UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

Título: TIC en la gestión integral de escuelas de enseñanza pre-universitaria
Autores: Corrons María Emilia
Director: Molinari Lía Hebe
Codirector: Pérez Juan Pablo
Carrera: Licenciatura en Sistemas

Resumen

Las Tecnologías de la Información y la Comunicación constituyen un instrumento protagónico clave para la producción e intercambio de información. Es gracias a ellas que día a día se logra mayor fluidez, transmisión y crecimiento del conocimiento. La aplicación de TIC en el ámbito de gestión y administración de información en escuelas de nivel pre-universitario plantea un panorama favorecedor. Teniendo esto en cuenta y contando con el sistema de gestión integral de escuelas Kimkëlen, se estudiarán los beneficios de desarrollar una puerta virtual a las instituciones educativas disponible no solo para las autoridades administrativas y docentes, sino también para los estudiantes y su círculo familiar. El objetivo del presente trabajo es, entonces, realizar una investigación de los aspectos inherentes a la informatización total de la gestión y comunicación en escuelas de nivel pre-universitario y de los beneficios que esto conllevaría, para luego, a partir de dicha investigación, proponer un modelo general, implementar una API REST de comunicación y los prototipos funcionales de las aplicaciones *web* que ayudarán a llevar a cabo dicha informatización.

Palabras Claves

TIC. Aplicaciones *web*. API RESTful. Informatización de Organizaciones. Sistemas académicos. Gestión de escuelas pre-universitarias. Sistema de gestión de escuelas. Comunicación. Kimkëlen. *Software* libre.

Conclusiones

La informatización completa del circuito de comunicación dentro de las escuelas traerá consigo diversos cambios no solo en la gestión académica sino también en la percepción que todos los usuarios activos de la comunidad educativa -autoridades, docentes, alumnos y tutores- poseen sobre la importancia del dato certero, la detección temprana y en tiempo real de situaciones e incidentes que suceden dentro del recinto escolar y la concepción de una base sólida de información histórica.

Trabajos Realizados

- Investigación y trabajo de campo sobre el estado de implantación de TIC en escuelas de enseñanza pre-universitaria y especificación de los beneficios que traería la informatización completa del circuito de comunicación de las mismas.
- Desarrollo de una API RESTful del sistema Kimkëlen.
- Desarrollo, en forma parcial, de aplicación para acceso de estudiantes.
- Desarrollo, en forma parcial, de aplicación para acceso de tutores.

Trabajos Futuros

- Incorporar autenticación a la API RESTful.
- Implementación completa y extensión de aplicaciones *frontend* considerando:
  - Aprovechar servicios de terceros, esto es, proveer *login* a través de redes sociales.
  - Agregar bandeja de notificaciones en ambas aplicaciones, de manera que tutores y estudiantes puedan recibir notificaciones enviadas por las autoridades del colegio.
- Realizar pruebas de aceptación y usabilidad de las aplicaciones *frontend* con el usuario final.

Fecha de la presentación: Octubre 2015

Indice

Agradecimientos

Dedico esta tesis a mi mamá y a mi papá, por darme la oportunidad de elegir una carrera y estudiar.

A mis cuatro abuelos; Aida y Juan, quienes me acompañan siempre. A mi abuelo Oscar y a mi abuela Hulda, que la vida no me dejó conocer pero siempre admiré mucho.

A Cati, por todo el amor que me brindó en interminables horas de estudio.

A mi hermana y a mis amigas Ayelén, Julieta, Paula, Cecilia, Daniela, Beatriz, Eva, Mariana y Melina, por sus palabras de aliento y apoyo durante toda mi carrera.

A Magalí, gran compañera y amiga desde el inicio de mi etapa universitaria, sin quien no hubiera llegado hasta aquí.

A Christian, por ayudarme muchísimo en mi crecimiento profesional.

A Juan Pablo y a Lía, por dirigirme en esta tesis y confiar en mí.

A mis actuales y pasados compañeros del CeSPI, por enseñarme siempre algo nuevo y hacer muy ameno el día a día.

A Ana, por las mañanas y tardes desarrollando juntas.

Y a mis compañeros y compañeras de facultad, por los buenos recuerdos que atesoro de una etapa hermosa que con esta tesina llega a su fin.

