



# **UNIVERSIDAD AUTÓNOMA DE ENCARNACIÓN**

## **SISTEMA WEB DE GESTIÓN ADMINISTRATIVA PARA EL COLEGIO PARROQUIAL PRIVADO SAN PIO X**

Melanie Evelin Florentin Alcantara

Tesis presentada a la Facultad de Ciencia, Arte y Tecnología como requisito para la obtención del título de Licenciado/a en Análisis de Sistemas Informáticos.

**Encarnación, Paraguay**

**Octubre, 2017**

Melanie Evelin Florentin Alcantara

**SISTEMA WEB DE GESTION ADMINISTRATIVA PARA EL COLEGIO  
PARROQUIAL PRIVADO SAN PIO X**

Tesis presentada a la Facultad de Ciencia, Arte y Tecnología, Universidad Autónoma de Encarnación (UNAE) como requisito para la obtención del título de Licenciado/a en Análisis de Sistemas Informáticos.

Línea de investigación: Informática, Educación y Sociedad.

Orientador/a: Ing. Hugo Sendoa

**Encarnación, Paraguay**

**Octubre, 2017**

**Autorizo la reproducción total o parcial de este trabajo, por cualquier medio convencional o electrónico, para fines de estudio e investigación, siempre que sea citada la fuente.**

Los hechos e ideas expresados en este trabajo de investigación son de responsabilidad exclusiva del/la autor/a.

### **FICHA CATALOGRÁFICA**

Florentin Alcantara, Melanie Evelin (2017). Sistema Web de Gestión Administrativa para el Colegio Parroquial Privado San Pio X. Encarnación, Universidad Autónoma de Encarnación, 72p.

Orientador: Ing. Hugo Sendoa

Tesis presentada a la Facultad de Ciencia, Arte y Tecnología, Universidad Autónoma de Encarnación (UNAE) como requisito para la obtención del título de Licenciado/a en Análisis de Sistemas Informáticos.

Línea de investigación: Informática, Educación y Sociedad

Palabras claves: Gestión Administrativa, sistema Web, Colegio Parroquial Privado San Pio X

MELANIE EVELIN FLORENTIN ALCANTARA

**SISTEMA WEB DE GESTION ADMINISTRATIVA PARA EL COLEGIO  
PARROQUIAL PRIVADO SAN PIO X**

Tesis presentada a la Facultad de Ciencia, Arte y Tecnología, Universidad Autónoma de Encarnación (UNAE) como requisito para la obtención del título de Licenciado/a en Análisis de Sistemas Informáticos.

Línea de Investigación: Informática, Educación y Sociedad  
Orientador/a: Ing. Hugo Sendoa

Aprobado en (lugar) \_\_\_\_\_, el ...../...../.....

Calificación: .... (en números) y letras.....

**Sínodo Examinador**

Prof. \_\_\_\_\_

Firma: \_\_\_\_\_

Prof. \_\_\_\_\_

Firma: \_\_\_\_\_

Prof. \_\_\_\_\_

Firma: \_\_\_\_\_



Agradezco a:

Primeramente a Dios por la oportunidad de haber culminado una etapa en mi formación.

A mi mamá y hermana por su acompañamiento y apoyo constante.

A los profesores y tutor por haber enseñado con profesionalismo.

A mis compañeros de la carrera por haber facilitado el trabajo con su espíritu de compañerismo.

A mis amigos por motivarme a seguir y no desistir.



*“Si quieres hacer algo grande en tu vida, debes recordar que la timidez esta solo en la mente. Si piensas con timidez, tus actos serán similares. Pero si piensas con seguridad, actuaras de la misma forma. Por ello nunca dejes que la timidez conquiste tu mente”* Gates B.



## TABLA DE CONTENIDO

<b>Sistema web de gestión administrativa para el Colegio Parroquial Privado San Pio X.....</b>	<b>4</b>
--	----------

<b>Planteamiento del problema .....</b>	<b>4</b>
---	----------

Preguntas Específicas de Investigación.....	5
Objetivos.....	5
General.....	5
Específicos .....	5
Justificación .....	6

<b>Revisión de la Literatura .....</b>	<b>7</b>
--	----------

Bases teóricas.....	7
Educación.....	7
Educación en Paraguay.....	7
Sistema Educativo Nacional .....	8
Estructura de la educación formal .....	8
Educación inicial .....	8
Educación Escolar Básica .....	8
Educación Media .....	9
Colegios Privados.....	9
Gestión administrativa.....	9
Software.....	10
Tipos de Software.....	10
Lenguajes de programación .....	12
Gestores de Base de Datos.....	12
Gestión de Configuración del Software.....	13
Sistemas de Gestión.....	13
Software Open Source .....	14
Software libre .....	14
Antecedentes .....	14
Rasgos históricos de la Educación Paraguaya .....	14
Colegios Privados de Encarnación. ....	15
Gestión Administrativa.....	16
Software para Colegios Privados. ....	16
Software de Gestión Administrativa para Colegios Privados de Encarnación. ....	18
Estado del arte de Lenguajes de programación.....	18
Frameworks de Programación .....	20
Gestores de Base de Datos.....	22

<b>Metodología o Materiales y Métodos.....</b>	<b>23</b>
--	-----------

Definición del tipo y diseño de investigación.....	23
Descripción de técnicas e instrumentos de recolección, medición, procesamiento y análisis de los datos .....	23
Procedimientos de aplicación de instrumentos .....	23
Metodología de Desarrollo de Software.....	24

Delimitación.....	24
Alcance .....	24
Limitaciones.....	25
<b>Resultados y Discusión .....</b>	<b>26</b>
Tareas administrativas cotidianas .....	26
Tiempo empleado para las tareas .....	27
Definición del Entorno Tecnológico del Software .....	31
Planificación del Proyecto .....	33
Desarrollo de la Aplicación .....	34
Pruebas y correcciones .....	46
<b>Conclusión .....</b>	<b>48</b>
<b>Lista de Referencias .....</b>	<b>50</b>
<b>Anexos .....</b>	<b>54</b>
<b>Anexo 1: Enlace al proyecto.....</b>	<b>54</b>
<b>Anexo 2: Modelado del Sistema.....</b>	<b>54</b>
<b>Anexo 3: Entrevista realizada.....</b>	<b>55</b>
<b>Anexo 4: Herramienta colaborativa .....</b>	<b>57</b>
<b>Anexo 5: Estructura del Proyecto.....</b>	<b>57</b>
<b>Anexo 6: Repositorio local SmartGit.....</b>	<b>58</b>
<b>Anexo 7: Código del Proyecto – Controlador alumno .....</b>	<b>58</b>
<b>Anexo 8: Código del Proyecto – Buscador por padres .....</b>	<b>59</b>
<b>Anexo 9: Código del Proyecto – Creación de cursos.....</b>	<b>60</b>
<b>Anexo 10: Código del Proyecto – Matriculación.....</b>	<b>61</b>
<b>Anexo 11: Código del Proyecto – Matriculación 2.....</b>	<b>62</b>
<b>Anexo 12: Código del Proyecto – Movimientos.....</b>	<b>62</b>
<b>Anexo 13: Código del Proyecto – Movimientos 2.....</b>	<b>64</b>
<b>Anexo 14: Código del Proyecto – Movimientos 3.....</b>	<b>66</b>
<b>Anexo 15: Código del Proyecto – Movimiento caja .....</b>	<b>67</b>
<b>Anexo 16: Código del Proyecto – Registro de pagos de servicios. ....</b>	<b>68</b>

<b>Anexo 17: Código del Proyecto – Registro producto .....</b>	<b>68</b>
<b>Anexo 18: Código del Proyecto – Cierre de caja.....</b>	<b>69</b>
<b>Anexo 19: Código del Proyecto – Factura .....</b>	<b>69</b>
<b>Anexo 20: Código del Proyecto – Impresión de factura en pdf .....</b>	<b>70</b>
<b>Anexo 21: Código del Proyecto – Routes.....</b>	<b>72</b>

## LISTA DE TABLAS

Tabla 1 - Comparativa .....	28
-----------------------------	----

## LISTA DE FIGURAS

Figura 1 – Logo de RoR.....	32
Figura 2 – Logo de herramientas utilizadas .....	32
Figura 3 – Imagen de generador de pdf.....	33
Figura 4 – <i>Scream de Trello</i> .....	34
Figura 5– Aplicación Login .....	35
Figura 6 - <i>Aplicación Apertura de Caja</i> .....	36
Figura 7 - Aplicación Informe de Caja .....	36
Figura 8 – <i>Aplicación Matriculación</i> .....	37
Figura 9 – Aplicación Listado de alumnos .....	37
Figura 10– <i>Aplicación Buscador por padres</i> .....	38
Figura 11 – Aplicación Creación de curso.....	39
Figura 12 – <i>Aplicación Curso Creado</i> .....	39
Figura 13 – Aplicación Cuenta Corriente del alumno .....	40
Figura 14 – <i>Aplicación Marcar producto</i> .....	41
Figura 15 – Aplicación Pagos parciales.....	41
Figura 16 – <i>Aplicación Alumnos matriculados por curso</i> .....	42
Figura 17 – Aplicación Stock de producto.....	43
Figura 18 – <i>Aplicación Pago de servicios</i> .....	43
Figura 19 – Aplicación Informe de caja .....	44
Figura 20 – <i>Aplicación Vista preliminar de Factura</i> .....	45
Figura 21 – Aplicación Vista preliminar de Factura en formato PDF.....	45

## **LISTA DE ABREVIATURAS**

RoR	Ruby on Rails
MVC	Modelo, Vista Controlador
POJO	Plain Old Java Object
DSL	Domain Specific Language

## **SISTEMA WEB DE GESTIÓN ADMINISTRATIVA PARA EL COLEGIO PARROQUIAL PRIVADO SAN PIO X**

Florentin Alcantara, Melanie Evelin

Ing. Hugo Sendoa

Eje temático: Informática, Educación y Sociedad

### **RESUMEN**

Este trabajo presenta el desarrollo de un sistema web de código abierto diseñado para la gestión administrativa de instituciones de educación privada y su implantación en el Colegio Parroquial Privado San Pio X de la ciudad de Encarnación, Paraguay. Dicho abordaje se debe a que toda empresa o institución privada necesita mecanismos de gestión administrativa fiables que no requieran de la contratación de una empresa por el costo que implica. El objetivo principal fue el desarrollo del producto, cuyas funcionalidades son el resultado de un análisis de requerimiento realizado en esta investigación, contempla desde el módulos de registro y matriculación de alumnos con los respectivos datos y obligaciones, la creación de cursos con sus aranceles correspondientes, un módulo de stock para productos a la venta, un módulo de caja y facturación para el manejo de los movimientos y pagos parciales; y finalmente informes para todos los módulos complementando las funcionalidades necesarias según el análisis. El sistema desarrollado es de código abierto debido a que los sistemas encontrados con similares características cuentan con licencias de pago. La investigación utilizada se considera aplicada y fue desarrollada utilizando la metodología de desarrollo KANBAN, en conjunto con el framework Ruby on Rails, aprovechando el gran número de librerías existentes que aceleran el desarrollo.

**Palabras-claves:** gestión administrativa, sistema web.

## **TEMBIAPO RENDA WEB OJEJAPOHA ÑEMOHENBY JESAREKORÃ MBO`EHAO TUPÃ JEROVIA REHEGUA SAN PIO X**

Apohára: Florentin Alcantara, Melanie Evelin

Sambyhyhára: Ing. Hugo Sendoa

Tembikuaareka rape: mba`ekuaa kuãvoja, Ñehekombó'e, Ava`aty

### **ÑEMOMBYKY**

Ko tembiapo oikuauka jejapopyrepe tembiapo renda WEB, ikatuha ojeike, ojejapóva tembiapo rendarã ñemohemby jesarekorã, pe mbo`ehao tupã jerovia rehegua ojehepyme`ẽhápe San Pio X ,táva Encarnación Paraguái pegua. Ko jehecharã ojejapo ojeikuaa rupi opa tembiapo renda, terã tembiapo renda ojehepyme`ẽhápe oikotevẽ pe jehecharã, ñemohembýpe ojejeroviáva ha ani reiporu ambue jehecharã, hepy rehe pe jejapo upéva. Ko jehupytyrãite, oñemoheñoi ã mba`égui, ojehepyme`ẽhápe pe hembiapo ojeiporúva pe tembiapo atýgui, oñehesa`ỹijo umi ojejerurévagai ko jeporekápe. Ñ tembiapo renda, ikatuha ojeike, ojejuhu rupi ijoguahaite ha oguerékóva kuatia hesegua ha ikatúva ojeiporu oikotevẽ heta jehepyme`ẽve. Ko jeporekarã ojejapóva ikatu ojeiporu oguereko rupi heseve ñemyatyrõve pe mbo`ehao Tupã jerovia jehepyme`ẽhçape San pio X. Ha oiporu guive, ojekupytyta heta mba`e porã, sa`i ojekyhyjéta ha pya`e ojejapóta tembiapo, oñembohekóta tembiapo`aty ojeiporúva; KAMBAN, tembiapo aty ipya`éva oje`eháicha ha ikatu jaiporu ko`ã tembiapópe. Ojeiporu FRAMEWORK RUBY ON RAILS Ojejapo haãgoa ko tembiapo renda. Ojejuhu rupi heta aty aranduka renda oĩva ombopya`eva ipahaitépe.

*ÑE`Ê TEKOTEVÊVA:* jesarekorã, ñemohemby, tembiapo renda Web, Mbo`Ehao Tupã

Jerovia Rehegua San Pio X



**ADMINISTRATIVE MANAGEMENT WEB SYSTEM FOR PRIVATE PARISH  
SCHOOL SAN PIO X**

Author: Florentin Alcantara Melanie Evelin

Advisor: Ing. Hugo Sendoa

Research Line: Computing, Education and Society

**SUMMARY**

This work presents the development of an open source web system designed for the administrative management of private education institutions and its implementation in the Private Parish School San Pio X in the city of Encarnación, Paraguay. This approach is due to the fact that every private company or institution needs reliable administrative management mechanisms that do not require the hiring of a company for the cost involved. The main objective was the development of the product, whose functionalities are the result of a requirement analysis carried out in this research, includes from the registration and enrollment modules of students with the respective data and obligations, the creation of courses with their corresponding tariffs, a stock module for products for sale, a cash module and invoicing for the handling of movements and partial payments; and finally reports for all the modules complementing the necessary functionalities according to the analysis. The system developed is open source because the systems found with similar features have paid licenses. The research used is considered applied and was developed using KANBAN development methodology, in conjunction with the Ruby on Rails framework, taking advantage of the large number of existing libraries that accelerate development.

*KEY WORDS:* administrative management, web system, Private Parish School San Pio X

## **Sistema web de gestión administrativa para el Colegio Parroquial Privado**

### **San Pio X**

#### **Planteamiento del problema**

En la actualidad la administración del Colegio Parroquial Privado San Pio X para el registro de cobros de las cuotas, matrículas, derecho a examen, pago de títulos, (y otras obligaciones como uniforme, libros, viáticos, etc.) y demás gastos con que cuenta la Institución se registran en planillas impresas hechas en Excel y al mismo tiempo se guardan en forma digital en el mismo formato.

La Institución viene trabajando en forma precaria con las planillas de Excel y Word que no son muy adecuados y que dificultan el proceso en el momento de preparar y presentar un informe administrativo ya que para realizar dichos informes primero se debe sumar y restar los egresos e ingresos y otros procedimientos que precisan de mucho tiempo para generar resultados certeros.

Esto a su vez puede ocasionar consecuencias negativas como la pérdida de los archivos en formato digital que casi siempre es causado por algún virus o por la falta de conocimiento en la manipulación de los archivos por parte del usuario, lo que hace necesario contar con las planillas impresas para completar en forma manual generando gastos extras y pérdida de tiempo.

