Sistema administrativo para el control de pagos y estudiantes de la Unidad Educativa Henri Pittier

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Firma

Ronald Navas

Caracas, 21/09/2018

Índice General

[Sinopsis 3](#_Toc525671643)

[Planteamiento del Problema 5](#_Toc525671644)

[Objetivo General 6](#_Toc525671645)

[Objetivos Específicos 6](#_Toc525671646)

[Metodología Empleada 7](#_Toc525671647)

[Desarrollo 9](#_Toc525671648)

[Análisis de los Requerimientos 9](#_Toc525671649)

[Diseño del Sistema 9](#_Toc525671650)

[Implementación de la Base de Datos 9](#_Toc525671651)

[Implementación del Sistema 10](#_Toc525671652)

[Pruebas 13](#_Toc525671653)

[Resultados 14](#_Toc525671654)

[Conclusiones y Recomendaciones 15](#_Toc525671655)

[Bibliografía 16](#_Toc525671656)

[Apéndices 17](#_Toc525671657)

# Índice de Figuras

[Figura 2 – Ejemplo de un esquema del patrón DAO 10](#_Toc525679516)

[Figura 3 – Ejemplo codificado del patrón DAO 11](#_Toc525679517)

[Figura 4 – Ejemplo de un esquema del patrón Comando 12](#_Toc525679518)

[Figura 5 – Ejemplo codificado del patrón Comando 12](#_Toc525679519)

[Figura 1 – Modelo Entidad Relación 17](#_Toc525679520)

[Figura 6 – Ejemplo de Stored Procedure 17](#_Toc525679521)

[Figura 7 – Módulo de registro de estudiantes 18](#_Toc525679522)

[Figura 8 – Módulo de registro de representantes 18](#_Toc525679523)

[Figura 9 – Módulo de generación de constancias de estudios 19](#_Toc525679524)

[Figura 10 – Módulo de control de salida de los estudiantes [Fuente Propia] 19](#_Toc525679525)

[Figura 11 – Módulo para generar los boletines informativos de los estudiantes 20](#_Toc525679526)

# Sinopsis

El objetivo del documento es plasmar los avances del desarrollo del sistema administrativo para el control de pagos y estudiantes de la Unidad Educativa Henri Pittier, como también el producto final. Dicho sistema surgió con el fin de automatizar el proceso de registro de los pagos de las mensualidades, como también el registro de los datos de los estudiante y representantes.

Al iniciar cada año escolar, se debe registrar la información de cada estudiante como también la de sus representantes. A su vez, se debe llevar un control para gestionar los pagos de las mensualidades realizados por los representantes. Con la información almacenada, la institución puede realizar distintas tareas como llevar un control de salida de los estudiantes, realizar constancias de estudios y boletines; todas estas tareas se realizan de manera manual ya que no cuentan con un sistema que les facilite el trabajo.

Para solucionar este problema, se desarrollará un sistema administrativo que se adaptará a las necesidades de la institución utilizando una base de datos relacional, con el fin de almacenar toda la información de manera flexible y que sea accesible al momento de consultar algún dato.

# Planteamiento del Problema

La Unidad Educativa Henri Pittier fue fundada en el año 2003, está ubicada en la 3ra avenida de Montalbán I entre la calle 1 y 2, Quinta San Miguel, Distrito Capital, Caracas. Especializada en educación preescolar, cuenta con más de 18 empleados (docentes y personal administrativo) y atienden a más de 80 estudiantes con edades comprendidas entre 1 y 6 años.

La institución educativa, a pesar de que posee más de 10 años de trayectoria, no cuenta con un sistema para poder administrar las finanzas, gestionar RRHH, organizar de manera digital la documentación de la institución, registro de alumnos y representantes, entre otras necesidades.

Dicha institución realiza estas actividades de manera manual, por lo que se pide realizar un sistema en donde se pueda llevar a cabo estas tareas de manera automática, de esta forma poder llevar un mayor control, garantizar disponibilidad de la información, ahorrar tiempo de respuestas a las solicitudes y reducir costos.

## Objetivo General

Desarrollar una aplicación de escritorio para llevar el control de información y gestión de pagos de los representantes y alumnos pertenecientes a la institución U.E Henri Pittier.