¡Gracias!

1. Introducción

En este capítulo se hará una introducción acorde a los temas que se abordarán en esta tesis, comenzando por la motivación y las situaciones problemáticas observadas, siguiendo por los objetivos y concluyendo con una breve descripción de cada uno de los subsiguientes capítulos que conforman este trabajo.

1.1. Motivación

En la sociedad actual, las TIC constituyen un instrumento clave para la comunicación, intercambio y producción de información. Como miembros de la sociedad de la información deseamos y podemos obtener y compartir cualquier información instantáneamente. El conocimiento se convierte en combustible y la tecnología de la informática y la comunicación en el motor. (Giner de la Fuente y Gil Estallo, 2004).

La presencia y hegemonía de sistemas informáticos para simplificar y mejorar los procesos que comprenden la administración y dirección de las Instituciones Educativas en Argentina se dibuja como un hecho tan exitoso como beneficioso.

Como en toda Organización, el uso de un sistema de información aporta a la sistematización de procesos, reduce gastos de insumos y mejora la gestión del espacio físico, reduciendo el almacenamiento e impresión de papeles y evitando archivos muertos difíciles de acceder y consultar. No obstante y sin restarle importancia, informatizar el entorno de gestión escolar también brindará grandes aportes a la integración de espacios. Es posible analizar desde distintas perspectivas, según el involucrado, cómo se plantea el circuito de comunicación y flujo de información dentro y hacia fuera de las instituciones educativas.

Imaginemos que una Institución educativa de nivel pre-universitario cuenta con el *software* de gestión integral **Kimkëlen** (CeSPI, 2014). Gracias a éste, tanto autoridades del colegio como administrativos, preceptores y docentes podrán realizar sus tareas de

gestión en forma automatizada, organizada y eficiente. El hilo conductor de esta tesina plantea que sería enriquecedor si los alumnos y sus tutores (familiares responsables), de acuerdo con la naturaleza de su relación con la institución, también pudieran involucrarse con el sistema como fuente de información y comunicación; Kimkëlen tomando el rol de puerta virtual hacia la casa de estudios. De esta forma se habrá logrado la informatización completa del circuito administrativo y de comunicación escolar.

A través de una aplicación de fácil acceso y uso que representará el *frontend* - la parte del *software* que interactúa con el o los usuarios - de Kimkëlen, los estudiantes serán capaces de enterarse en tiempo real cuando los docentes carguen sus calificaciones, y podrán consultarlas desde una computadora o desde cualquier dispositivo con acceso a Internet. Serán capaces de consultar sus inasistencias a clase, sus sanciones disciplinarias, mapa de aulas y horarios, recibir notificaciones, ver su historial de cursadas en la institución, boletines anuales y demás información relativa a su progreso académico. Imaginemos que la aplicación proveerá, además, integración con entornos educativos virtuales. De esta manera, si un curso posee una instancia en una plataforma educativa como lo es, por ejemplo, *Moodle* (Dougiamas, 2014), el alumno podría descargar material necesario para la cursada o cualquier recurso que el docente allí administre.

Sin ir más lejos, al igual que los alumnos los familiares responsables de los mismos, es decir, sus tutores, también contarán con un *frontend web* a través del cual podrán consultar en tiempo real las inasistencias y retardos de sus hijos, calificaciones, apercibimientos y sanciones disciplinarias. Podrán ser notificados acerca de reuniones, actos escolares, excursiones o citaciones por parte de los docentes y autoridades de la institución. Este acceso podrá efectuarse a través de dispositivos móviles, como celulares o tabletas.

Además, sacando provecho de la oferta de servicios de Internet disponibles que en los últimos tiempos ha crecido enormemente, y dado que la mayoría de los usuarios utilizan múltiples aplicaciones de terceros -tales como *webmail* o redes sociales-, tanto

el acceso a la aplicación de estudiantes como el acceso a la aplicación de tutores podrá realizarse por medio de un protocolo de autenticación contra redes sociales.

1.2. Situación problemática

Las problemáticas observadas son varias. En principio, la falta de un espacio institucional con capacidades efectivas para generar y agilizar el diálogo y la interacción tanto entre la institución educativa y los estudiantes como entre la institución educativa y los tutores de éstos es una de ellas.