Como es de conocimiento en una empresa y en este caso una institución educativa donde hay movimientos de ingresos, egresos y sus constantes variaciones, los informes de estados de cuentas lleva su tiempo preparar con exactitud y cuando se elabora en forma manual se retrasa en el momento de presentar a la comunidad educativa, en especial a las familias que hoy día viven en un tiempo muy acelerado y con muchos compromisos de rutinas por lo que urge contar

con un sistema dinámico, práctico y sobre todo facilitador cuando se requiera contar con los informes, que éste sea en el menor tiempo posible.

Existe una necesidad imperiosa de brindar seguridad y transparencia en los informes que se presentan garantizando que los datos que arroja son certeros y confiables ante cualquier instancia que se requiera informar. Y de esta manera la institución salvaguarde su confiabilidad y que pueda ofrecer garantías a todas las familias que acuden a solicitar su servicios educativos y porque no a los docentes para que trabajen con total tranquilidad que sus derechos como trabajador están resguardados y bien documentados.

Ante esta problemática y teniendo los recursos informáticos necesarios, surge la siguiente interrogante: ¿Es posible desarrollar un Sistema web de Gestión Administrativa para el Colegio Parroquial Privado San Pio X?

### **Preguntas Específicas de Investigación**

1. ¿Cómo funciona la gestión administrativa actualmente en el Colegio Parroquial Privado San Pio X?
2. ¿Qué datos hay sobre Sistemas de Gestión Administrativa ya existentes?
3. ¿Cuáles son los datos más importantes que se necesitan para la elaboración de los requisitos del sistema?
4. ¿Es posible desarrollar un sistema web de gestión administrativa?

### **Objetivos**

**General.** Desarrollar un Sistema Web de Gestión Administrativa para el Colegio Parroquial Privado San Pio X

#### **Específicos.**

- Analizar los requerimientos necesarios a ser implementados por medio de la aplicación de los diferentes instrumentos para la obtención de información.
- Indagar sobre la existencia de posibles herramientas de software que cumplan con los requisitos obtenidos.

- Determinar los componentes, módulos e interfaces necesarias a implementar.
- Desarrollar el sistema web de Gestión Administrativa

### **Justificación**

La tecnología hace que cada día que pasa dependamos de ella, ofreciendo infinidades de herramientas que ayudan con las gestiones administrativas de tal manera a agilizar el trabajo del encargado, obligando así a que las empresas o instituciones informaticen sus gestiones. Existen sistemas para cada tipo de gestión con diferentes características dependiendo del ámbito en el cual se lo quiera usar.

Ante la problemática mencionada anteriormente con el registro de los datos administrativos de la Institución y los problemas que surgen al perderse de alguna u otra forma esta información, se ha planteado desarrollar un Sistema Web de Gestión Administrativa para el Colegio Parroquial Privado San Pio X que registre todas sus operaciones, brindando así una centralización de los datos de acceso inmediato, fundamental para la toma de decisiones.

El desarrollo de este sistema adquiere importancia debido a que pretende convertirse en una herramienta, que mediante el procesamiento de datos de los registros administrativo, ayude a el acceso a la información de forma inmediata, disminuyendo el uso de planillas impresas basado en un software de código abierto (open source) que no requiere de la adquisición de licencias y el cual puede ser ampliado, distribuido o replicado en otras instituciones de manejo similar.

El software de código abierto es aquel que incluye el código fuente y está disponible para modificarlo, distribuirlo y mejorarlo. Para ser considerado open Source debe de contar con ciertos requisitos (Spano, 2010)

## Revisión de la Literatura

### Bases teóricas

#### Educación

Según la Ley N° 1.264 General de Educación en el artículo 11 *“se entiende por educación el proceso permanente de comunicación creativa de la cultura de la comunidad, integrada en la cultura nacional y universal, para la realización del hombre en la totalidad de sus dimensiones”* (Ministerio de Educación y Cultura, 1998).

#### Educación en Paraguay

Acorde al Consejo Educativo Departamental, *“Las condiciones socioeconómicas y demográficas en las cuales se desarrolla el sistema educativo constituyen información relevante que deben ser consideradas para el diseño de políticas educativas y de intervenciones específicas”* (Consejo Departamental de Educación, 2016).

Durante 17 años la sociedad paraguaya fue testigo del proceso de cambio de la Reforma Educativa o al menos lo escucho por varios años sin ver los resultados que esperaban o que prometían con la dicha reforma. Cansados de escuchar lo mismo la sociedad exige que se revea en el ámbito educativo y que las instituciones educativas tanto como colegios, escuelas y universidades desean un espacio con calidad donde se formen las nuevas generaciones.

El MEC asume la tarea de socializar sus propuestas para el sector educativo a través del diálogo y la participación de la sociedad.

En base a la última estadística realizada el Paraguay ha registrado que en las zonas rurales hay un elevado índice de niños y jóvenes estudiando. Lo contrario pasa en las zonas urbanas en donde el mayor porcentaje de niños y jóvenes se encuentran en las calles trabajando (Consejo Departamental de Educación, 2016).

### **Sistema Educativo Nacional**

En el artículo nro. 26 de la Ley 1.264 General de Educación dice que “*El Sistema Educativo Nacional se incluye en régimen general, la educación de régimen especial y otras modalidades de atención educativa. El régimen general puede ser formal, no formal y refleja*” ( Ley Nro 1.264 General de Educación, 1998).

### **Estructura de la educación formal**

En el artículo 27 y en adelante hasta el artículo nro. 55 de la Ley nro. 1.264 General de Educación menciona la estructura formal la cual está formada por los siguientes niveles:

#### ***Educación inicial***

Comprende dos ciclos, el primer ciclo será hasta los tres años y el segundo hasta los cuatro años, las clases serán impartidas por profesionales en el caso de que no hubiese posibilidad de contar con dicho profesional lo podrá ejercer otra profesional no especializado solo con la autorización del Vice Ministro de la Educación.

La lengua que se enseñará será la oficial materna y así también la segunda lengua pero con tratamientos didácticos ( Ley Nro 1.264 General de Educación, 1998).

#### ***Educación Escolar Básica***

Esta comprende nueve grados que deberán cursarse de manera obligatoria y gratuita en las instituciones públicas. Las áreas y sus contenidos serán determinados por el Ministerio de Educación y Ciencia. Una vez cursado los nueve grados los alumnos recibirán por parte de la institución una constancia con sus calificaciones obtenidas durante el transcurso de los años y una orientación para el futuro académico y profesional del alumno ( Ley Nro 1.264 General de Educación, 1998).

### ***Educación Media***

Comprende el bachillerato o la formación profesional y tendrá tres cursos académicos. El objetivo principal es la incorporación del alumno en la vida social, al trabajo productivo o a la educación de nivel superior. A los que acceden a la formación profesional o bachillerato podrán acceder a pasantías laborales sin vinculación laboral. Una vez terminado satisfactoriamente los tres cursos se le otorgará el título de bachiller para acceder a la formación profesional superior (Ley Nro 1.264 General de Educación, 1998).

### **Colegios Privados**

La Ley Nro. 1.264 General de Educación en el Capítulo VI Educación Pública y Privada en sus artículos 61 y 62 menciona que la educación podrá ser gestionada de forma privada por personas, empresas, asociaciones o instituciones con recursos del estado. Las instituciones que pretendan otorgar títulos deberán estar reconocidas por las autoridades educativas competentes de la Republica. Podrán prestar de este servicio las iglesias o confesiones religiosas, inscriptas en el registro Nacional de Culto, las fundaciones, sociedades, asociaciones y empresas con personería jurídica, y las personas de existencia visible (Ministerio de Educacion y Cultura, 1998).

### **Gestión administrativa**

Es la forma en la que se utilizan los recursos escasos para conseguir los objetivos deseados. Se realiza a través de 4 funciones específicas planeación, organización, dirección y control.

En los últimos años se han creados nuevos sistemas de gestión que automatizan los procesos con el fin de mejorar la calidad y la eficacia en las tareas que realiza, estos sistemas son

los informáticos que permite la integración de los distintos procesos de manera a agilizar la circulación de la información y los documentos. (Principios de la Gestion Administrativa, 2009)

## **Software**

Según Pressman, Roger S. en su libro de Ingeniería del Software- Un enfoque práctico define software como *“Un producto y al mismo tiempo es el vehículo para entregar un producto. En su forma de producto, brinda el potencial de cómputo incorporado en el hardware de cómputo o, con más amplitud, en una red de computadoras a las que se accede por medio de un hardware local”*

Se lo puede definir como el producto de los desarrolladores que después le dan mantenimiento a largo plazo con actualizaciones (Pressman, El Software y la Ingenieria del Software, 2010).

### ***Tipos de Software***

#### **Software de sistema.**

Está formado por todos aquellos programas cuya finalidad es servir al desarrollo o al funcionamiento de otros programas, se caracterizan por estar próximos al hardware, procesan por sobre todo datos indeterminados así también estructura de información compleja (Aguilera, 2015).

Se caracteriza por la integración con el hardware de la computadora, por el uso intenso de usuarios múltiples, recursos compartidos, la administración de un proceso sofisticado.

#### **Software de tiempo real.**

Está formado por todos aquellos programas que miden, analizan y controlan los sucesos del mundo real a medida que ocurren, debiendo de reaccionar de forma correcta a los estímulos de entrada en un tiempo máximo prefijado.



Hace énfasis en el tiempo que realiza o en que se produce las acciones y lleva más en cuenta la rapidez o que las acciones se produzcan en el momento adecuado. No solo deberán de funcionar correctamente la parte lógica.

Dependiendo del nivel de exigencia temporal se pueden clasificar en: Críticos, esenciales, incrementables y no esenciales. Así también dependiendo de la arquitectura: propietarios y abiertos. Dependiendo de la arquitectura del sistema centralizados y distribuidos (Aguilera, 2015).

### **Software de Gestión.**

Estos programas utilizan grandes cantidades de información almacenadas en base de datos con objeto de facilitar las transacciones comerciales o la toma de decisiones. Además de las tareas convencionales de procesamiento de datos, en las que el tiempo de procesamiento no es crítico y los errores pueden ser corregidos.

Son herramientas que sirven de manera a gestionar las tareas y agilizarlas. La mayoría lo utiliza como un medio para compartir informaciones. Como ser a través de los Blogs, fueron creadas para coordinación de información de modo a poder crearlas, modificarlas, organizarlas y potenciar sus esfuerzos (Aguilera, 2015).

### **Software científico y de ingeniería.**

Se encarga de realizar cálculos complejos sobre datos numéricos de todo tipo. En este caso la corrección y exactitud de las operaciones que realizan es uno de los requisitos básicos que deben de cumplir.

Sin embargo estas están abandonando los algoritmos convencionales y el diseño asistido por computadora, la simulación de sistemas y otras aplicaciones interactivas han comenzado a hacerse en forma real (Aguilera, 2015).

### **Software de inteligencia artificial**

El software basado en lenguajes procedimentales es útil para realizar de forma rápida y fiable operaciones que para el ser humano son tediosas e incluso inabordables. Sin embargo, es difícilmente aplicable a problemas que requieran la aplicación de funciones intelectuales más elevadas, por triviales que nos puedan parecer.

Las aplicaciones en esta área incluye la robótica, sistemas expertos, reconocimiento de patrones, redes neurales artificiales, demostración de teoremas y juegos (Aguilera, 2015).

### **Lenguajes de programación**

#### **Frameworks de Desarrollo**

Se refiere a una estructura de software integrado por componentes personalizables e intercambiables para el desarrollo de una aplicación. Se puede considerar como una aplicación genérica incompleta y configurable a la que se le puede añadir elementos para desarrollar una aplicación concreta (Medición de atributos POO en frameworks de desarrollo PHP., 2012).

#### **Gestores de Base de Datos**

El Sistema de Gestión de base de datos es una aplicación que permite a los usuarios definir, crear y mantener la base de datos, además de proporcionar un acceso controlado a la misma (Marqués, 2011).

Tiene como característica la definición de la base de datos mediante el lenguaje de definición de datos, permite la inserción, actualización, eliminación y consulta de datos, el acceso controlado a la base de datos mediante un sistema de seguridad, de integridad, un sistema de control de concurrencia, un sistema de control de recuperación y un diccionario de datos o catalogo que con la descripción de los datos de la base de datos. (Marqués, 2011)

## **Gestión de Configuración del Software**

Es un “*conjunto de actividades diseñadas para administrar el cambio mediante la identificación de los productos de trabajo que es probable que cambien, el establecimiento de las relaciones entre ellos, la definición de mecanismos para administrar diferentes versiones de dichos productos de trabajo y el control de los cambios impuestos, así como la auditoria y los reportes de cambios realizados*” (Pressman, Administracion de la Configuración del Software, 2010).

La gestión de configuración del software ayuda a tener de manera ordenada el control de las versiones del sistema con ayuda de técnicas y herramientas que ayudan a definir una estrategia para la gestión de cambio para que se pueda entregar el producto final deseado y a tiempo.

La evolución del software hace que se necesite de gestionar la configuración, durante el desarrollo, durante la explotación y en todos los casos ya que algunas veces se hace necesario contar con versiones antiguas.

## **Sistemas de Gestión**

Un sistema de gestión es una herramienta o aplicación informática que permite controlar todos y cada uno de los aspectos de una empresa (pedidos, producción, control de presencia, facturación, ventas, administración, etc.) (GRUPO I.A.G, 2004).

De esta forma la empresa podrá tener una visión de los movimientos diarios con que cuenta y fijarse en los puntos negativos o positivos con lo que cuenta durante el proceso.

## **Software Open Source**

El software Open Source es aquel que incluye el código fuente y está disponible, más bien es una metodología ya que se suele confundir con el software libre. Para ser considerado open Source debe de contar con algunos requisitos:

El software debe ser libre de distribuir, se debe permitir modificaciones derivadas del mismo, la licencia no debe discriminar a ninguna persona y la licencia no debe limitar ningún campo de aplicación o emprendimiento. El software que cuenta con una licencia Open Source permite que una comunidad de desarrolladores lo mejore, lo corrija, lo pruebe y lo mantenga (Spano, 2010).

## **Software libre**

La Fundación de Software Libre lo define como *“el software que se puede utilizar, copiar, estudiar, modificar y redistribuir sin ninguna restricción”* Para ser libre debe de considerarse cuatro reglas la primera es la de ejecutar un programa con cualquier propósito, la segunda es la de poder modificarla como uno quiera, la tercera y cuarta la libertad de redistribuirla con las modificaciones hechas en las últimas versiones. Para ello se necesita de utilizar el código fuente (Free Software Foundation, 1985).

## **Antecedentes**

### **Rasgos históricos de la Educación Paraguaya**

Hasta los fines de la década del sesenta el sistema educativo paraguayo seguía manteniendo gran parte de sus características históricas. Es decir, un fuerte predominio de las escuelas localizadas en zonas rurales o semi-urbanas, altas tasas de repitencia y deserción, una pronunciada diferencia entre los niveles de calidad entre las unidades educativas rurales y las urbanas.

Una característica no menos relevante del sistema educativo tradicional y que se agudizó en los últimos años es su fuerte tendencia a la división. Este fenómeno estructural explica en gran medida la forma que históricamente ha caracterizado a la pirámide educacional paraguaya: una extrema agudización en los tramos superiores y un fuerte engrosamiento en la base. Dicha estructura refleja una compleja conjunción de determinantes que condicionan la capacidad diferenciada de acceso a la educación de los diferentes sectores sociales. De igual manera, representa los rangos que corresponden a las unidades educativas en cuanto a su nivel de calidad.