## Objetivos Específicos

1. Diseñar e implementar una base de datos para almacenar la información del alumnado y sus respectivos representantes.
2. Desarrollar un módulo para gestionar la información de los estudiantes junto con la de sus representantes.
3. Desarrollar un módulo para gestionar los pagos realizados por los representantes.
4. Desarrollar un módulo para la generación de constancias de estudios de los alumnos.
5. Desarrollar un módulo para controlar los permisos de salida de los estudiantes.
6. Desarrollar un módulo para la creación de boletas del alumnado.
7. Asegurar el acoplamiento de los módulos desarrollados garantizando la unificación del sistema.

# Metodología Empleada

La metodología ágil que se usó para desarrollar el sistema fue eXtreme Programming o XP, ya que ofrece mayor flexibilidad a la hora de realizar cambios a los requisitos, tiene retroalimentación frecuente con el cliente por lo que no hay problemas de comunicación y brinda simplicidad. La metodología XP es la mejor opción para llevar a cabo una programación organizada y planificada, todo con el fin de que no haya errores durante todo el proceso.

XP o eXtreme Programming consta de cuatro (4) fases, las cuales son:

* 1era Fase o Fase de Planificación del Proyecto: En esta fase se realiza una recopilación de todos los requerimientos del proyecto, la primera tarea es definir las historias de usuario por lo que es necesario comunicarse con el cliente, estas se utilizan para estimar tiempos de desarrollo de la aplicación. Luego de esto, se realiza el *Release Planning*, aquí se indicarán las historias de usuario que se crearan para las diversas versiones del sistema, estableciendo fechas de realización y prioridad, en este proyecto se hicieron iteraciones de dos (2) semanas (10 días específicamente). También se estimó la velocidad del proyecto, esto se hace calculando cuantas historias de usuarios se pueden realizar en una iteración. Otro punto de esta fase fueron las reuniones con el cliente, estas se hicieron diariamente, para discutir inconvenientes y buscar posibles soluciones.
* 2da Fase o Fase de Diseño:en esta fase se realizan los diseños que utilizará el sistema, se definen los estándares de diseños que serán utilizados tanto en la base de datos, como en la aplicación, se determinan los riesgos que pudieran ocurrir en las distintas etapas del proyecto, por último, se realiza una refactorización. El objetivo de este último paso es buscar la optimización.
* 3era Fase o Fase de Codificación: En esta fase se procede a desarrollar el sistema. XP (o eXtreme Programming) recomienda programar en parejas, lo cual fue necesario en este proyecto, ya que la otra parte del sistema fue desarrollada por otro estudiante, es por ello que se tuvo que llegar a un acuerdo para determinar la metodología del sistema, normas de uso del repositorio y los estándares de programación, todo con el fin de que el código sea consistente, escalable y sea fácil de comprender.
* 4ta Fase o Fase de Pruebas: El objetivo de esta etapa o fase es evaluar todo el sistema desarrollado y asegurar el cumplimiento de las historias de usuario que se pactaron con el cliente, para ello se realizaran pruebas a los distintos módulos del sistema y se verificará el comportamiento que tiene estos ante diversos escenarios, todo esto con el propósito de que los resultados sean los esperados.

# Desarrollo

## Análisis de los Requerimientos

Para poder solventar el problema que tiene el cliente, se tuvo que pautar una reunión, en donde se determinaron los objetivos a desarrollar. Se estudiaron y analizaron todos los requerimientos requeridos por el cliente, buscando todos los posibles caminos para realizar y solucionar su problema.

## Diseño del Sistema

El sistema consta de dos grandes componentes los cuales son la base de datos y la aplicación de escritorio, para ello se diseñó la forma en la que se almacenarían los datos y la interfaz de usuario. Debido a que la parte restante del proyecto sería desarrollada por otro compañero, se definieron los estándares que seguiríamos tanto para la base de datos como para la aplicación.

El diseño principal de la aplicación de escritorio está conformado por un menú administrativo (realizado con Swing, una biblioteca grafica de Java), en donde se pueden acceder a los distintos módulos del sistema (módulos de empleados, módulos de representantes y módulos de estudiantes, entre otros).

