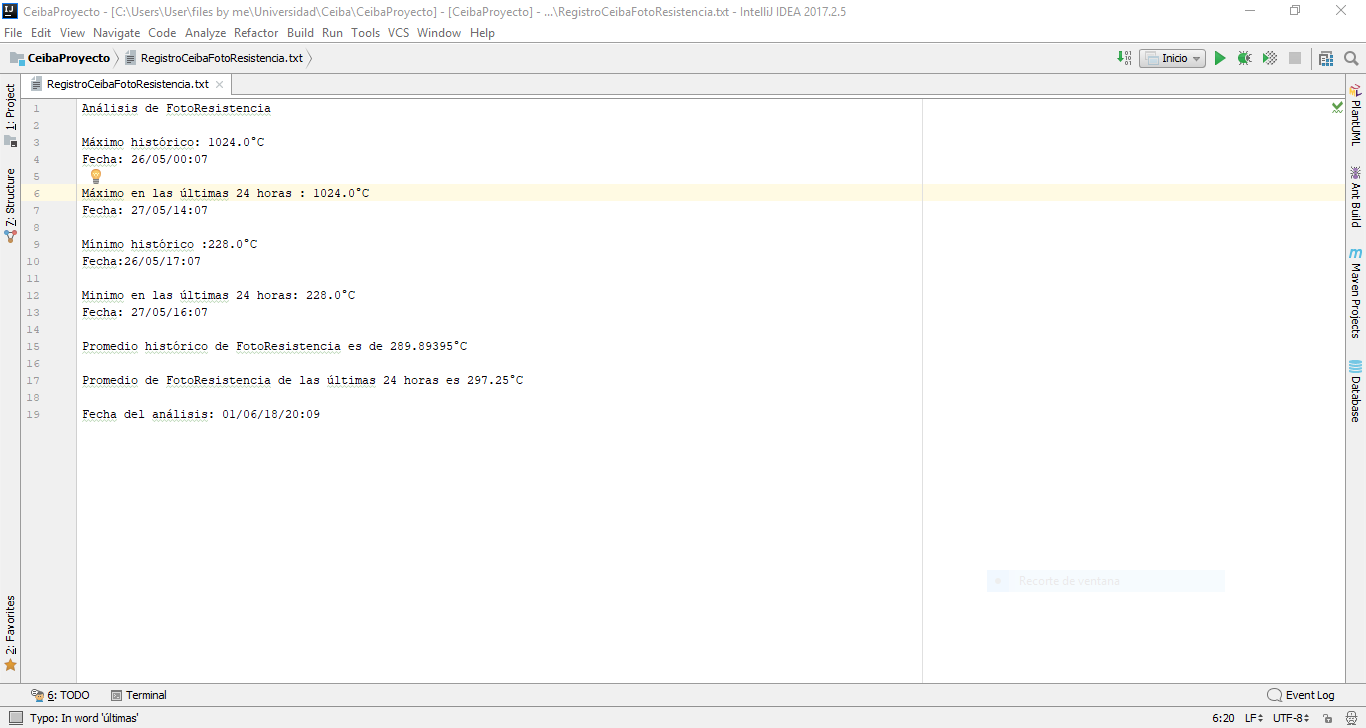
Este módulo fue desarrollado en el IDE IntelliJ Idea en el lenguaje JAVA. En el proyecto corresponde a la clase Analítica.

A continuación, se explica brevemente los diferentes procesos que realiza este módulo. Para más información refiérase al [JavaDoc](https://eafit-my.sharepoint.com/:f:/g/personal/msuribec_eafit_edu_co/Em4iA3ctcCxEmumUaaKyHDEB6zbUtEXBedDYKqzuOPgkbg?e=yGad9Q) realizado como expansión de este manual.

|  |  |  |  |
| --- | --- | --- | --- |
| Proceso | Descripción | Variables para las cuales se realiza | Nombre Archivos |
| Guardar datos  (ejemplo 1) | Guarda en un archivo todos los datos que se encuentran en el servidor al momento de la ejecución, además de la fecha de actualización del archivo | Temperatura y fotorresistencia | Datos<<NombreVariable>> |
| Guardar Análisis  (ejemplo 2) | Guarda en un archivo el análisis de la variable al momento de la ejecución  Además de la fecha del análisis | RegistroCeiba<<NombreVariable>> |

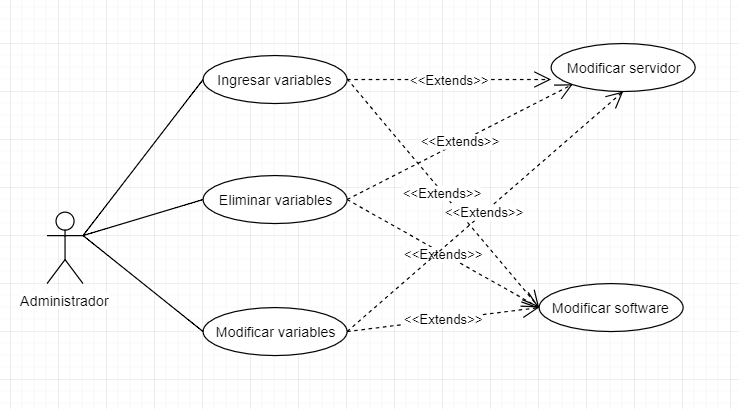
Ejemplo 1

Ejemplo 2



# CASOS DE USO

**Administrador:** Tendrá la labor de ingresar, eliminar, modificar y realizar diferentes acciones sobre los registros de las variables de temperatura y fotorresistencia en la base de datos, y en los códigos implementados en JAVA y en Arduino.



# Diagrama de clases

Imagen que contiene texto

Descripción generada con confianza muy alta