Es poco lo que el sistema educativo tradicional ha podido corregir en cuanto a ciertas peculiaridades en el orden más estrictamente pedagógico. El carácter frontal de la enseñanza, la tendencia enciclopedista y el sesgo a la memorización quedaron fuertemente arraigados en la escuela tradicional, a pesar de haberse dado a lo largo del tiempo importantes esfuerzos para su control. Los intentos de reforma educativa de 1954 y la de 1974 prestaron especial atención a tales problemas aunque sin alcanzar progresos sustantivos (Rivarola, 2000).

### **Colegios Privados de Encarnación.**

En la ciudad de Encarnación cuenta con varias instituciones privadas algunas de ella son:

Colegio Privado Subvencionado Antonio Provolo

Colegio Inmaculada Concepción

Colegio Santa María

El Principito

Colegio Privado Girasoles

Colegio Divina Esperanza

Colegio Integrity

Colegio Privado Luz y Vida

### Colegio Parroquial Privado San Pio X

Este último nombrado es el que se toma como punto de partida. Dicho colegio viene funcionando hace más de 50 años en la labor educativa, formando jóvenes para el futuro. Es dependiente de la Iglesia Catedral y trabaja de forma privada.

### **Gestión Administrativa**

La administración y conducción de una institución educativa no pasa muy lejos de lo que es la conducción de una empresa, es por eso que debe tener en cuenta las cuatro funciones básicas que debe considerar: la planeación, la organización, la comunicación y el control. La gestión administrativa debe de ser estructurada y presentada a las instancias correspondiente para no generar confusiones (Cavassa, 2004).

### **Software para Colegios Privados.**

Existen varios proyectos que tienen características similares a las del sistema desarrollado, la diferencia está en que este sistema es de código abierto.

#### **gEscolar-CM:**

Este sistema es de origen venezolano para el Control de Mensualidades de Colegios, Escuelas, Pre-Escolares, Maternales, Academias y cualquier institución en donde la forma de pago este basada en cuotas, ya sean mensuales, quincenales o semanales (colegiosyliceos.com, 2000).

A pesar de que cuenta con los distintos módulos de cobros de cuotas, matriculación, emisión de facturas, historial de pagos la diferencia está en la cotización que es en Bolívar (moneda venezolana), y tiene un costo.

Algunas características:

- Sistema para el Control de Mensualidades de Colegios, Escuelas, Pre Escolares, Maternales, Academias y cualquier institución en donde la forma de pago está basada en cuotas, ya sean mensuales, quincenales o semanales.
- Manejo de múltiples cursos por estudiante.
- Emisión de Facturas.
- Manejo de Abonos o Pagos parciales.
- Reporte de Estudiantes morosos detallado por curso o tarifa o resumido por curso o tarifa.
- Historial de pagos del Representante.
- Exportación de Datos a Ms-Excel, Pdf, Html

**aulica:**

Este sistema cuenta con los módulos de Administración de inscripciones, cuotas, deudas y becas. Registro de movimientos de caja. Recaudación a través de Pago Fácil, Rapípagos y Bancos. Este sistema también tiene un costo, y la cotización está basada en pesos argentinos lo que lo hace no tan factible en nuestro país (aulica).

**Algunas características:**

- Contabilidad: Asientos contables manuales y automáticos. Estructura contable adaptable a la institución. Emisión de libro diario, mayor y balance.
- Presupuesto: Registra los ingresos y egresos que estima obtener tras finalizar el ejercicio contable.
- Comunicación: Comunicación fácil y ágil con todos los miembros de la comunidad educativa: preceptores, profesores, padres y alumnos.

**Sistema Saberes:**

Es un software de gestión académica y administrativa para instituciones educativas que permite la integración de los datos académicos, administrativos, financieros, psicológicos y médicos de la comunidad educativa es de origen colombiana (SistemaSaberes, 2009).

**Características:**

- No requiere que se instale ningún servidor, programa o equipo específico dentro de la institución; Use los equipos que ya tiene.
- Cada miembro de la comunidad educativa accede a su información según su rol en la institución, con su usuario y su contraseña.
- Los datos estarán protegidos con los mejores mecanismos de copias de respaldo, protegidos y disponibles.

**Software de Gestión Administrativa para Colegios Privados de Encarnación.**

Según la información recolectada en la zona de Encarnación a ciertas empresas/negocios que se dedican al desarrollo y venta de sistemas, no se ha desarrollado uno con las mismas características.

**Estado del arte de Lenguajes de programación**

Según la estadística realizada en el año 2016 por la empresa Holandesa TIOBE siguen siendo los más utilizados:

**Java:** el más utilizado por su legibilidad y simplicidad, además por su estabilidad que asegura el funcionamiento de las aplicaciones que la utilizan (TIOBE, 2015).

El lenguaje que permanece a través del tiempo y es el más utilizado; compilado e interpretado en tiempo real. La rapidez al momento de ejecutar los programas es otra de las



características, así también se destaca por hecho de que cualquier programa creado en JAVA se puede ejecutar desde cualquier ordenador con distintos tipos de sistemas operativos.

**C:** es el segundo más utilizado generalmente para sistemas de escritorio. Es de propósito general que lo hace muy flexible (TIOBE, 2015).

Este lenguaje de programación se usa mayormente para desarrollar sistemas operativos, no es orientado a objeto pero en C++ se realizaron modificaciones en la cual permiten las clases, métodos y atributos, encapsulación y polimorfismo en si orientada a objeto. Tal vez no sea un lenguaje para escribir directamente aplicaciones de gestión grafica en forma nativa

**C++:** es orientado a objetos surge como una continuación y ampliación de C.

C++ es una extensión de C que fue diseñado en base a la POO, no hay mucha diferencia con C es por eso que los programadores de C no tienen mucha dificultad en aprender el mismo. Este lenguaje también es uno de los más utilizado por su potencia. La mayor parte de los programadores lo utiliza para desarrollar video juegos.

**C#:** este lenguaje de programación se destaca por su sencillez y modernidad.

El lenguaje de programación C# trabaja en con el framework .NET que es una plataforma de Windows de código abierto. Microsoft trabaja con esta tecnología y el código fuente se pueden ver en el repositorio de GitHub. Muchos desarrolladores optan por este lenguaje de programación. Según el índice de TIOBE se encuentra en el cuarto lugar desde el año 2015

**PHP:** se utiliza para desarrollar plataformas web junto con la base de datos MySQL.

Una de las características que tiene este lenguaje de programación es que cuenta con una gran documentación. Es utilizado por varias empresas ya que posee la habilidad de manejar una gran cantidad de datos, además de ser multiplataforma y libre.

Cabe destacar que muchos frameworks están basados en PHP (TIOBE, 2015).

**Ruby:** es un lenguaje de programación dinámico y de código abierto que está enfocado en la simplicidad y productividad (TIOBE, 2015).

Es un lenguaje de programación interpretado, de alto nivel y orientado a objetos. Es denominado un lenguaje multiparadigmas. Fue creado para mejorar la productividad se hizo conocido gracias al framework RoR hoy en día no es de preferencia para muchos desarrolladores por la desventaja que tiene con la velocidad.

### **Frameworks de Programación**

**Struts 2:** se utiliza para crear aplicaciones web orientadas a empresas optimizando el proceso de desarrollo. Se utiliza cuando la carga de datos es grande (OpenWebinars, 2016).

Se caracteriza por su diseño simplificado, proporciona un Framework de validación que nos permitirá desacoplar las reglas de validación del código de las acciones. Permite también usar cualquier clase Java normal (POJO) como una acción, en struts 2 desaparecen los ActionForm se usan los JavaBean que leen directamente las propiedades, arranque rápido, esto quiere decir que no hace falta volver a reiniciar el servidor.

Struts 2 es un framework basado en el patrón MVC no es el más utilizado, sin embargo hay empresas que cuentan con sistemas programados con dicho lenguaje de programación que no hacen la migración por el hecho que les conllevaría un inmenso trabajo.

**CodeIgniter:** es uno de los framework utilizado por los desarrolladores de PHP por su velocidad y potencia, además que resulta fácil de aprender por su sencillez.

Así como los demás framework ya mencionados este trabaja con el patrón MVC. Se destaca por poder trabajar con la mayoría de los entornos o servidores. El núcleo del sistema solo necesita algunas librerías a diferencias de los otros que requiere de más recursos. Posee una

característica de auto-carga que proporciona la inicialización de librerías, asistente y complementos de forma automática durante el proceso de arranque del sistema.

**Symfony2:** es un framework donde su código, componentes y librerías están publicados bajo la licencia MIT de software libre. Es uno de los framework recomendado y requerido por las empresas. Se utiliza generalmente para desarrollar proyectos grandes.

Symfony se construyó en base a otros frameworks, tomo lo mejores conceptos. En cuanto a la arquitectura cuenta con un micro-kernel altamente optimizado. Cada característica de Symfony está desarrollada con un bundle, que es un conjunto estructurado de archivos.

**Laravel:** la característica de este framework es que es fácil de aprender y usar en el momento de desarrollar una aplicación web. Cuenta con su propio motor de plantillas en la cual se puede escribir directamente el código sobre ellas.

Cuenta con dos repositorios en uno de ellos está el núcleo y el segundo seria el proyecto base, para que este funcione de debe descargar todas sus dependencias a través de composer. En laravel con un simple comando se podrá actualizar automáticamente todas las dependencias hasta los paquetes de terceros.

**Sinatra:** se define como un Domain Specific Language o DSL que deja al desarrollador elegir las herramientas adicionales dependiendo al tipo de desarrollo que este esté haciendo.

Sinatra no sigue el típico patrón MVC como lo hacen los otros framework. Se enfoca en la rápida creación de aplicaciones web en Ruby. Como es un framework minimalista, no fue pensado para resolver grandes problemas. Esto hace que no sea rápido y menos popular.

**Ruby on Rails:** es de código abierto. Su objetivo es favorecer la convención antes de la configuración, disminuir la repetición de código (aulaFormativa).

Como es de conocimiento RoR es un framework de programación desarrollado en el lenguaje Ruby, de código abierto que utiliza el paradigma MVC el cual es utilizado para el desarrollo de aplicaciones web. Es utilizado por varios desarrolladores por la rapidez que lo caracteriza y por el hecho de que simplifica las tareas repetitivas. Otro de los puntos positivos con que cuenta es que rails posee una serie de plugins que están a disposición del público. Además se pueden hacer modificaciones y aplicarlas fácilmente.

### **Gestores de Base de Datos**

**MySQL:** Es uno de lo más utilizado debido a que es open source y de fácil instalación. Es un gestor relacional de base de datos, que organiza información en distintos archivos dependiendo el motor que se utilice y en los cuales podemos guardar un simple registro hasta un complejo sistema relacional orientado a objetos (Mussa, 2008).

Básicamente ejecuta todas las plataformas incluyendo Linux. Una de sus características es que se asocia más a las aplicaciones basadas en la web.

**Microsoft SQL server:** es un sistema gestor de base de datos relacionales producido por Microsoft. Es un sistema cliente/servidor que funciona como una extensión del sistema operativo Windows. Entre otras características proporciona integridad de datos, optimización de consultas, control de concurrencia y backup y recuperación (SQL Server , 2016).

Cuenta con varias características que lo hacen especial y que se quiera usar. Una de ellas es que trae soporte aproximado como para diez motores de almacenamientos con sus características y por sobre todo bajo costo.

**PostgreSQL:** es un Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos. Es un gestor de bases de datos de código abierto, utiliza el modelo cliente/ servidor y multiproceso, permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación (Martinez, 2010).

## **Metodología o Materiales y Métodos**

### **Definición del tipo y diseño de investigación**

El tipo de investigación seleccionado es la aplicada. Se caracteriza porque busca la aplicación de conocimientos adquiridos, a la vez que se adquieren otros, después de implementar y sistematizar la práctica basada en investigación (Sampieri, 2010).

En cuanto al diseño de investigación fue la investigación cualitativa ya que la misma consiste en el relevamiento de datos antes, durante y después de esa manera poder determinar y elaborar los requerimientos para su posterior análisis.

### **Descripción de técnicas e instrumentos de recolección, medición, procesamiento y análisis de los datos**

Para la investigación aplicada se implementó la entrevista como primer instrumento, donde se elaboró la entrevista de forma personal con la persona a cargo de la administración que se encuentra en el anexo 3 acompañado de la observación y la investigación documental del Colegio Parroquial Privado San Pio X.

### **Procedimientos de aplicación de instrumentos**

Se elaboró una entrevista personal con el administrador de la institución con preguntas cerradas que hacían referencia principalmente a la parte administrativa de modo a obtener informaciones certeras, y así poder realizar los requerimientos del sistema.

Se realizó una investigación en la web para determinar si existen sistemas que coinciden con los requerimientos obtenidos, por otro lado también se realizó un sondeo en la zona de Encarnación donde se les pregunto a cuatro empresas si contaban con algún sistema que cuente con los requerimientos y el presupuesto del mismo para la elaboración de los requerimientos del sistema.

## **Metodología de Desarrollo de Software**

La metodología a utilizada fue KANBAN ya que se basa en disminuir retrasos y crear un sistema de producción eficiente.

Los sistemas KANBAN consisten en un conjunto de formas de comunicarse e intercambiar información entre los diferentes operarios de una línea de producción, de una empresa, o entre proveedor y cliente. Su propósito es simplificar la comunicación, agilizándola y evitando errores producidos por falta de información (PDCA Home, 2012).

Las principales reglas de KANBAN son tres: Visualizar el trabajo y las fases del ciclo de producción o flujo de trabajo, determinar el límite de “trabajo en curso” (o Work In Progress) y medir el tiempo en completar una tarea (lo que se conoce como “lead time”) (Garzas, 2011).

Otra de las metodologías que se utilizó para el desarrollo del sistema fue la metodología de código abierto. El software código abierto es aquel que incluye el código fuente y está disponible y permite que una comunidad de desarrolladores lo mejore, lo corrija, lo pruebe y lo mantenga. Siempre cumpliendo los requisitos que se deben tener en cuenta para que se lo pueda considerar como software de código abierto.

## **Delimitación**

### **Alcance**

El sistema de gestión administrativa contará con los siguientes módulos:

Módulo de Inscripción: en este módulo se guardaran los datos personales de los alumnos de los padres, tutores o encargados y el curso al cual ingresara.

Módulo de creación de curso: en el cual tendrá que registrar los cursos y el tipo de bachillerato correspondiente a cada curso con sus respectivos aranceles.

Interfaz de Cuenta Corriente (por alumno): al matricular al alumno se crean automáticamente la cuenta corriente con el monto total de sus obligaciones.

Módulo de Stock: como la institución cuenta con artículos de uniformes, insignias y librería tendrá un stock donde podrá registrar dichos artículos que se encuentra en venta en la institución.

Módulo Caja: así como todo sistema, tendrá la apertura y cierre de caja.

Módulo de Facturación: contará con una vista pre-impresa de la factura.

Informes: en este podrá obtener los informes que necesite en cuanto a los informes del estado de cuenta de los alumnos y al de la institución.

### **Limitaciones**

El sistema no contempla el registro de pagos o adelantos de salario de profesores.

Registro de horas cátedras.

Registro de las tareas académicas.

El módulo de facturación no cuenta con timbrado.

## Resultados y Discusión

Como el objetivo número uno se propone Analizar los requerimientos necesarios a ser implementados por medio de la aplicación de los diferentes instrumentos para la obtención de la información.