## Implementación de la Base de Datos

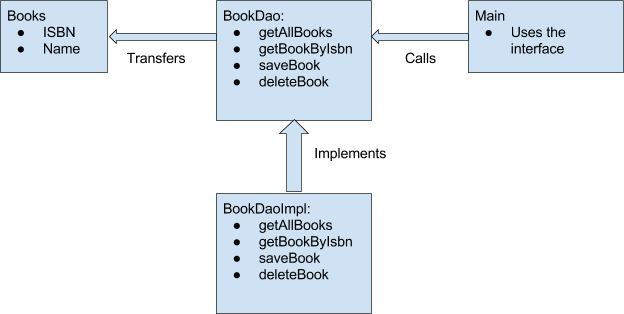
La base de datos que se utilizó fue del tipo relacional, ya que para este proyecto la cantidad de datos a almacenar es reducida, además de que una base de datos relacional ofrece consistencia, atomicidad y durabilidad. Debido a que se eligió una base de datos relacional, se realizó un modelo Entidad Relación o mejor conocido como E/R. Posteriormente se crearon los Scripts y los Stored Procedures, siguiendo los estándares previamente pautados, de esta forma se implementaría la base de datos y los procedimientos que almacenarían la información en la misma.

## Implementación del Sistema

Para separar los datos de la aplicación, la interfaz de usuario y la lógica de control, se utilizó el estilo de arquitectura de software Modelo Vista Controlador (MVC). El sistema a desarrollar fue realizado en Java en su versión 7, con lenguaje de programación Java.

El sistema se separó en una aplicación de escritorio (interfaz de usuario), en donde el cliente registrará y consultará la información almacenada y el Web Service o Servicio Web, en donde habrá una serie de protocolos que permitirá el intercambio de datos, el objetivo del servicio será analizar las peticiones ordenadas por la aplicación de escritorio para luego dar una respuesta. Con el fin de desarrollar una aplicación robusta, fácil de mantener y con control de cohesión y acoplamiento, se utilizaron patrones de diseño los cuales fueron:

* DAO (Data Access Object): Este patrón es utilizado para crear una capa de persistencia. Este encapsula el acceso a la base de datos, por lo que cuando se necesite acceder o interactuar con la base de datos, esto se hará a través de las clases de este tipo. Una gran ventaja es que, ya que la comunicación está aislada de otras clases, a la hora de realizar cambios, las demás clases no sufrirán alteraciones.

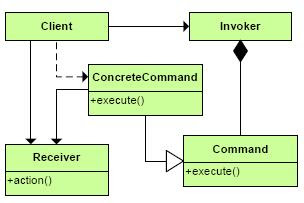


#### Figura 2 – Ejemplo de un esquema del patrón DAO

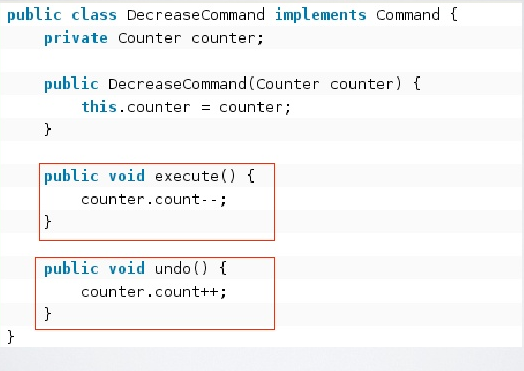


#### Figura 3 – Ejemplo codificado del patrón DAO (Fuente propia)

* Comando: Este encapsula un mensaje como un objeto, especifica una forma sencilla y simple de separar la ejecución de un comando, del entorno que genero el comando. El objetivo del patrón Comando (o Command) es desacoplar el objeto que invoca a una operación de aquel que tiene el conocimiento necesario para realizarla.



#### Figura 4 – Ejemplo de un esquema del patrón Comando



#### Figura 5 – Ejemplo codificado del patrón Comando

## Pruebas

Se realizaron pruebas exhaustivas al sistema para comprobar el buen funcionamiento del mismo, dichas pruebas se hicieron al final de cada iteración y al finalizar el sistema. Los resultados obtenidos de cada prueba comprobaron el buen funcionamiento de cada módulo desarrollado.