Tanto el alumno como sus tutores no cuentan con medios para obtener la traza de su progreso académico desde el ingreso hasta el egreso del organismo educativo.

El uso de herramientas en formato papel tales como cuadernos de comunicaciones, notas y demás documentos impresos:

- no permite disponer de datos de importancia dentro del ámbito administrativo y comunicativo escolar en tiempo real.
- no optimizan la utilización de recursos e insumos.
- no aporta a la automatización completa de tareas.
- causa repetición de la misma información.

Las TIC no sólo han revolucionado la difusión de información, sino que también nos han expuesto a nuevas ideas y formas de pensamiento. Acoplado al cambio de las estructuras sociales, el modo de pensar de una nueva generación hace que sea importante para todas las partes interesadas tener una relación más interactiva con la escuela.

1.3. Objetivos

Los objetivos de este trabajo de tesis son estudiar, analizar, diagnosticar los desafíos y sugerir estrategias de mejoras para superar los problemas que prevalecen en el área

de gestión y comunicación en instituciones educativas. Tras la realización de un estudio minucioso se defenderá la idea de que la mejor vía para crear, mejorar, ampliar e integrar espacios de comunicación inherentes al campo de acción de estas instituciones es a través de su informatización.

Los aportes que se realizarán comprenden:

- Generar una idea compuesta del estado de implantación de TIC en el marco de administración de Instituciones educativas de enseñanza pre-universitaria.
- Contribuir con la extensión del sistema **Kimkëlen**, ERP de Gestión Integrada de Escuelas *open source*, liberado por el CeSPI en el año 2013 (Desarrollo-CeSPI, 2014). La autora de esta tesis propuso e implementó esta extensión bajo la supervisión de sus directores y el consentimiento del CeSPI. Uno de los eslabones faltantes de Kimkëlen y que se va a gestar e implementar parcialmente en la presente tesis es la extensión del mismo con el desarrollo de las aplicaciones *frontend*, que serán accesibles por alumnos y sus tutores. Esto supone, además, la adaptación y mejoras necesarias de la aplicación *backend* y la implementación de una API REST, que alimentará a las aplicaciones *frontend* a través de servicios REST.
- Fomentar el uso de herramientas informáticas adecuadas.
- Promover el buen uso de la red tecnológica para el aprovechamiento de los recursos y de sus servicios.
- Detección temprana por parte de alumnos y padres de situaciones e incidentes que suceden dentro del recinto escolar.

1.4. Estructura

En el **capítulo 2** se plantea un recorrido detallado sobre los problemas planteados en 1.2.

En el **capítulo 3** se introducirá el sistema Kimkëlen, que representa el sistema base que aquí se plantea extender. Además, se presentarán otras opciones similares a éste que se pueden hallar hoy en día en la comunidad de *software* libre y en el mercado.

El **capítulo 4** se dará un panorama de la situación actual de implantación de TICs en la sociedad y en particular en el sector de gestión administrativa y comunicación escolar. Se hará una reseña histórica de la evolución de las aplicaciones *web.* Se introducirán conceptos sobre arquitecturas de desarrollo de sistemas de *software*. Sobre el final del capítulo se hará la descripción de la extensión de Kimkëlen propuesta para dar un cierre al marco teórico de la tesina.

El **capítulo 5** comprende el marco práctico del desarrollo planteado en el capítulo 4. Se presentarán los resultados obtenidos de las encuestas que se han realizado a usuarios actuales del sistema Kimkëlen. Se presentará en detalle el modelo y desarrollo de las aplicaciones propuestas y la conexión necesaria que éstas deberán tener para funcionar.

En el **capítulo 6** se enuncian las conclusiones obtenidas a partir del trabajo realizado y se hará un repaso sobre los aportes de la tesina. Además, se citan los trabajos futuros que se podrían realizar siguiendo la línea de trabajo expuesta.

2. Problemática

En el ámbito administrativo escolar, la prevaleciente existencia de herramientas no digitales tales como cuadernos de comunicaciones, boletines de calificaciones y demás documentos no informatizados arrastra consigo diversas situaciones que pueden y deberían mejorarse, y que exigen una adaptación acorde a la evolución tecnológica. Aquello que antes nos era desconocido y de difícil alcance, se ha transformado en una herramienta indispensable para la época que vivimos.