Todo análisis de sistemas informático requiere conocer el problema que intervendrá. Según el libro de Sampieri *“Una vez que seleccionamos el diseño de investigación apropiado y la muestra adecuada, de acuerdo con nuestro problema de estudio e hipótesis la siguiente etapa consiste en recolectar los datos pertinentes sobre los atributos, conceptos o variables de las unidades de análisis o casos”* (Sampieri R. H., 2010). Una vez conocido los distintos instrumentos para la recolección se ha seleccionado la entrevista como primer mecanismo de recolección de datos porque es el que se adecua a la situación. Se elaboró una entrevista al personal administrativo que constó de 15 preguntas abiertas, ya que se buscó obtener información más amplia sobre el tema. Las distintas preguntas estaban enfocadas a obtener la percepción del potencial usuario del sistema con respecto a los siguientes puntos:

### **Tareas administrativas cotidianas**

Este ítem se consideró necesario ya que en el radica la actividad principal a la cual el sistema estaría abocado, si bien se enfatizó sobre puntos como cantidad de cuotas, medios de pagos de cuotas, registros de pagos por alumnos, cierre diario de caja, registro de ventas de uniforme y demás; se buscó determinar cuáles eran las tareas administrativas cotidianas que desempeñaba la secretaria.



**Tiempo empleado para las tareas**

Como segundo ítem se consideró el tiempo empleado para realizar las tareas cotidianas, es un factor muy importante debido a que se pierde tiempo elaborando manualmente dichas tareas. Con el sistema se busca agilizar y disminuir el tiempo que conlleva las mismas.

El método de recolección de datos fue mediante una entrevista, si bien ayuda a tabular los datos, no pueden representar toda la operativa. Es por ello que se optó por acompañar con una observación de sus tareas cotidianas registrando todos los eventos que resulten de importancia al observador. Se aprovechó que el mismo formaba parte del plantel administrativo, para realizar las tareas.

Los puntos importantes que formaron parte de la observación consistían en:

Observar el trabajo cotidiano del personal administrativo así de esta manera se pudo hacer constar la forma en que trabaja y realiza sus actividades.

Medir el tiempo que le tomaba realizar cada tarea administrativa, esto fue posible gracias a que el observador se encontraba presente en el momento en el que realizaba dichas tareas, de esa forma medía el tiempo aproximado.

La metodología que utilizaba para agilizar sus tareas: no utilizaba ninguna metodología.

Se pudo determinar en base a las respuestas obtenidas en la entrevista que la administración del Colegio Parroquial Privado San Pio X viene trabajando de forma deficiente, registrando sus movimientos y formularios de inscripción en planillas impresas y en archivos Excel, dando la posibilidad de que estos se pierdan o que los archivos se dañen por algún virus o mala manipulación del usuario y estas no se vuelvan a recuperar, perdiendo así información valiosa para la institución. Así también se ha podido determinar con la observación que dichas

acciones necesitan de un determinado tiempo, que implica el corroborar cada uno de los datos minuciosamente para evitar errores y poder entregar un informe con datos exactos.

Como objetivo número dos se propone indagar sobre la existencia de posibles herramientas de software que cumplan con los requisitos obtenidos.

Se procedió a la indagación en la web con el fin de recabar datos sobre sistemas existentes que cuentan con funciones similares.

Se han encontrado varios sistemas he escogido tres de ellos los cuales son:

<b>gEscolar</b>	<b>aulica</b>	<b>Sistema Saberes</b>
<ul style="list-style-type: none"> <li>• Sistema para el Control de Mensualidades de Colegios, Escuelas, Pre Escolares, Maternales, Academias y cualquier institución.</li> <li>• Manejo de múltiples cursos por estudiante.</li> <li>• Emisión de Facturas.</li> <li>• Manejo de Abonos o Pagos parciales.</li> <li>• Reporte de Estudiantes morosos detallado.</li> <li>• Exportación de Datos a Ms-Excel, Pdf, Html</li> <li>• Pago moneda extranjera</li> </ul>	<ul style="list-style-type: none"> <li>• Contabilidad: Asientos contables manuales y automáticos.</li> <li>• Presupuesto: Registra los ingresos y egresos que estima obtener tras finalizar el ejercicio contable.</li> <li>• Comunicación: Comunicación fácil y ágil con todos los miembros de la comunidad educativa: preceptores, profesores, padres y alumnos.</li> <li>• Pago moneda extranjera</li> </ul>	<ul style="list-style-type: none"> <li>• No requiere que se instale ningún servidor, programa o equipo específico dentro de la institución.</li> <li>• Cada miembro de la comunidad educativa accede a su información según su rol en la institución, con su usuario y su contraseña.</li> <li>• Los datos estarán protegidos con mecanismos de copias de respaldo, protegidos y disponibles.</li> <li>• Pago moneda extranjera</li> </ul>

**Tabla 1 - Comparativa**

El inconveniente que se encontró en estos es que son extranjeros y no cotizan la moneda nacional, así también tienen costos que van desde los USD 440 hasta USD1.750 teniendo en cuenta que estos precios varían de acuerdo a los módulos que el usuario solicite. También se ha

realizado un sondeo a cuatro empresas de la ciudad de Encarnación, a las cuales se les pregunto si cuentan con sistemas con las características requeridas en el párrafo anterior y solicitándoles un presupuesto de un sistema administrativo para una institución educativa, a la cual respondieron que si bien no cuentan con un sistema lo pueden desarrollar y el costo aproximado seria de 3.000.00 a 8.000.000 de gs., el cual varía dependiendo de los requerimientos.

Debido a la falta de disponibilidad financiera, se optó por desarrollar el sistema de código abierto que contemple las funcionalidades necesarias para la institución.

Como objetivo número tres se proponen determinar los componentes, módulos e interfaces necesarias a implementar.

Luego de haber hecho el relevamiento de datos correspondiente y de acuerdo a los resultados obtenidos de los instrumentos de recolección de datos seleccionados, se procedió a la elaboración de los requerimientos. Obteniendo lo siguiente:

**Login**, donde la persona a cargo pueda tener un nombre de usuario y contraseña para poder acceder al sistema.

**Módulo de inscripción**, este deberá contener los datos que normalmente utiliza en su formulario de inscripción para el registro del alumno en la institución y el curso en el cual está habilitado para registrarse.

**Creación de los cursos** deberá contar con la lista de los cursos y el tipo de bachillerato con el que cuenta la institución y los aranceles correspondientes.

**Cuenta corriente**, cada alumno deberá contar con una cuenta corriente con todas sus obligaciones correspondientes al curso en el cual se matricule.

**Stock de artículos**, como la institución cuenta con un stock de artículos de uniformes, insignias y de librerías deberá contar con un registro de los mismos de tal forma a tener un inventario actualizado.

**Módulo de caja**, como toda institución privada esta deberá contar con la posibilidad de registrar todos sus ingreso y egreso de modo que al finalizar el día esta pueda contar con un informe del estado actual de su caja.

**Módulo de facturación**, la facturación es uno de los requisitos más importantes con el que debe contar de manera a tener de forma impresa un comprobante de los egresos.

**Módulo Registro de pagos a docentes** el cual deberá contar con la lista de los docentes y parte administrativa con sus respectivas horas cátedras y monto del sueldo asignado, de tal forma a poder registrar los adelantos o descuentos por las horas no trabajadas.

**Módulo de Pagos de servicios:** deberá de contar con la forma de registrar los egresos con que cuenta la institución como ser pago de servicios públicos y demás gastos.

Una vez analizado los requerimientos obtenidos se pudo determinar los componentes, módulos e interfaces necesarias para desarrollar el sistema administrativo, para ello se contempla que el sistema indefectiblemente debe contar con:

**Login:** contara con un nombre de usuario y contraseña con la cual podrá acceder al sistema. Solo el administrador, es decir, la persona encargada tendrá acceso al nombre usuario y contraseña ya que no se contempla en el sistema manejo de roles.

**Módulo de Inscripción:** este contará con los mismos campos con lo que contaba el formulario de inscripción que utilizaban de forma impresa estos son: Datos del alumno, número de documento, nombre completo y el curso al cual se matriculara, datos de los padres, tutores o encargados como ser la profesión de cada uno, numero de contactos, dirección.

Para que se pueda inscribir a un alumno, el curso deberá estar creado con anterioridad.

**Módulo de creación de curso:** contará con dos campos una lista de los cursos y otra con la lista de tipo de bachillerato con que cuenta la institución y el arancel correspondiente a cada curso.

**Cuenta corriente para cada alumno:** al inscribir al alumno se generara automáticamente una cuenta corriente correspondiente al curso al cual se inscribió con todas sus obligaciones: inscripción, cuotas, derecho a examen y título.

**Módulo de stock:** cuenta con cuatro campos que hacen referencia al tipo de artículo una descripción del mismo, la cantidad total con la que cuenta y el precio.

**Módulo de caja:** este módulo es el que se verá al inicio del sistema con un campo en el cual deberá de ingresar el monto con el que abrirá la caja, tendrá campos estáticos como el monto de la apertura de caja y la fecha. Esta al terminar el día se deberá cerrar.

**Módulo de pago de servicio:** cuenta con dos campos en el cual se debe de ingresar la descripción del pago abonado y el monto total.

**Módulo de facturación (impresión de facturas):** cada vez que se hace el cobro de una cuota esta le mostrara una vista previa de la factura a imprimir.

Cabe destacar que en cada módulo se pueden imprimir los informes de los mismos.

### **Definición del Entorno Tecnológico del Software**

Una vez analizado todos los requerimientos y determinar cuáles son los componentes necesarios, se procede a la fase de desarrollo del sistema. Se optó por el framework RoR porque agiliza el proceso de desarrollo además de poseer con una inmensa cantidad de plugins que ayudan a este proceso, la metodología utilizada fue la de KANBAN que permite tener de manera

controlada el proceso y avance del proyecto y poder determinar las tareas a las que se le deben de dar prioridad. El IDE fue Sublime Text un editor de código multiplataforma fácil de utilizar.



**Figura 1 – Logo de RoR**

Para el diseño del modelado de datos se utilizó la herramienta MySQL WorkBench que permite diseñar de forma visual la base de datos. En cuanto a su arquitectura se puede decir que es open source. Como Gestor de base de datos se utilizó MySQL por sus diferentes características, por la seguridad e integridad de los datos que ofrece.

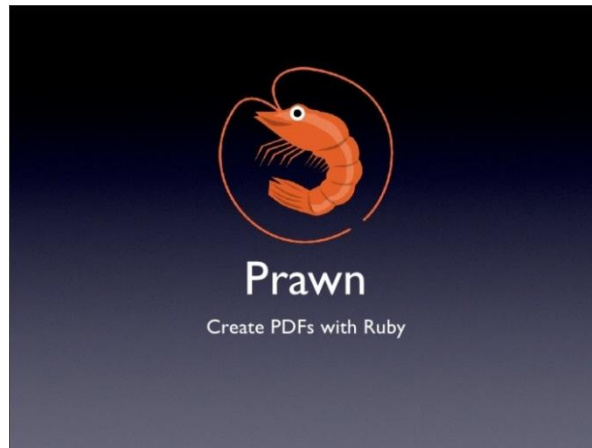


**Figura 2 – Logo de herramientas utilizadas**

Como herramienta colaborativa y control de versiones se optó por el repositorio de GitHub en donde se alojó la documentación y el código fuente del sistema de gestión, posee un visor de ramas donde se puede comparar los progresos realizados en las distintas ramas. En el siguiente link se puede ver el proyecto en cuestión <https://github.com/meleve/SanPi>

Para el modulo caja en la interfaz de factura se utilizó una Gema llamada PRAW que es un generador de Pdf en Ruby que posee una gran funcionalidad, simple de configurar y con las

características de poder ilustrar con vectores que da la posibilidad de dibujar líneas, curvas, elipse, etc. Que hace posible que se tenga el modelo de una Factura.



**Figura 3 – Imagen de generador de pdf**

Para el desarrollo del sistema se utilizaron varias gemas que hicieron posible agilizar algunas funcionalidades con la que debía contar. Dentro del repositorio de las gemas se encuentra el Autocomplete que sirvió para encontrar rápidamente el nombre de un alumno que ya se encontraba registrado en la base de datos.

### **Planificación del Proyecto**

La planificación del proyecto se realizó mediante la herramienta colaborativa Trello, que es una herramienta de gestión de proyectos que mejora la comunicación en el grupo, donde se crean tableros a los cuales se les pueden agregar *cards* con las tareas a desarrollar dentro del proyecto. Se creó cuatro tableros uno de ellos es la lista de tareas en esta se fueron creando las tareas detalladas para el desarrollo del sistema, así también durante el desarrollo se fueron agregando las dificultades con las que se contaba en cuanto a los errores y validaciones, el segundo tablero fue la de tareas en procesos en la cual se iban moviendo las tareas que se estaban desarrollando, en el tercer tablero se agregaban las dificultades con la que se contaban durante el desarrollo del sistema y por ultima el tablero de las tareas terminadas. De esta forma se pudo

lograr el desarrollo del sistema ya que las modificaciones del sistema pueden hacerse sobre la marcha.



Figura 4 – Scream de Trello

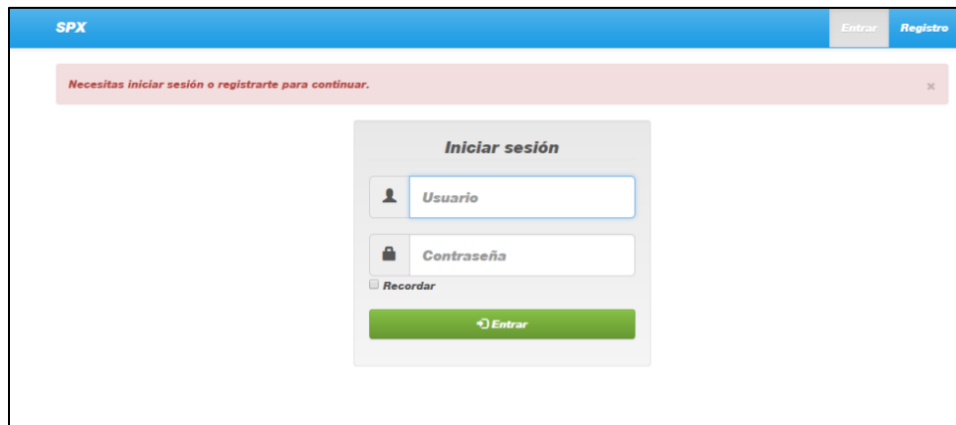
## Desarrollo de la Aplicación

Como último y principal objetivo se propone desarrollar el sistema web de gestión administrativa.

Para la creación de las diferentes interfaces se utilizaron las gemas de Ruby que son paquetes de librerías, estas se encuentran alojadas en un gestor de gemas que actúa como un repositorio llamado RubyGem. Dichas gemas se instalan en el sistema y éstas se pueden utilizar con una llamada que aporta el sistema de gemas de Ruby (Rubio, 2012).

En base a los requerimientos obtenidos la primera vista corresponde al Login que se le asignará al usuario para poder ingresar al sistema en cuestión.





**Figura 5– Aplicación Login**

En este caso, para la interfaz de usuario se utilizó la gema DIVICE. Es una librería que permite al usuario la autenticación, crear cuentas, modificar el perfil, iniciar y cerrar sesión, enviar recordatorios de contraseña, dicha gema cuenta con más características las cuales se las puede ir agregando de acuerdo a la necesidad que requiera el sistema que se esté desarrollando.

En las siguientes vistas (figuras siguientes) se podrá observar que cuentan con un formato de tablas parecidas con la que se pueden obtener los informes, guardarlos e imprimir en el formato PDF y Excel. Además de contar con un buscador. Esto permite que las vistas se mantengan en orden para así poder dar un mejor aspecto al sistema.

Como parte de los requerimientos obtenidos se realizó el módulo de caja creando las validaciones correspondientes, el usuario antes de ingresar al sistema y realizar las tareas cotidianas deberá de ingresar un monto inicial para su caja; una vez hecho dicho paso se crea la caja con la fecha y el monto de apertura.

De esta manera se podrá tener un mejor control de los movimientos de la caja, finalizando el día el usuario deberá cerrar la caja y obtendrá el resumen de los ingresos y egresos del día; la interfaz cuenta con la opción de imprimir los informes en el formato PDF y Excel de manera a poder presentarlas cuando se requiera a las instancias debidas.