# Resultados

El resultado que se obtuvo de este trabajo, fue una aplicación eficaz en donde se automatizaron los procesos administrativos que lleva la Unidad Educativa Henri Pittier, entre algunas tareas tenemos la gestión total de la información de los representantes y estudiantes de dicha institución, el control de pago de mensualidades realizada por los representantes, control de salida de los estudiantes, gestión de constancias de estudios y boletines informativos.

Dicho sistema es capaz de gestionar la información a través de una base de datos relacional, junto con procedimientos almacenados, permiten mantener la integridad, mantenibilidad y seguridad de la información almacenada.

# Conclusiones y Recomendaciones

La institución educativa no contaba con un sistema para poder gestionar la información de los estudiantes y representantes, administrar los pagos realizados y gestionar documentos, dichos procesos se llevaban a cabo de manera manual, por lo que el índice de error era considerablemente elevado. Estos procesos se lograron automatizar con el sistema desarrollado, la institución podrá gestionar de manera eficaz y segura el registro de representantes, estudiantes y pagos de mensualidades, como también consultar todo tipo de información relacionados a ellos, el cliente podrá generar constancias de estudios y boletines informativos, como también llevar un control de salida de los estudiantes.

Algunas de las recomendaciones a consideran son:

* Al iniciar cada año escolar se debe realizar mantenimiento al sistema, todo esto para no almacenar información que no será de utilidad a futuro y asegurar el buen funcionamiento del sistema.
* Para mejorar el funcionamiento del manejador de imágenes, se recomienda convertir las imágenes de los representantes y estudiantes en tipo BLOB (Binary Large Object) para luego ser almacenados en la base de datos
* Realizar búsquedas por autocompletado para facilitar el acceso a la información en el caso de que la cantidad de representantes aumente

# Bibliografía

Ainhoa, L. (s.f.). *Bases de datos relacionales vs. no relacionales: ¿qué es mejor?* Obtenido de https://aukera.es/blog/bases-de-datos-relacionales-vs-no-relacionales/

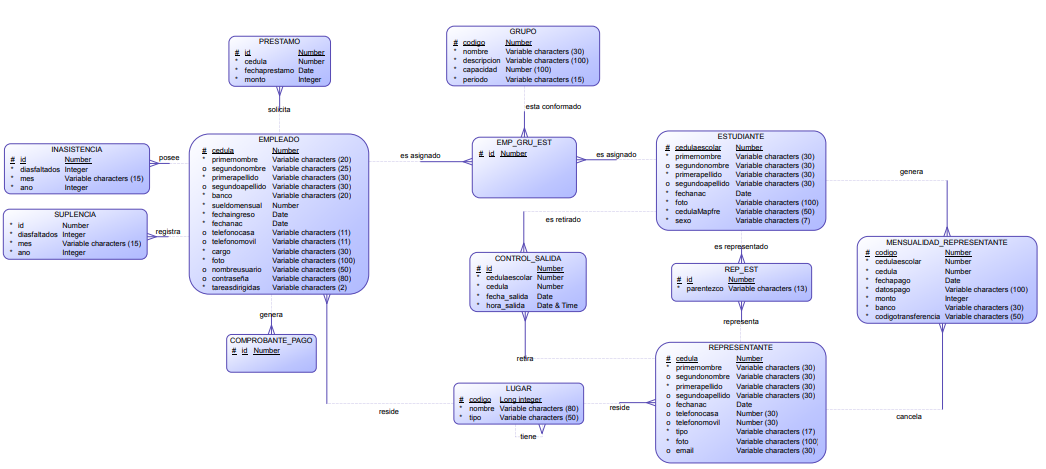
Herrera , O. (29 de Marzo de 2009). *Patron de diseño Commmand.* Obtenido de http://micro-howto.blogspot.com/2009/03/patron-de-diseno-command.html

Juarez, M. (07 de Junio de 2011). *Command.* Obtenido de http://migranitodejava.blogspot.com/2011/06/command.html