Se han encontrado diversos problemas que prevalecen en el circuito de gestión escolar, los cuales se enumeran a continuación.

2.1. Misma información repetida y físicamente separada

La duplicación de información es un problema muy común. Se tiene la misma información en distintos documentos impresos ya sean planillas, archivos, etcétera. Esta información no está unificada ni normalizada y en muchos casos está desactualizada. Por lo tanto, deja de ser confiable y válida y pierde así su valor característico.

Nada quita que en un sistema de gestión informatizado no pueda generarse redundancia de datos -que se entiende como el almacenamiento de mismos datos varias veces en diferentes lugares-. La redundancia de datos acarrea problemas tales como:

- ➔ incremento de trabajo.
- ➔ desperdicio de espacio de almacenamiento.
- ➔ inconsistencia de datos.

En definitiva, se presentan los mismos problemas que si no se posee un sistema informático. Pero la ventaja radica en que si una base de datos está bien diseñada no

debería haber redundancia de datos (exceptuando la redundancia de datos controlada que se emplea para mejorar el rendimiento de las consultas).

2.2. Disponibilidad insuficiente de datos en tiempo real

El no poseer informatizado el circuito completo frena la evolución de las Instituciones educativas hacia un nuevo modelo que persiga los fundamentos adoptados por la sociedad y las organizaciones modernas: poder obtener y compartir cualquier información instantáneamente, de forma confiable y en tiempo real.
De esta forma, informatizar el circuito completo de comunicación logrará la detección temprana por parte de alumnos y padres de situaciones que suceden dentro de la institución y el acceso inmediato a la información disponible.

2.3. Utilización de recursos no optimizada, información en condición de riesgo y existencia de tareas que podrían ser automatizadas

Dentro del ámbito administrativo escolar se distinguen muchas tareas que son aún realizadas por personas, aparejando con ello el error humano que es un factor importante a considerar y que puede ser erradicado si se automatizan dichas tareas por completo. Un ejemplo de este tipo de tareas sería el cálculo de la sumatoria total de inasistencias que registra un estudiante en un período dado.
Por otro lado, lograr la digitalización de los documentos a través del almacenamiento digital de los mismos permitirá proteger la información, brindar una fácil distribución y difusión de la misma, optimizar el espacio físico, convirtiendo montones de papel en información de acceso inmediato (Readsoft, 2014).
De esta forma, se verán eliminados problemas tales como:

- Extravío de datos: Se mitiga el riesgo de la pérdida de información y datos ocasionada por incendios, robos, inundaciones y otras catástrofes impredecibles.
- Mutabilidad y falsificación de la información: Se conservan los documentos intactos a lo largo del transcurso del tiempo. No se deterioran ni pueden sufrir adulteración dolosa sin dejar rastro de ello.
- Acceso a la información: Al informatizar se permite el acceso simultáneo de varios usuarios a la misma información.

Consecuentemente, algo destacado de este proceso de digitalización es la contribución al cuidado del medio ambiente. El ahorro en gastos en papel y tinta de impresión que se puede lograr no es en absoluto despreciable.

2.4. Comunicación no optimizada e informal

Actualmente no se crea un registro claro de la comunicación de la escuela para con los padres de los estudiantes. La comunicación entre ellos es indirecta, siempre a través a algún medio no informatizado tan fácil de extraviar como de adulterar.

A su vez, Rodríguez, Pérez y Corrons (2014) señalan que “la comunicación estudiante-escuela se reduce a la cantidad de horas que el estudiante pasa dentro del recinto escolar. No existe una vía eficiente para poder comunicarse con los estudiantes fuera del horario escolar, ya fuera por situaciones extraordinarias o no, pero sí necesarias por parte de la Institución educativa.”

Como alternativa y posible solución, sabemos que el uso de las tecnologías integra en nuevas prácticas sociales a la cultura, la comunicación y el conocimiento. Desde la perspectiva de espacio de comunicación, el ciberespacio suprime las limitaciones contextuales de carácter espacial y temporal para la comunicación, generando una nueva concepción temporal, el “tiempo atemporal” (Castells, 2000: 507), y otra espacial, el “espacio global”. Esto permite establecer canales de comunicación, tanto

sincrónicos como asincrónicos, que amplían las posibilidades de comunicación y en consecuencia las interacciones sociales. La comunicación virtual permite que no sea preciso compartir espacio ni tiempo, amplía el espectro, trasciende.