Figura 6 - Aplicación Apertura de Caja

Caja Nro	Concepto	Ingreso	Egreso	Saldo
5	Apertura de caja	100000	0	100000
5	Cuota	150000	0	250000
5	Cuota	150000	0	400000
5	Matricula	50000	0	450000
5	Matricula	50000	0	500000
5	Venta de Camisa	30000	0	530000
5	ANDE	0	10000	520000

Figura 7 - Aplicación Informe de Caja

Para el registro de los alumnos deberá primeramente ingresar a través del formulario que proporciona el sistema accediendo desde el botón que posee en la parte superior derecha. Si el alumno ya se encuentra registrado lo podrá buscar con su número de cedula en el primer campo el cual se utilizó una Gema llamada Autocomplete que sirve para obtener de manera más rápida el nombre de un alumno que ya este registrado en la base de datos. En el caso de que el alumno sea nuevo en la institución esta se deberá completar el formulario con los datos requeridos.

**REGISTRO DE ALUMNO**

Ingrese un nombre para buscar

Nombre:  Apellido:  Fecha de nacimiento:

Lugar de nac:  Domicilio:  Nombre de la madre o tutor:

Profesion de la madre:  Contacto:  Nombre del padre o tutor:

Profesion del padre:  Contacto:

Curso:

[← Atras](#)

**Figura 8 – Aplicación Matriculación**

Una vez registrado se obtiene una lista con todos los alumnos registrados en la institución, en esta vista se podrá editar la información de los alumnos en caso de que sea necesario.

En la **Figura 9** muestra el listado de los alumnos y en la parte de arriba el botón mencionado anteriormente.

**Alumnos**

Buscar:

C.I.Nº	Nombres y Apellidos	
235465	Juan Gimenez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
875454	Orlando Cubilla	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
875454	David Lopez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
5645667	Milagros Gomez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
5645667	Milagros Gomez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
6543245	Jorge Gomez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
23345546	Jose Perez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
23345546	Jose Perez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>
23455764	Liza Benitez	<input type="button" value="o"/> <input type="button" value="✎"/> <input type="button" value="✕"/>

Mostrando de 1 al 9 de 9 registros

**Figura 9 – Aplicación Listado de alumnos**

Otro de los requisitos consistía en diseñar una interfaz de modo a que se pueda buscar por el nombre de los padres en el caso de que sean dos o más hermanos. De esta forma se podrá agilizar el proceso de pago. En la **Figura 10** se puede observar la función de buscar por nombre de Padre como se había mencionado anteriormente.

The screenshot shows a web application interface for a school's administrative system. At the top is a navigation bar with the following items: SPX, Alumnos, Buscar por (highlighted), Curso-Matriculación, Productos, Pagos de servicios, Caja, Factura, mef@gmail.com, and Salir. Below the navigation bar is a section titled 'Buscador por Padre'. It features a search input field labeled 'Nombre y Apellido' with a dropdown arrow, and a green 'Buscar' button. Below the search field, there are two results listed, each with a checkbox and the name of the parent: 'Melanie Florentin' and 'Micaela Florentin'. Under each name, there is a grid of checkboxes for various fees: '1 Matricula' (checked), '3 Cuota', '6 Cuota', '9 Cuota', '2 Derecho de Examen', '1 Cuota', '4 Cuota', '7 Cuota', '10 Cuota', '2 Cuota', '5 Cuota', '8 Cuota', and '1 Derecho de Examen'. At the bottom of the results section, there is a green 'Pagar' button and a blue 'Atras' button.

**Figura 10– Aplicación Buscador por padres**

Para poder hacer todas las acciones dichas anteriormente primeramente deberá estar creado el curso con sus aranceles correspondiente. En esta interfaz deberá de indicar el curso a crear, la especialidad y el año correspondiente y luego ir agregando los respectivos detalles del curso que serían el monto y la cantidad de la matrícula, cuotas, derecho a examen y título en el caso de que sea el último curso. En el caso de que el alumno tenga un descuento se creara un curso especial con aranceles diferentes.

Figura 11 – Aplicación Creación de curso

En la vista siguiente se muestra los cursos creados en la cual se pueden acceder en los detalles de las mismas.

Curso	Especialidad	Seccion	Año			
1°	Bachillerato con énfasis en Ciencias Sociales	Unica	2017	👁	✎	✕
2°	Bachillerato con énfasis en Ciencias Sociales	Unica	2017	👁	✎	✕

Figura 12 – Aplicación Curso Creado

Volviendo al punto anterior en la etapa de matriculación del alumno se crean por detrás la cuenta corriente del mismo con todas sus obligaciones de acuerdo al curso en el que se haya matriculado.

Para la creación de esta interfaz se utiliza el controlador donde se escribió la lógica para que en esta se puedan hacer los pagos parciales o totales de la deuda. La interfaz muestra la lista con las obligaciones del alumno con tres opciones, una de ella es la de pago parcial la cual se realiza accediendo desde la opción pagar como se muestra en la imagen ahí podrá ingresar el monto parcial el cual será debitado de la deuda y se mostrara el saldo pendiente. La segunda opción es la de marcar varios mediante el chek box a la vez se puede marcar un producto y la cantidad al cual se puede acceder mediante el botón que muestra en la parte superior y la tercera opción es la de seleccionar todo de manera a cancelar la deuda del año, en todas estas opciones luego de pagar se cambia el estado de las mismas a pagado, así también se obtiene como comprobante la factura de las transacciones.

SPX

Alumnos

Buscar por

Curso-Matriculación

Productos

Pagos de servicios

Caja

Factura

met@gmail.com

Salir

Cuenta Corriente

Alumno:

Melanie Florentin

Curso:

1° Bachillerato con énfasis en Ciencias Sociales 2017

Saldo:

1460000

N°:

Agregar Producto

+

csv

Buscar:

Cuota N°	Descripción	Importe	Estado		
1	Matricula	0	Pagado	Pagar	
1	Cuota	10000	Pendiente	Pagar	
1	Derecho de Examen	25000	Pendiente	Pagar	
2	Cuota	150000	Pendiente	Pagar	
2	Derecho de Examen	25000	Pendiente	Pagar	
3	Cuota	150000	Pendiente	Pagar	
4	Cuota	150000	Pendiente	Pagar	

Figura 13 – Aplicación Cuenta Corriente del alumno

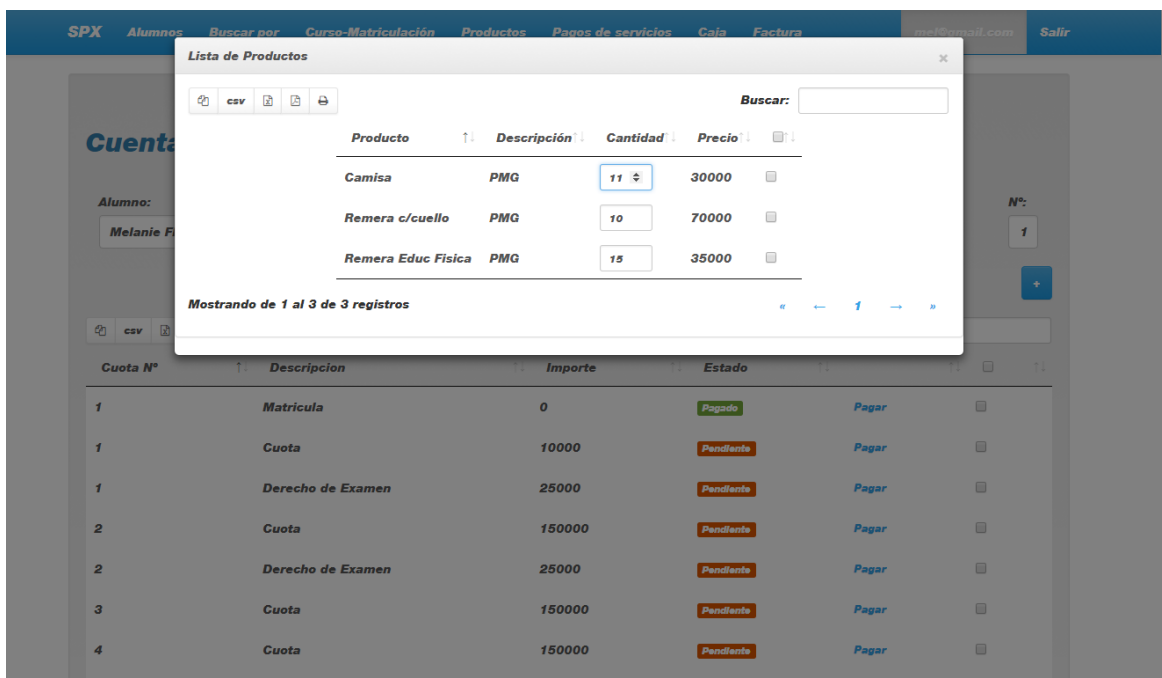


Figura 14 – Aplicación Marcar producto



Figura 15 – Aplicación Pagos parciales

En el módulo de curso se puede acceder a los detalles del curso donde se puede ver los alumnos matriculados en el mismo, además cuenta con dos opciones para matricular al curso superior, una de ella es la matriculación individual y la otra es la matriculación masiva en la cual se debe de marcar todos los alumnos y seleccionar la acción correspondiente.

SPX	Alumnos	Buscar por	Curso-Matriculación	Productos	Pagos de servicios	Caja	Factura	me@gmail.com	Salir
-----	---------	------------	---------------------	-----------	--------------------	------	---------	--------------	-------

<b>Curso</b>					
Curso: 1º Bachillerato con énfasis en Ciencias Sociales			Seccion: Unica		Año: 2017
<a href="#">←Salir</a> <a href="#">/Editar</a>					
<div> <div>📄</div> <div>csv</div> <div>📄</div> <div>📄</div> <div>📄</div> </div>		<div> <div>Buscar:</div> <input type="text"/> </div>			
Alumno	↑↓	Curso	↑↓	Año	↑↓
Jose Perez		1º		2017	<div> <div>📄</div> <div>🔍</div> </div>
Juan Gimenez		1º		2017	<div> <div>📄</div> <div>🔍</div> </div>
Milagros Gomez		1º		2017	<div> <div>📄</div> <div>🔍</div> </div>
Mostrando de 1 al 3 de 3 registros					
<div> <div>«</div> <div>←</div> <div>1</div> <div>→</div> <div>»</div> </div>					
<div> <div>Matricular todo</div> <div>Matricular todo</div> <div>Matricular</div> </div>					

Figura 16 – Aplicación Alumnos matriculados por curso

Otro de los requerimientos fue la de crear una interfaz de productos, para tener un inventario de los mismos y en base a eso poder actualizar el estado de stock. En la interfaz deberá ingresar el nombre y descripción del producto, el precio de la misma al igual que la cantidad existente. En ella también se podrá ver la lista de los productos con que cuenta y se podrá actualizar el stock si así lo requiera. En el caso de que no el producto no esté más a la venta se podrá eliminar el producto.

En la Figura 12 se observa la interfaz mencionada. En la parte superior derecha cuenta con el botón para agregar nuevos productos. Así también los botones de actualizar stock y eliminar.





Para la creación de la interfaz de Caja también se ha utilizado el controller para la lógica, en el cual se definió la tarea que es la de sumar o restar los ingresos o egresos. En la interfaz se puede observar el resultado en la tabla.

Caja Nro	Concepto	Ingreso	Egreso	Saldo
6	Apertura de caja	100000	0	100000
6	Cuota	5000	0	105000
6	Cuota	15000	0	120000
6	Cuota	50000	0	170000
6	Cuota	10000	0	180000
6	Derecho de Examen	25000	0	205000
6	Cuota	100000	0	305000
6	Cuota	150000	0	455000

Figura 19 – Aplicación Informe de caja

Se utilizó la gema PRAW para imprimir la factura en formato PDF. Praw es una gema con una cantidad inmensa de funcionalidades, con soporte de dibujo vectorial, posee una variedad de herramientas de bajo nivel para las necesidades básicas del diseño.

En todas las interfaces al realizar los pagos ya sean parciales y totales se muestra una vista previa de la factura con los detalles de pagos a nombre de uno de los padres y en la descripción se muestra el nombre del o los alumnos en el caso de que se pague en forma conjunta, esta se pueden descargar en el formato PDF como comprobante.

N° Factura: 28  
Fecha: 10/30/2017

Institucion:  
Colegio Parroquial Priv. "San Pio X"

Tel:  
(071)203778

Razon Social:  
JazminFlorez

TOTAL:  
80000

Pago:  
Contado

**FACTURA**

N°	Descripción	Importe
1	Matricula - Jazamin Florez	50000
	Pago de: Camisa - Jazamin Florez	30000
<b>TOTAL:</b>		<b>80000</b>

Dirección Padre Kreusser c/Juan León Mallorquin  
Correo: [colegio.spx@gmail.com](mailto:colegio.spx@gmail.com)

[Salir](#) [Factura](#)

Figura 20 – Aplicación Vista preliminar de Factura

**Colegio Parroquial Privado San Pio X**  
Padre Jose Kreusser N°306 e/Juan Leon Mallorquin  
Telef: (071)203778 Encarnacion - Paraguay

Timbrado N° XXXXXXXX  
**RUC: 80056885-0**  
Inicio Vigencia: 05/septiembre/2016  
Fin de Vigencia: 30/septiembre/2017  
**FACTURA N° 28**

Fecha: 10/30/2017  
Nombre: Jazamin Florez

N°	Descripción	Importe
1	Matricula	50000
	Pago de: Camisa	30000
<b>TOTAL</b>		<b>80000</b>

Figura 21 – Aplicación Vista preliminar de Factura en formato PDF

En el controlador y las vistas se definieron toda la lógica y validaciones correspondientes a las funcionalidades del sistema.

El usuario no necesitara una capacitación previa ya que el sistema posee una interfaz amigable, intuitiva y fácil de usar.

Como en el marco de este proyecto no se contempla la implantación del sistema, no se procederá a la misma.

Al decidir desarrollar un sistema de gestión administrativa implico realizar una investigación previa sobre los conceptos que se utilizan en el ramo para así poder adecuarlos a los requerimientos del sistema. Como el software se desarrolló bajo la metodología de código abierto y como se había mencionado anteriormente para poder ser software de código abierto debe cumplir ciertos requisitos, es por eso que se optó por las herramientas open source para el desarrollo del mismo, desde el framework de programación hasta el gestor de base de datos, la ventaja de esto es que el desarrollo del sistema no tiene costo alguno.

En cuanto al framework de desarrollo puedo decir que al iniciar con el proyecto resulta fácil y más rápido pero a medida que fue avanzando, debido a las librerías que se fue agregando el tiempo de carga de las interfaces fue más lenta. Una de las soluciones que encontré fue la de cambiar el servidor que tiene por defecto Rails y ejecutarlo en modo producción. Cabe mencionar que el uso de las librerías debe ser moderado y en base a lo que el sistema requiera.

### **Pruebas y correcciones**

Durante el desarrollo del sistema se realizaron pruebas de caja negra con el personal administrativo de la institución que será el usuario final del sistema de gestión. Las mismas consistían en ingresar al sistema e ir recorriendo los módulos e interfaces con que contaba el sistema para probar el funcionamiento de cada una de ellas.

Algunas de las pruebas realizadas fueron la comprobación de valores límites para algunos campos, como en el caso de los pagos parciales no permitirá ingresar un valor más alto al de lo adeudado mostrando una alerta.

Pruebas de validaciones de los campos, no se podrán dejar campos obligatorios en blanco si lo hacen el sistema volverá a cargar la misma interfaz mostrando el error.

Pruebas de rendimiento del sistema con la carga de ciertas cantidades de datos comprobando si lo hace lento.

Así también se realizaron las pruebas en el navegador Google Chrome y Mozilla Firefox para comprobar el diseño responsivo del sistema.