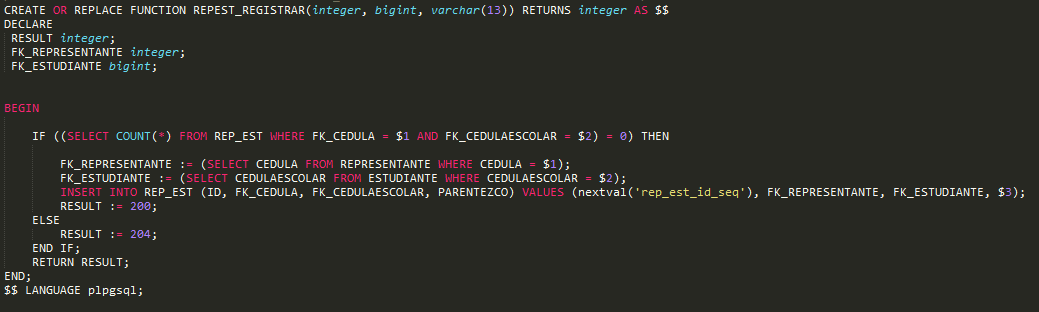
Richardmx. (27 de Mayo de 2008). *Que es Data Access Object.* Obtenido de http://www.javamexico.org/blogs/richardmx/que\_es\_data\_access\_object

Shubham. (2 de Abril de 2018). *DAO Design Pattern.* Obtenido de https://www.journaldev.com/16813/dao-design-pattern

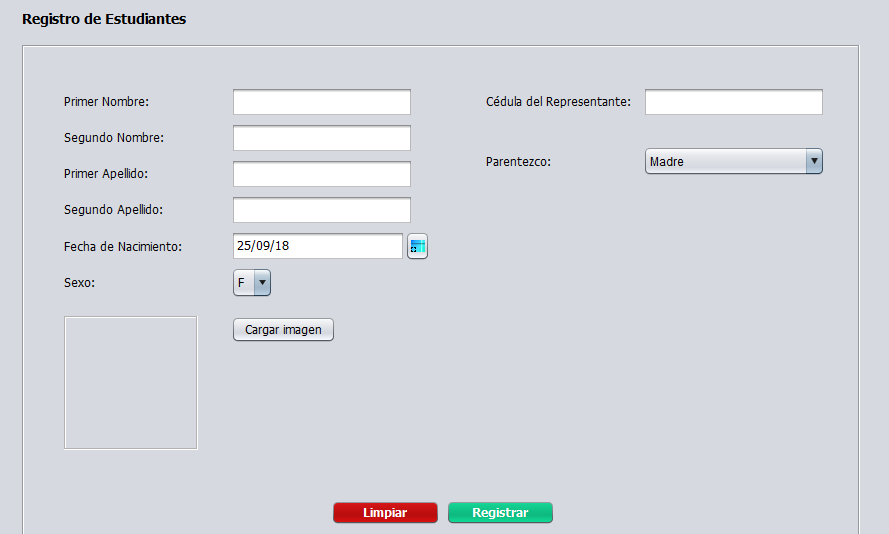
# Apéndices



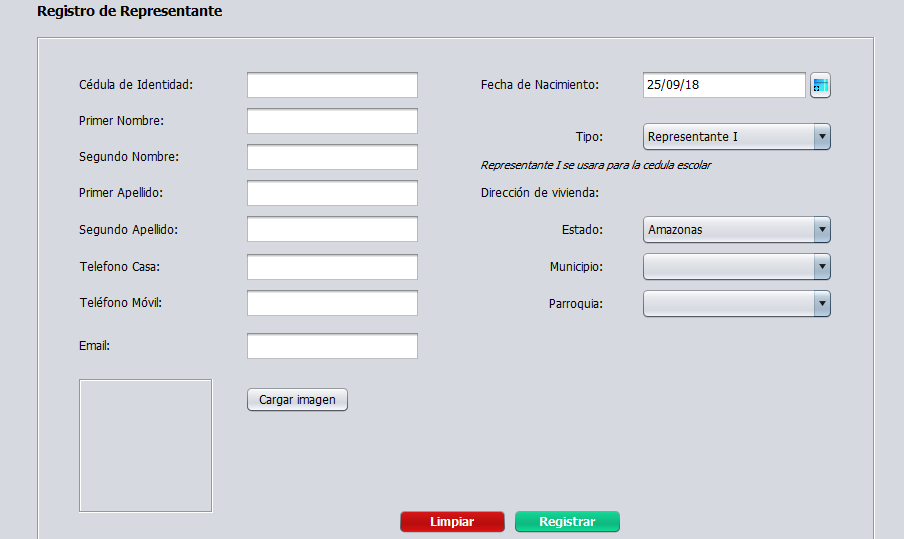
#### Figura 1 – Modelo Entidad Relación [Fuente Propia]