Cabe aclarar que no se está planteando que la comunicación virtual y la comunicación personal/real sean mutuamente excluyentes. Como dice Buch (2013), los maravillosos avances en las comunicaciones pueden transformarse en una verdadera adicción e impedir el pensamiento autónomo, además de alejar a la gente de un contacto real haciéndole creer que está conectada.

La comunicación virtual no va a reemplazar a la comunicación personal, sino que la idea es ampliar el espectro de comunicación, que ambas formas se sustenten una sobre la otra.

Si bien es cierto que muchas escuelas y docentes utilizan hoy en día algún medio tecnológico para comunicarse con los estudiantes fuera del horario escolar -como pueden ser los grupos de *Facebook* (Facebook, 2014)-, estos medios no representan una vía eficiente y formal de comunicación. No pertenecen al ámbito académico.

2.5. No aprovechamiento de la posible integración con diversas tecnologías *web* disponibles hoy en día.

La oferta de tecnologías web es enorme y las personas se animan día a día a involucrarse con éstas un poco más.

En la mayoría de los sitios web es requerido acceder a recursos utilizando credenciales, y esto termina ocasionando que los usuarios tengan distintos nombres de usuario y contraseñas que recordar.

La tendencia que está ganando terreno es ingresar o acceder a los sistemas *web* por medio de aplicaciones de terceros, como *Google connect* o redes sociales. Los usuarios de Internet están cada vez más acostumbrados a utilizar una única cuenta para identificarse en todas las páginas que visitan. Integrando el registro y la validación

de usuarios de una aplicación *web* propia con una cuenta de usuario popular como *Facebook* (Facebook, 2014) o *Twitter* (Twitter, 2014) se consigue facilitar enormemente el acceso y registro de nuevos usuarios a nuestro sistema *web* y facilita al usuario enormemente, ya que no le es necesario recordar muchas contraseñas de acceso o estar ingresando información repetida en formularios de creación de usuarios.

3. Background y estado del arte

Dentro del campo de los sistemas de gestión de escuelas o estudiantes -SIS: Student Information Systems- (Joshi, 2013) existen varias opciones disponibles que permiten realizar la gestión integral de un establecimiento de enseñanza media, superior o preuniversitaria. A continuación se describe en profundidad el *software* Kimkëlen y otras opciones similares a éste.

3.1. Kimkëlen

Kimkëlen -palabra que proviene de la lengua autóctona Mapuche cuyo significado es "conocimiento"- es un sistema de gestión integrada de escuelas, que permite administrar y centralizar toda la información y registros propios de una institución educativa secundaria. Funciona de manera *online* y permite el acceso y la consulta inmediata por parte de quienes que tengan un usuario de acceso al *backend*.
Rodríguez *et al.* (2014) dijeron que el proyecto nace en el año 2008 como una solución viable tras el análisis de los problemas tradicionales que aquejan a colegios de gestión pre-universitaria, cuyo volumen de estudiantes y complejidad de gestión es grande.

Figura 1. Sistema de alumnos Kimkëlen

La versión del sistema que se liberó bajo licencia GNU GPLv2 es la 2.15.0. La última versión disponible al momento de realización de este trabajo es la 2.19.6. La liberación del código se hizo posible gracias a más de tres años de trabajo por parte de los desarrolladores del Ce.S.P.I. en conjunto con el apoyo de las escuelas pertenecientes a la UNLP, lo que garantiza un nivel de madurez importante del sistema en diversos aspectos tales como:

- diseño
- adaptabilidad
- usabilidad

Además de estar disponible para su libre utilización desde el año 2013, este sistema posee una comunidad de usuarios que dan cuenta de su estabilidad y posibilidad de adaptación (Desarrollo-CeSPI, 2014).