A medida que se iban realizando las pruebas se fueron detectando los errores los cuales se elaboró una lista la cual se fue registrando en la herramienta colaborativa Trello dándoles prioridades a las más importantes.

## **Conclusión**

Durante el proceso de la recolección de datos, se describió el problema principal con el que contaba la institución de esa manera se ha determinado el impacto que tendrá el sistema administrativo en la Institución, como se describió anteriormente las instituciones cuentan con unas innumerables tareas que deben realizarlas manualmente afectando a un factor determinante que es de suma importancia, el “tiempo”. Todas estas tareas se necesitan acelerar para asegurar la productividad.

La innovación de la tecnología nos permite descubrir y desarrollar nuevas herramientas que se adecuen con este tipo de tareas y adaptarlas a la realidad de cada institución.

Para ello se realizó una indagación en la web y un sondeo a cuatro empresas de la ciudad de Encarnación donde no se ha encontrado sistemas con los requisitos obtenidos mediante el instrumento de recolección de datos, de esta manera se pudo determinar los módulos e interfaces necesarias para ser desarrollada.

Así fue posible desarrollar el sistema de acuerdo a los requisitos obtenidos mediante la entrevista con el personal administrativo, con la cual se obtuvo las funcionalidades con la que debería de contar el sistema de gestión y en base a esos datos se optó por desarrollar con el framework de programación RoR por el hecho de que simplifica las tareas repetitivas y además porque poseen una serie de plugins las cuales se pueden agregar fácilmente y modificarlos.

El resultado del desarrollo fue un sistema que permite tener de manera ordenada el control de ingresos y egresos mediante el procesamiento de datos de los registros administrativos teniendo un acceso inmediato a las informaciones que requiera en su momento. Estas informaciones son guardadas en una base de datos cuidando la integridad de las mismas.

## **Recomendaciones y Trabajos Futuros**

- Se recomienda como líneas futuras agregar funcionalidades:
  - a. El registro de las horas trabajadas con un sensor de huella dactilar.
  - b. Registro de los pagos a los docentes.
  - c. Registrar los adelantos.
  - d. Calcular las horas trabajadas.
  - e. La creación de roles de manera a que el plantel docente pueda contar un usuario e integrar la parte académica de parte de los docentes en donde puedan registrar el proceso de los alumnos y poder elaborar un informe del proceso parcial del alumno.
- Migrar a una arquitectura Software como Servicio (SaaS) que permita que muchas instituciones lo utilicen libremente desde la nube.
- La implantación del sistema web de gestión administrativa.

## Lista de Referencias

Ley Nro 1.264 General de Educación. (25 de mayo de 1998). *Ministerio de Educación y Ciencia*.

Obtenido de [https://www.mec.gov.py/cms\\_v2/resoluciones/16-ley-12641998](https://www.mec.gov.py/cms_v2/resoluciones/16-ley-12641998)

*Ministerio de Educacion y Cultura*. (26 de mayo de 1998). Obtenido de

[https://www.mec.gov.py/talento/Normativas/ley\\_1264.pdf](https://www.mec.gov.py/talento/Normativas/ley_1264.pdf)

*Ministerio de Educacion y Cultura*. (26 de 05 de 1998). Obtenido de

[https://www.mec.gov.py/cms\\_v2/resoluciones/16-ley-12641998](https://www.mec.gov.py/cms_v2/resoluciones/16-ley-12641998)

*GRUPO I.A.G.* (2004). Obtenido de <http://www.grupoia.com/faq/que-es-un-sistema-de-gestion-o-mis>

*EcuRed*. (14 de diciembre de 2010). Obtenido de <https://www.ecured.cu/Framework>

*GENBETA*. (9 de 02 de 2012). Obtenido de <http://www.genbeta.com/herramientas/sublime-text-un-sofisticado-editor-de-codigo-multiplataforma>

*OpenWebinars*. (8 de noviembre de 2016). Obtenido de <https://openwebinars.net/blog/los-7-mejores-frameworks-de-java-de-2016/>

*SQL Server*. (2016). Obtenido de <https://www.microsoft.com/es-xl/sql-server/sql-server-2016>

Acosta, J. C. (2012). *Medición de atributos POO en frameworks de desarrollo PHP*. Argentina: In XVIII Congreso Argentino de Ciencias de la Computación.

Aguilera, S. (2015). *Tipos de Software*. Obtenido de

[http://repositorio.ub.edu.ar/bitstream/handle/123456789/5213/FInform-502-U4-7-](http://repositorio.ub.edu.ar/bitstream/handle/123456789/5213/FInform-502-U4-7-TiposdeSw-2015.pdf?sequence=1&isAllowed=y)

[TiposdeSw-2015.pdf?sequence=1&isAllowed=y](http://repositorio.ub.edu.ar/bitstream/handle/123456789/5213/FInform-502-U4-7-TiposdeSw-2015.pdf?sequence=1&isAllowed=y)

*aulaFormativa*. (s.f.). Obtenido de <http://blog.aulaformativa.com/listados-ruby-frameworks/>

*aulica*. (s.f.). Obtenido de <http://www.aulica.com.ar/>



aulica. (s.f.). *aulica.com*. Obtenido de <http://www.aulica.com.ar/>

Carlo, L. (s.f.). *Prezi*. Obtenido de [https://www.google.com.py/?gws\\_rd=ssl#q=que+es+trello](https://www.google.com.py/?gws_rd=ssl#q=que+es+trello)

Cavassa, C. R. (2004). *La Gestion administrativas en las Instituciones educativas*. Mexico:

Noriega.

colegiosyliceos.com. (2000). *colegiosyliceos.com*. Obtenido de

<http://www.colegiosyliceos.com/gescolarcm.html>

Color, A. (13 de mayo de 2009). Principios de la Gestion Administrativa. *ABC Color*.

Consejo Departamental de Educación. (01 de Noviembre de 2016). Plan Educativo

Departmental. Asunción, Cental, Paraguay.

*De Conceptos*. (s.f.). Obtenido de <http://deconceptos.com/ciencias-sociales/caja-en-contabilidad>

*Definición.De*. (s.f.). Obtenido de <http://definicion.de/gestion/>

*Definición.DE*. (s.f.). Obtenido de <http://definicion.de/colegio/>

*DefiniciónABC*. (s.f.). Obtenido de <http://www.definicionabc.com/economia/administrativa.php>

*EcuRed*. (s.f.). Obtenido de [http://www.ecured.cu/Sistema\\_inform%C3%A1tico](http://www.ecured.cu/Sistema_inform%C3%A1tico)

*FracktalWeb*. (s.f.). Obtenido de <http://fraktalweb.com/blog/sistemas-web-para-que-sirven/>

Free Software Fundation. (1985). Obtenido de <https://www.fsf.org/es>

Garzas, J. (22 de 11 de 2011). *JavierGarzasBlog*. Obtenido de

<http://www.javiergarzas.com/2011/11/kanban.html>

Internet, B. d. (15 de mayo de 2008). *Informática XP*. Obtenido de

<http://informaticaxp.net/clasificacion-y-tipos-de-software>

Joaquin. (15 de mayo de 2008). *InformáticaXP*. Obtenido de

<http://informaticaxp.net/clasificacion-y-tipos-de-software>

Lopez, J., & Perez, P. (2012). *ASP .NET for newbies*. Espanha: Abc.

Marqués, M. (2011). Sistema de Gestión de base de datos. En M. Marqués, *Base de Datos* (pág. 3).

Martinez, R. (02 de 10 de 2010). *PostgreSQL - es*. Obtenido de

[http://www.postgresql.org.es/sobre\\_postgresql](http://www.postgresql.org.es/sobre_postgresql)

*Master Magazine*. (s.f.). Obtenido de <http://www.mastermagazine.info/termino/7216.php>

*monografias.com*. (s.f.). Obtenido de <http://www.monografias.com/trabajos25/gestion-administrativa/gestion-administrativa.shtml#gestion>

Mussa, Y. (2008). *Fx2*. Obtenido de <http://fx2.com.uy/mysql-un-aliado-para-la-gestion-de-base-de-datos>

PDCA Home. (2012). *PDCA*. Obtenido de <http://www.pdcahome.com/metodo-kanban/>

Perez, J. (2010). *Técnicas de programación etc*. Obtenido de unae.edu: <https://www.unae.edu.py>

Pressman, R. S. (2010). Administración de la Configuración del Software. En R. S. Pressman, *Ingeniería del Software Un Enfoque Práctico* (pág. 501). MC GRAW HI.

Pressman, R. S. (2010). El Software y la Ingeniería del Software. En R. S. Pressman, *Ingeniería del Software- Un enfoque práctico* (pág. 3). Mexico: MC Graw Hill.

proemsa. (s.f.). *Software de Administración para Colegios o Centros Educativos*. Obtenido de <http://www.proemsaofware.com/software-de-administracion-para-colegios-o-centros-educativos/>

*RAE* . (s.f.). Obtenido de Real Academia Española: <http://www.rae.es/>

Rivarola, D. M. (2000). Obtenido de <http://archivo.cepal.org/pdfs/2000/S00090772.pdf>

Rubio, M. (2 de enero de 2012). *Altenwald*. Obtenido de <http://altenwald.org/2012/01/02/las-gemas-de-ruby/>

*Saberes*. (s.f.). Obtenido de <http://www.sistemasaberes.com/>

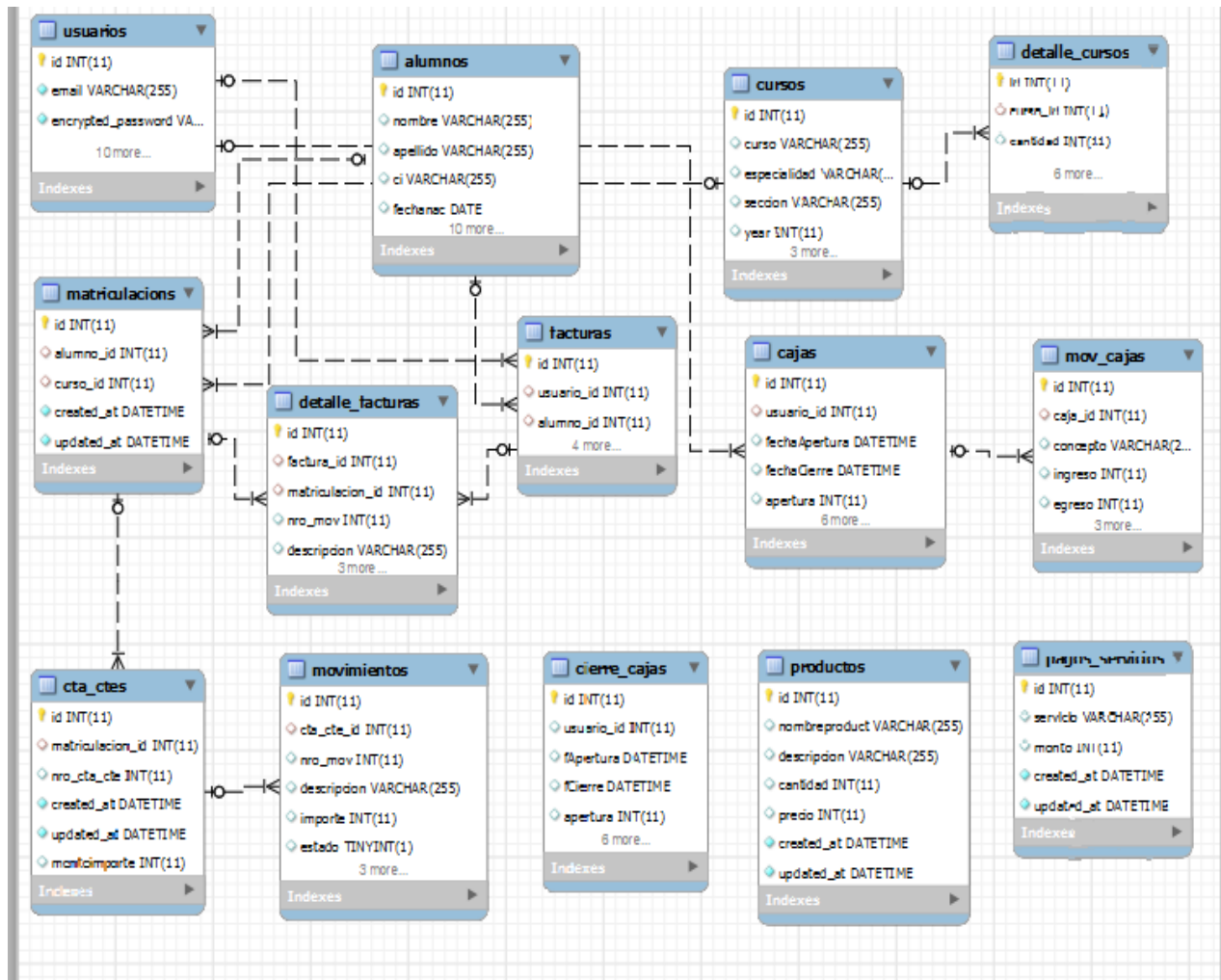
- Sampieri. (2010). Metodología de la investigación. En R. Sampieri, *Metodología de la investigación*. McGrawHi.
- Sampieri, R. H. (2010). *Metodologia de la investigación*. Mexico: MC GRAW HILL.
- Sampieri, R. (s.f.). Metodología de la Investigación.
- SistemaSaberes. (2009). *SistemaSaberes*. Obtenido de <http://www.sistemasaberes.com/para-que/>
- Spano, D. d. (2010). *El open source como facilitador del open access. Impacto y visibilidad de las revistas científicas*. Obtenido de [https://s3.amazonaws.com/academia.edu.documents/36153922/ponencia\\_spano\\_elis.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1503963768&Signature=%2F80MmzTLjgSoK%2F13NXS2jitD%2Fqc%3D&response-content-disposition=inline%3B%20filename%3DEl\\_Open\\_Source\\_como\\_TIOBE](https://s3.amazonaws.com/academia.edu.documents/36153922/ponencia_spano_elis.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1503963768&Signature=%2F80MmzTLjgSoK%2F13NXS2jitD%2Fqc%3D&response-content-disposition=inline%3B%20filename%3DEl_Open_Source_como_TIOBE).
- TIOBE. (2015). *TIOBE*. Obtenido de <https://www.tiobe.com/tiobe-index/>
- Vargas, G. (31 de marzo de 2009). Obtenido de MG Mejora tu Gestion: <http://mejoratugestion.com/mejora-tu-gestion/que-es-un-sistema-de-gestion/>

## Anexos

### Anexo 1: Enlace al proyecto

Proyecto: <https://github.com/meleve/SanPi/tree/master/sanpioxadmin>

### Anexo 2: Modelado del Sistema



### **Anexo 3: Entrevista realizada**

#### **Entrevista al Personal Administrativo del Colegio Parroquial Privado San Pio X.**

**1- ¿Cómo realiza el cobro de las cuotas?**

El cobro de cuotas se realiza los primeros días del mes, siendo las fechas de vencimiento del uno al cinco de cada mes. Se realiza por medio de planillas con la lista de alumnos y los meses correspondientes. Además se emite boleta legal como comprobante de pago.

**2- ¿De qué manera registra el cobro de las cuotas?**

Se registra en las planillas que contiene la lista de alumnos, los cursos y meses correspondientes. Además de los duplicados de las boletas, los cuales se archivan.

**3- ¿Cómo registra el pago de los servicios?**

El pago de servicios se realiza de maneras diferentes, en algunos casos la empresa envía cobradores, y en los otros casos se paga en cooperativas o bancos.

**4- ¿De qué manera se realiza el pago a los docentes?**

El pago a docentes se realiza en forma quincenal, y se registra por medio de planillas, se utiliza auto factura en algunos casos, y con facturas en el caso de los docentes que cuentan con ella.