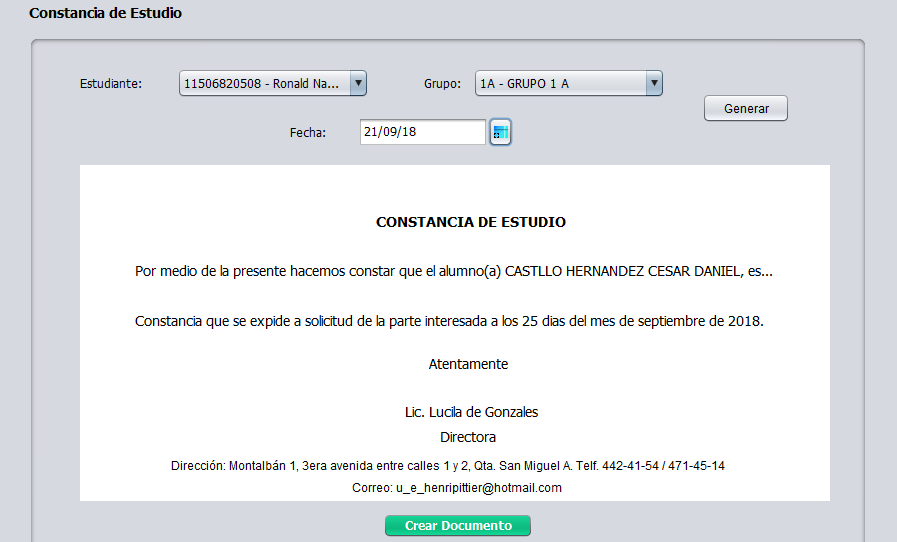
#### Figura 6 – Ejemplo de Stored Procedure [Fuente Propia]



#### Figura 7 – Módulo de registro de estudiantes [Fuente Propia]



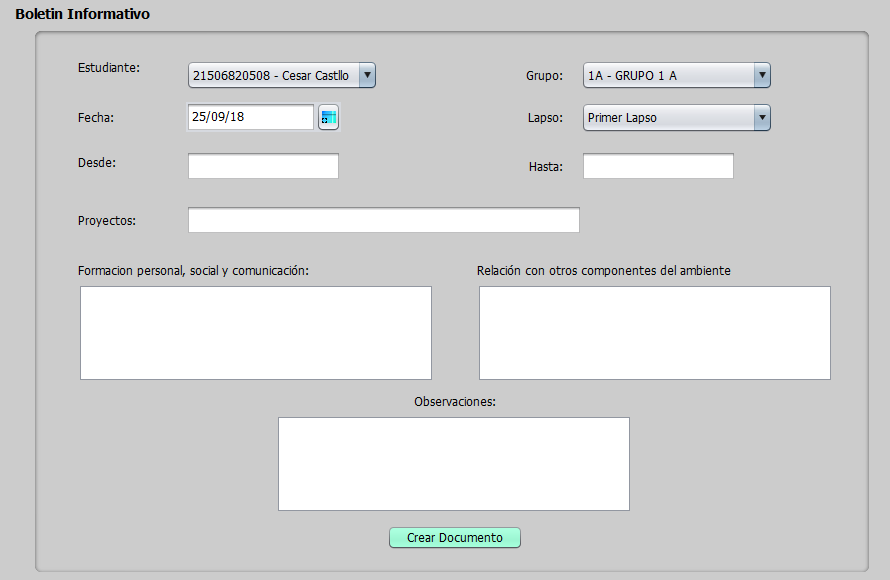
#### Figura 8 – Módulo de registro de representantes [Fuente Propia]



#### Figura 9 – Módulo de generación de constancias de estudios [Fuente Propia]



#### Figura 10 – Módulo de control de salida de los estudiantes [Fuente Propia]



#### Figura 11 – Módulo para generar los boletines informativos de los estudiantes [Fuente Propia]