**5- ¿Qué medio utiliza para la matriculación de los alumnos?**

La matriculación se realiza por medio de fichas de inscripción, con la firma de los padres o encargados.

**6- ¿Utiliza algún sistema o tiene alguna forma de resguardar sus datos? ¿Cuál?**

Sí. Las fichas con los datos de alumnos y sus padres o encargados se archivan en biblioratos, dividiéndolos por cursos.

**7- ¿Le resulta fácil utilizar las planillas para registro de sus cobros de cuotas y pagos de servicios? ¿Por qué?**

Resulta fácil a veces, ya que se tiene a mano la carpeta, en la mesa de oficina. Pero a la vez un poco dificultoso ya que algunas veces se traspapela con otras carpetas.

**8- ¿De qué manera obtiene los contactos de los padres de alumnos?**

A través de las fichas de inscripción.

**9- ¿Cómo registra los documentos que son entregados por los alumnos?**

Los documentos se archivan en los biblioratos, divididos por cursos.

**10- ¿De qué manera realiza los informes de egresos e ingresos?**

Los informes de ingresos y egresos se realizan de manera semanal, registrando los ingresos de cada día en planillas de Excel, los cuales se imprimen al finalizar la semana y se registra en bibliorato.

**11- ¿Cómo obtiene un informe de los alumnos que están con cuentas pendientes?**

Por medio de la observación de planillas donde se registran los pagos, los cuales son observados continuamente por la administración.

**12- ¿Le gustaría de alguna forma automatizar el proceso de gestión administrativa y de esa forma agilizar su trabajo? ¿Por qué?**

Sí sería bueno, ya que optimizaría el tiempo y además que las planillas corren peligro de desaparecer, mojarse o quemarse, eso haría desaparecer registros valiosos.

**13- ¿Se realizan pagos parciales? ¿Cómo?**

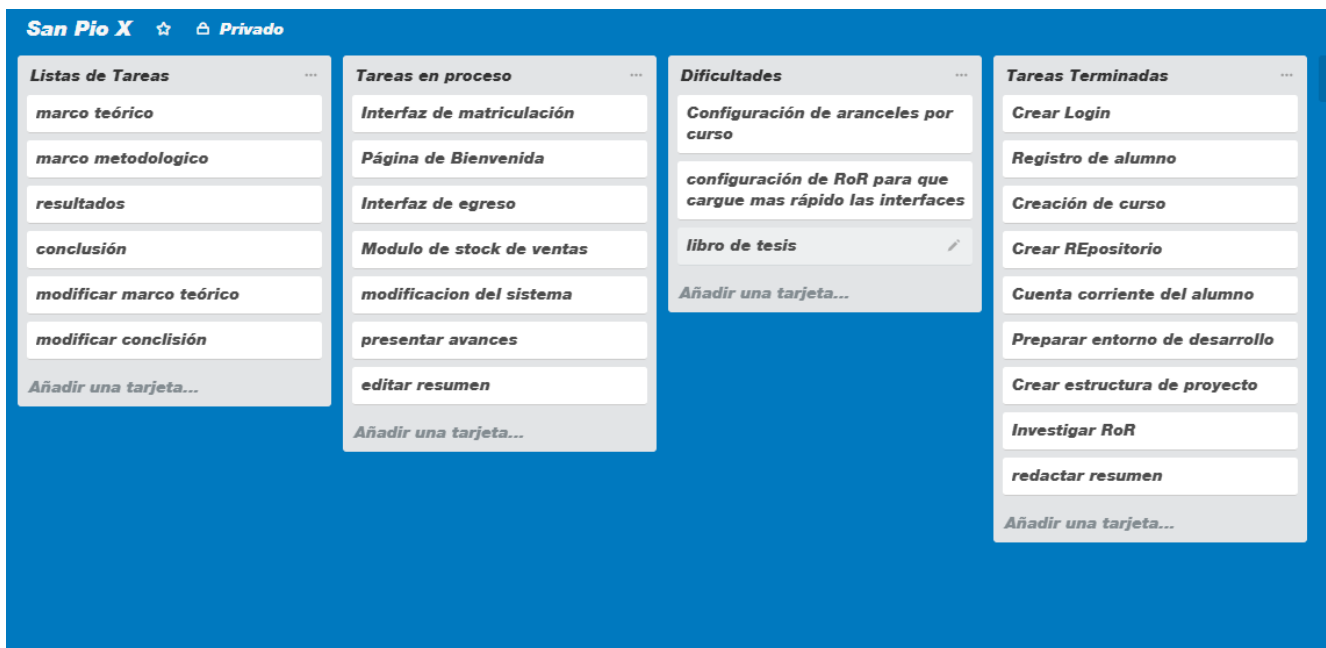
Si, los alumnos traen de acuerdo a lo que pueden y en el mes van cancelando su deuda.

## 14- ¿Cuenta con productos como remeras, libros, insignias y demás a la venta? ¿De qué manera lleva el stock de las mismas?

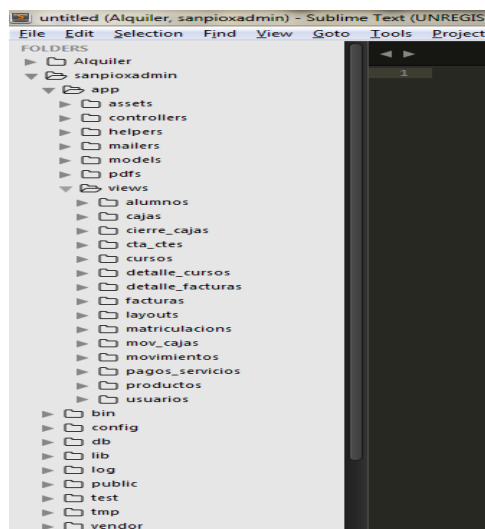
Si y se registran en una planilla la cual no la actualizamos constantemente.

### Anexo 4: Herramienta colaborativa

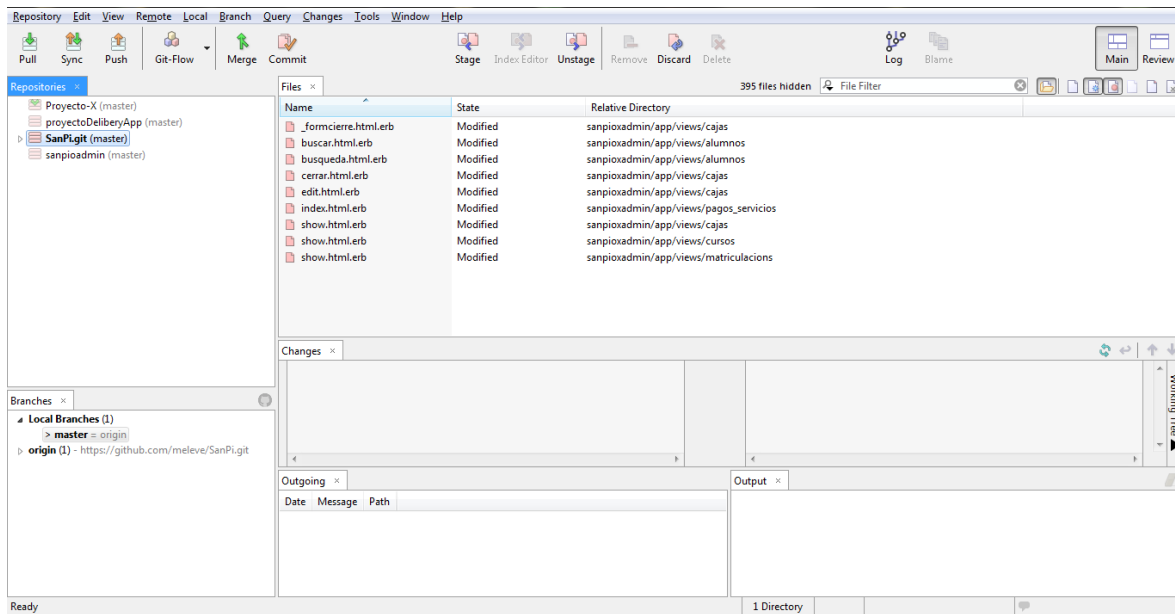
Como herramienta colaborativa se eligió Trello que ayuda a organizar las tareas con los tableros que ofrece para dividir de acuerdo a las dificultades.



### Anexo 5: Estructura del Proyecto



## Anexo 6: Repositorio local SmartGit



## Anexo 7: Código del Proyecto – Controlador alumno

Código utilizado en el controlador del alumno para crear las cuentas corrientes de los mismos una vez que se matriculan. Utiliza un each para generar de acuerdo a la cantidad de matrícula, cuota, derecho a examen y título que se ingresa en la vista.

```
Def create

  @alumno = Alumno.new(alumno_params)

  respond_to do |format|
    if @alumno.save
      @matriculacion = Matriculacion.create(alumno_id: @alumno.id, curso_id:
params[:curso][:curso_id])
      if @matriculacion
        @cta_cte = CtaCte.new
        @cta_cte.matriculacion_id = @matriculacion.id
        @cta_cte.nro_cta_cte = @matriculacion.id
        @cta_cte.save

        @detalle_cursos = DetalleCurso.where(curso_id:
@matriculacion.curso_id)
```



```

      @movimiento = Movimiento.new

      @detalle_cursos.each do |detalle|
        (1..detalle.cantidad).each do |i|

          @movimiento = Movimiento.create(cta_cte_id: @cta_cte.id, nro_mov:
i, descripcion: detalle.descripcion , importe: detalle.importe , estado: false)

          sumaimporte = @cta_cte.montoimporte.to_i + detalle.importe.to_i

          @cta_cte.update(montoimporte: sumaimporte)

        end
      end

      format.html { redirect_to @matriculacion, notice: 'Se ha m registrado
correctamente.' }

      format.json { render :show, status: :created, location: @matriculacion }

    end

    format.html { redirect_to @matriculacion, notice: 'Se ha registrado
correctamente.' }

    format.json { render :show, status: :created, location: @matriculacion }

  else

    format.html { render :new }

    format.json { render json: @alumno.errors, status: :unprocessable_entity }

  end

end

end

end

```

## Anexo 8: Código del Proyecto – Buscador por padres

Código perteneciente al buscador por padres. Su función es buscar a los alumnos que sean hermanos por el nombre de los padres o encargados. Esto trae como resultado el nombre de los alumnos con sus cuentas corrientes.

```

Def buscar
    @alumno = Alumno.new
end

Def busqueda
    @alumno = Alumno.new
    @alumnos = Alumno.where(nombrema: params[:alumno][:nombrema])
end

```

## Anexo 9: Código del Proyecto – Creación de cursos

Parte del código perteneciente al controlador de cursos en donde se crean los detalles del mismo en base a las cantidades ingresadas a través de la vista.

```

Def create
    @curso = Curso.new(curso_params)
    #CREA LOS DETALLES DEL CURSO, LAS CANTIDAD DE CUOTAS
    respond_to do |format|
        if @curso.save
            @detalle_cursos = DetalleCurso.where(curso_id: @curso.id)
            @resultado=0
            @detalle_cursos.each do |detalle|
                @resultado = detalle.cantidad * detalle.importe
                if (@curso.montototal != nil)
                    @curso.update(montototal: @resultado + @curso.montototal)
                else
                    @curso.update(montototal: @resultado)
                end
            end
            format.html { redirect_to @curso, notice: 'El curso se creo correctamente' }
            format.json { render :show, status: :created, location: @curso }
        else
            format.html { render :new }
            format.json { render json: @curso.errors, status: :unprocessable_entity }
        end
    end
end
end
end

```

## Anexo 10: Código del Proyecto – Matriculación

Código perteneciente al controlador de matriculación. Se encarga de crear por medio de un each y los detalles dados la cuenta corriente por alumno.

```
def create

  @matriculacion = Matriculacion.new(matriculacion_params)
  respond_to do |format|
    if @matriculacion.save
      @cta_cte = CtaCte.new
      @cta_cte.matriculacion_id = @matriculacion.id
      @cta_cte.nro_cta_cte = @matriculacion.id
      @cta_cte.save
      @detalle_cursos = DetalleCurso.where(curso_id: @matriculacion.curso_id)
      @suma = 0
      #CREA LA CUENTA CORRIENTE PARA EL ALUMNO
      @movimiento = Movimiento.new
      @detalle_cursos.each do |detalle|
        (1..detalle.cantidad).each do |i|
          @movimiento = Movimiento.create(cta_cte_id: @cta_cte.id, nro_mov: i,
descripcion: detalle.descripcion , importe: detalle.importe , estado: false)
          @cta_ct = CtaCte.find(@cta_cte.id)
          @suma += detalle.importe
          @cta_ct.update(montoimporte: @suma)
        end
      end
      format.html { redirect_to @matriculacion, notice: 'Se creo
correctamente' }
      format.json { render :show, status: :created, location: @matriculacion
}
    else
      format.html { render :new }
      format.json { render json: @matriculacion.errors, status:
:unprocessable_entity }
    end
  end
end
```

## Anexo 11: Código del Proyecto – Matriculación 2

Código perteneciente al controlador de matriculación. Funciona al momento de seleccionar varias cuotas u obligaciones al mismo tiempo para pagarlas juntas.

```
def edit_multiple

  @matriculaciones = Matriculacion.find(params[:matriculaciones_id])
  end

  #SELECCION MULTIPLE PARA PAGAR VARIOS

def update_multiple

  params[:matriculaciones].values.each do |matriculacion|
    @matriculacion = Matriculacion.create(alumno_id:
matriculacion['alumno_id'].to_i, curso_id:
matriculacion['curso_id'].to_i)
    @cta_cte = CtaCte.new
    @cta_cte.matriculacion_id = @matriculacion.id
    @cta_cte.nro_cta_cte = @matriculacion.id
    @cta_cte.save
    @detalle_cursos = DetalleCurso.where(curso_id:
@matriculacion.curso_id)
    @movimiento = Movimiento.new
    @detalle_cursos.each do |detalle|
      (1..detalle.cantidad).each do |i|
        @movimiento = Movimiento.create(cta_cte_id: @cta_cte.id,
nro_mov: i, descripcion: detalle.descripcion , importe:
detalle.importe , estado: false)
        sumaimporte = @cta_cte.montoimporte.to_i +
detalle.importe.to_i
        @cta_cte.update(montoimporte: sumaimporte)
      end
    end
  end
  redirect_to cursos_url
end
```

## Anexo 12: Código del Proyecto – Movimientos

Este código pertenece al controlador del movimiento este se ejecuta cuando se realizan los pagos parciales generando vista previa de la factura con los detalles correspondientes.

```
def create
```

```
@movimiento = Movimiento.new(movimiento_params)
respond_to do |format|
  if @movimiento.save
    format.html { redirect_to @movimiento, notice: 'Se ha registrado
correctamente.' }
    format.json { render :show, status: :created, location: @movimiento }
  else
    format.html { render :new }
    format.json { render json: @movimiento.errors, status:
:unprocessable_entity }
  end
end

#actualiza el estado de la cuenta corriente
def update
  respond_to do |format|
    @monto = 0
    timporte = 0
    impo = 0
    cMonto = 0
    mTotal = 0
    resultado = 0
    @total = 0
    id = 0
    @cta_ct = CtaCte.find(@movimiento.cta_cte_id)
    #el monto impo = es el total del importe adeudado
    #el mTotal = a la entrega parcial o total que realiza
    #se resta el importe parcial o total con el importe real
    if @movimiento.estado == false
      if @movimiento.update(movimiento_params)
        impo = @movimiento.importe
        mTotal = @movimiento.totalimporte
        resultado = impo.to_i - mTotal.to_i
        if @movimiento.update(importe: resultado)
          if @movimiento.importe == 0
            @movimiento.estado = 1
            @movimiento.update(estado: @movimiento.estado)
          end
        end
      end

      #cMonto = es el importe total adeudado
      #el mTotal = a la entrega parcial o total que realiza
      @total = @total.to_i + @movimiento.importe.to_i
      @matriculacion = Matriculacion.find(@cta_ct.matriculacion_id)
```

```

        factura = Factura.create!(usuario_id: current_usuario.id, alumno_id:
@matriculacion.alumno_id, total: mTotal)
        factura.update(nro_fac: factura.id)
        detallefactura = DetalleFactura.create!(factura_id: factura.id,
matriculacion_id: @matriculacion.id, nro_mov: @movimiento.nro_mov, descripcion:
@movimiento.descripcion, importe: mTotal)
        @cta_ct = CtaCte.find(@movimiento.cta_cte_id)
        cMonto = @cta_ct.montoimporte
        monto = cMonto.to_i - mTotal.to_i
        @cta_ct.update(montoimporte: monto)
        #crea movimiento de pago parcial
        @caja = Caja.where(estado: 0).last
        @mov_caja = MovCaja.create!(caja_id: @caja.id, concepto:
@movimiento.descripcion, ingreso: @movimiento.totalimporte.to_i, egreso: 0,
saldo: @caja.cierre.to_i + @movimiento.totalimporte.to_i)
        resultado = impto.to_i - mTotal.to_i
        @caja.update(cierre: @caja.cierre.to_i +
@movimiento.totalimporte.to_i, entrada: @caja.entrada.to_i +
@movimiento.totalimporte.to_i)
        format.html { redirect_to factura, notice: 'Se registro el pago
correctamente' }
        format.json { render :show, status: :ok, location: factura }
    else
        format.html { render :edit }
        format.json { render json: @movimiento.errors, status:
:unprocessable_entity }
    end
  else
    format.html { redirect_to @matriculacion, notice: 'Ya no se puede
pagar.' }
    format.json { render :show, status: :ok, location: @matriculacion }
  end
end
end
end

```

## Anexo 13: Código del Proyecto – Movimientos 2

Este fragmento de código se ejecuta al momento de elegir varias cuotas y a la vez se puede elegir un producto cualquiera. El cual lo suma a lo seleccionado con las cuotas y demás, generando al final una vista preliminar de la factura. Se utiliza para la vista de matriculación.

```
def pago_conjunto
```

```

    cta = CtaCte.new
    id = 0
    @total = 0
    @matriculacion = nil
    params[:movimientos_id].each do |id|
        movimiento = Movimiento.find(id)
        cta_cte = CtaCte.find(movimiento.cta_cte_id)
        id = cta_cte.id
        #montoimporte = es el importe total adeudado
        #importe = al pago total de la deuda LA ENTREGA
        saldo = cta_cte.montoimporte.to_i - movimiento.importe.to_i
        cta_cte.update(montoimporte: saldo.to_i)
        #crear movimiento
        @caja = Caja.where(estado: 0).last
        @mov_caja = MovCaja.create!(caja_id: @caja.id, concepto:
movimiento.descripcion, ingreso: movimiento.importe.to_i, egreso: 0,
saldo: @caja.cierre.to_i + movimiento.importe.to_i)
        @total = @total.to_i + movimiento.importe.to_i
        @caja.update(cierre: @caja.cierre.to_i + movimiento.importe.to_i,
entrada: @caja.entrada.to_i + movimiento.importe.to_i)

        #movimiento.update(importe: movimiento.importe.to_i -
movimiento.importe.to_i, estado: true, totalimporte: importe)
        @matriculacion = Matriculacion.find(cta_cte.matriculacion_id)
    end
    matriculacion = nil
    factura = nil
    #CREA FACTURA
    factura = Factura.create!(usuario_id: current_usuario.id, alumno_id:
@matriculacion.alumno_id, total: @total)
    factura.update(nro_fac: factura.id)
    respond_to do |format|
        params[:movimientos_id].each do |id|
            movimiento = Movimiento.find(id)
            cta_cte = CtaCte.find(movimiento.cta_cte_id)
            matriculacion = Matriculacion.find(cta_cte.matriculacion_id)
            detallefactura = DetalleFactura.create!(factura_id: factura.id,
matriculacion_id: matriculacion.id, nro_mov: movimiento.nro_mov,
descripcion: movimiento.descripcion, importe: movimiento.importe.to_i)

            movimiento.update(importe: 0, estado: true, totalimporte:
movimiento.importe.to_i)

```

```

        format.html { redirect_to factura, notice: 'Movimiento was
successfully destroyed.' }
        format.json { render :show, status: :ok, location: factura }
      end
    end
    #seleccion de producto
    if params[:productos_id] != nil
      params[:productos_id].each do |pid|
        producto = Producto.find(pid)
        cant = params[pid]
        resultado = producto.cantidad.to_i - cant.to_i
        producto.update(cantidad: resultado)
        @caja = Caja.where(estado: 0).last
        @mov_caja = MovCaja.create!(caja_id: @caja.id, concepto: "Venta
de "+producto.nombreproduct, ingreso: cant.to_i * producto.precio.to_i,
egreso: 0, saldo: @caja.cierre.to_i + (cant.to_i * producto.precio.to_i))
        @caja.update(cierre: @caja.cierre.to_i + (cant.to_i *
producto.precio.to_i), entrada: @caja.entrada.to_i + (cant.to_i *
producto.precio.to_i))
        detallefactura = DetalleFactura.create!(factura_id: factura.id,
matriculacion_id: matriculacion.id, nro_mov: nil, descripcion: "Pago de:
"+producto.nombreproduct, importe: producto.precio.to_i*cant.to_i)
        factura.update(total:
factura.total.to_i+detallefactura.importe.to_i)
      end
    end
  end
end

```

## Anexo 14: Código del Proyecto – Movimientos 3

Este fragmento de código se ejecuta cuando se utiliza el buscador por padres. Donde se puede hacer la selección múltiple con más de un alumno el cual también muestra una vista preliminar de la factura con los detalles.

```

def pago_masivo

  cta = CtaCte.new
  id = 0
  @total = 0
  @matriculacion = nil
  params[:movimientos_id].each do |id|
    movimiento = Movimiento.find(id)
    cta_cte = CtaCte.find(movimiento.cta_cte_id)
  end
end

```



```

      id = cta_cte.id
      saldo = cta_cte.montoimporte.to_i - movimiento.importe.to_i
      cta_cte.update(montoimporte: saldo.to_i)
      @caja = Caja.where(estado: 0).last
      @mov_caja = MovCaja.create!(caja_id: @caja.id, concepto:
movimiento.descripcion, ingreso: movimiento.importe.to_i, egreso: 0,
saldo: @caja.cierre.to_i + movimiento.importe.to_i)
      @total = @total.to_i + movimiento.importe.to_i
      @caja.update(cierre: @caja.cierre.to_i + movimiento.importe.to_i,
entrada: @caja.entrada.to_i + movimiento.importe.to_i)

      @matriculacion = Matriculacion.find(cta_cte.matriculacion_id)
    end
    #crea la factura
    factura = Factura.create!(usuario_id: current_usuario.id, alumno_id:
@matriculacion.alumno_id, total: @total)
    factura.update(nro_fac: factura.id)
    respond_to do |format|
      params[:movimientos_id].each do |id|
        movimiento = Movimiento.find(id)
        cta_cte = CtaCte.find(movimiento.cta_cte_id)
        matriculacion = Matriculacion.find(cta_cte.matriculacion_id)
        detallefactura = DetalleFactura.create!(factura_id: factura.id,
matriculacion_id: matriculacion.id, nro_mov: movimiento.nro_mov,
descripcion: movimiento.descripcion, importe: movimiento.importe)

        movimiento.update(importe: 0, estado: true)
        format.html { redirect_to factura, notice: 'Movimiento was
successfully destroyed.' }
        format.json { render :show, status: :ok, location: factura }
      end
    end
  end
end
end

```

## Anexo 15: Código del Proyecto – Movimiento caja

Crea el movimiento de caja.

```

def create

  @mov_caja = MovCaja.new(mov_caja_params)
  respond_to do |format|
    if @mov_caja.save
      format.html { redirect_to @mov_caja, notice: 'Mov caja was successfully
created.' }
    end
  end
end

```

```

        format.json { render :show, status: :created, location: @mov_caja }
      else
        format.html { render :new }
        format.json { render json: @mov_caja.errors, status:
:unprocessable_entity }
      end
    end
  end
end

```

## Anexo 16: Código del Proyecto – Registro de pagos de servicios.

La función de este fragmento de código registra el pago de algún servicio y genera el movimiento correspondiente.

```

def create
  @pagos_servicio = PagosServicio.new(pagos_servicio_params)
  respond_to do |format|
    if @pagos_servicio.save
      @caja = Caja.where(estado: 0).last
      @mov_caja = MovCaja.create!(caja_id: @caja.id, concepto:
@pagos_servicio.servicio, ingreso: 0, egreso: @pagos_servicio.monto, saldo:
@caja.cierre.to_i - @pagos_servicio.monto.to_i)
      @caja.update(cierre: @caja.cierre.to_i - @pagos_servicio.monto.to_i,
salida: @caja.salida.to_i + @pagos_servicio.monto.to_i)
      format.html { redirect_to @pagos_servicio, notice: 'Se ha registrado
el pago correctamente.' }
      format.json { render :show, status: :created, location:
@pagos_servicio }
    else
      format.html { render :new }
      format.json { render json: @pagos_servicio.errors, status:
:unprocessable_entity }
    end
  end
end

```

## Anexo 17: Código del Proyecto – Registro producto

Se encarga de registrar el producto con sus detalles, descripción, cantidad y precio.

El cual se puede editar para actualizar el stock.

```
def create

  @producto = Producto.new(producto_params)
  respond_to do |format|
    if @producto.save
      format.html { redirect_to @producto, notice: 'Se creo correctamente el
producto.' }
      format.json { render :show, status: :created, location: @producto }
    else
      format.html { render :new }
      format.json { render json: @producto.errors, status:
:unprocessable_entity }
    end
  end
end
```

## Anexo 18: Código del Proyecto – Cierre de caja

En este fragmento de código una vez cerrada la caja lo que realiza es el registro de la apertura y cierre de caja.

```
def create

  @cierre_caja = CierreCaja.new(cierre_caja_params)
  @cierre_caja.fApertura = @caja.fechaApertura
  @cierre_caja.cierre = @caja.apertura
  respond_to do |format|
    if @cierre_caja.save
      format.html { redirect_to @cierre_caja, notice: 'Cierre caja was
successfully created.' }
      format.json { render :show, status: :created, location: @cierre_caja }
    else
      format.html { render :new }
      format.json { render json: @cierre_caja.errors, status:
:unprocessable_entity }
    end
  end
end
```

## Anexo 19: Código del Proyecto – Factura

En este se indican que una vez realizado los pagos se deberá mostrar la vista de la factura.

```
def show

  respond_to do |format|
    format.html
```

```

      format.pdf do
        pdf = ReportPdf.new(@factura)
        send_data pdf.render, filename: 'show.pdf', type: 'application/pdf'
      end
    end
  end

  # GET /facturas/new
  def new
    @factura = Factura.new
  end

  # GET /facturas/1/edit
  def edit
  end

  # POST /facturas
  # POST /facturas.json
  def create
    @factura = Factura.new(factura_params)
    respond_to do |format|
      if @factura.save
        format.html { redirect_to @factura, notice: 'Se creo la factura
correctamente.' }
        format.json { render :show, status: :created, location: @factura }
      else
        format.html { render :new }
        format.json { render json: @factura.errors, status: :unprocessable_entity
      }
    end
  end
end
end
end

```

## Anexo 20: Código del Proyecto – Impresión de factura en pdf

Para la impresión de la factura se utilizó la gema Praw que permite diseñar el modelo de factura.

```

class ReportPdf <
  Prawn::Document

    def initialize(factura)
      super()
      @factura = factura
      header
      text_content
      table_content
    end
  end
end

```

```

        table_content2
    end
    def header
    end
    def text_content
        bounding_box([0, 600], :width => 1200, :height => 100) do
            text "Fecha: #{@factura.created_at.strftime('%m/%d/%Y')}", size:
10, style: :bold
        end
        bounding_box([0, 580], :width => 300, :height => 300) do
            text "Nombre: #{@factura.alumno.nombre}
#{@factura.alumno.apellido}", size: 10, style: :bold
        end

        move_down -420
        bounding_box([0, cursor], :width => 290, :height => 90) do
            transparent(0.5) { stroke_bounds }
            move_down 10
            indent(20) do
                text "Colegio Parroquial Privado San Pio X", size: 20, style:
:bold
            end
            indent(30) do
                text "Padre Jose Kreusser N°306 e/Juan Leon Mallorquin",
size: 10
                text "Telef.:(071)203778 Encarnacion - Paraguay", size: 10
                # stroke_horizontal_rule
            end
        end

        move_down -300
        bounding_box([300, 700], :width => 240, :height => 90) do
            transparent(0.5) { stroke_bounds }
            move_down 10
            indent(20) do
                text "Timbrado N° XXXXXXXX", size: 10, style: :bold
                text "RUC: 80056885-0", size: 15, style: :bold
                text "Inicio Vigencia: 05/setiembre/2016", size: 8
                text "Fin de Vigencia: 30/setiembre/2017", size: 8
                text "FACTURA N°", size: 15, style: :bold
                text "#{@factura.nro_fac}", size: 10
            end
        end
    end
end

```

```

end
def table_content
  table detalles_rows do
    row(0).font_style = :bold
    self.header = true
    self.row_colors = ['DDDDDD', 'FFFFFF']
    self.column_widths = [40, 300, 200]
  end
end

def detalles_rows
  move_down 45
  [['Nº', 'Descripcion', 'Importe']] +
    @factura.detalle_facturas.map do |detalle|
      [detalle.nro_mov, detalle.descripcion, detalle.importe]
    end
end

def table_content2
  table detalles_rows2 do
    row(0).font_style = :bold
    self.header = true
    self.row_colors = ['DDDDDD', 'FFFFFF']
    self.column_widths = [340, 200]
  end
end

def detalles_rows2
  move_down 40
  [['TOTAL ', @factura.total]]
end
end

```

## Anexo 21: Código del Proyecto – Routes

```

Rails.application.routes.draw
do

```

```

  resources :detalle_facturas
  resources :facturas
  resources :pagos_servicios
  resources :cajas
  resources :cierre_cajas

```

```
resources :mov_cajas
resources :productos

resources :movimientos do
  collection do
    put :pago_conjunto
    put :pago_masivo
  end
end

resources :cta_ctes
resources :matriculacions do
  collection do
    get :edit_multiple
    put :update_multiple
  end
end

resources :cursos do
  resources :detalle_cursos
end

devise_for :usuarios, controllers: { sessions:
"usuarios/sessions", registrations: "usuarios/registrations",
passwords: "usuarios/passwords" }, :path_names => {:sign_in
=> 'login', :sign_up => 'registro', :sign_out => 'logout'}
  as :usuario do
    get 'sign_in' => 'usuarios/sessions#new', :as =>
:new_usuario_session_path
    get 'sign_up' => 'usuarios/registrations#create', :as =>
:usuario_registration_path
    delete 'sign_out' => 'usuarios/sessions#destroy', :as =>
:destroy_usuario_session_path
    get 'new' => 'usuarios/sessions#destroy', :as =>
:new_usuario_password_path
  end
resources :alumnos do
  get :autocomplete_alumno_ci, :on => :collection
  collection do
    get "buscar"
    post "busqueda"
  end
end

cajaabierto = nil
cajaabierto = Caja.where(estado: 0).last
# The priority is based upon order of creation: first
```

```
created -> highest priority.  
# See how all your routes lay out with "rake routes".  
# You can have the root of your site routed with "root"  
if cajaabierto != nil  
  root 'alumnos#index'  
else  
  root 'cajas#new'  
End  
end
```