El proyecto posee amplia aceptación en los colegios secundarios pertenecientes al Sistema de Pregrado Universitario en los cuales se implementó bajo el soporte del Ce.S.P.I. Entre ellos están comprendidos, de la Universidad Nacional de La Plata:

- Liceo Víctor Mercante
- Colegio Nacional Rafael Hernández
- Bachillerato de Bellas Artes

y de la Universidad de Buenos Aires:

- Colegio Nacional de Buenos Aires
- Escuela de Educación Técnico Profesional en Producción Agropecuaria y Agroalimentaria
- Escuela Superior de Comercio Carlos Pellegrini

También está siendo adaptado para funcionar en una escuela de educación primaria exclusivamente perteneciente también a la Universidad Nacional de La Plata; la escuela Graduada Joaquín V. González.

Desde que su código fuente fue liberado a la comunidad, el sistema está siendo probado en muchas más instituciones educativas pertenecientes a otras universidades nacionales, tales como:

- Universidad Nacional de Cuyo
- Universidad Nacional de Lomas de Zamora
- Universidad Nacional del Centro de la Pcia. de Buenos Aires
- Universidad Nacional de Tucumán
- Universidad Nacional de San Luis
- Universidad Nacional de la Patagonia San Juan Bosco
- Universidad Nacional de Catamarca
- Universidad Nacional de Jujuy

como así también se han recibido consultas desde otros países como Chile y México.

3.1.1 Características

Los aspectos principales que permite manejar el sistema están comprendidos en los siguientes puntos.

3.1.1.1. Gestión de carreras, planes de estudio y años lectivos

Como punto de partida del sistema se permite la creación de:

- ➔ Años lectivos: Permite crear uno por año y solo el año actual debe estar marcado como vigente.
- ➔ Planes de estudio: Según las resoluciones de cada Colegio en particular, debe crearse el plan de estudio que comprenda la cantidad de años y materias que se cursan en una carrera.
- ➔ Carreras: Una escuela podría tener una o más carreras que pueden ser cursadas por los alumnos. Por ejemplo: Ciclo Básico, Bachillerato.

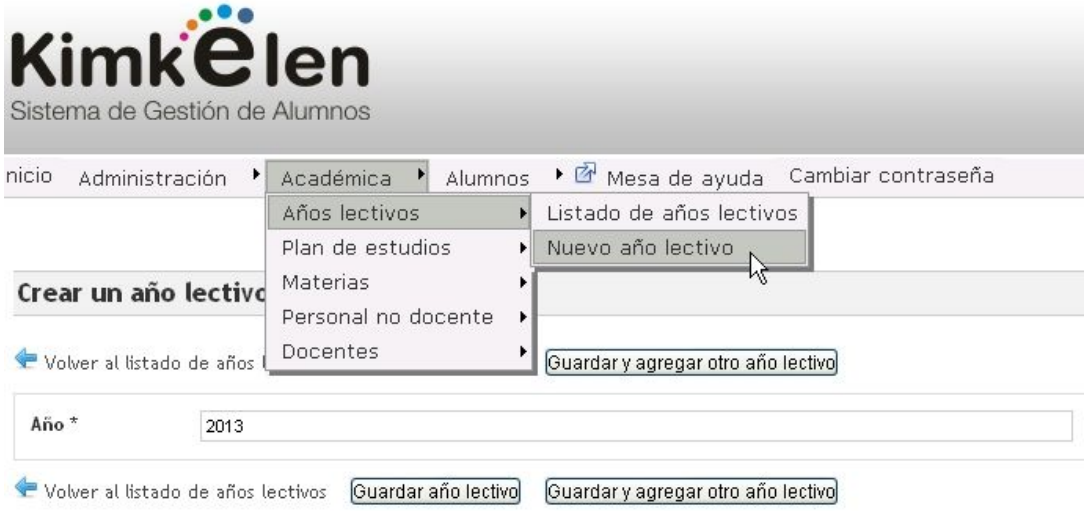

Figura 2. Menú Académica

3.1.1.2. Manejo de estudiantes

Conformando el corazón del sistema, se cuenta por supuesto con la creación de los registros de los estudiantes. Toda la traza de su historia académica dentro de la escuela y datos relacionados es registrada:

- ➔ Datos personales.

- ➔ Datos de sus tutores.
- ➔ Matriculación de los alumnos en un año lectivo y en una o más carreras.
- ➔ Registro de las inasistencias del alumno.
- ➔ Registro de los apercibimientos y sanciones disciplinarias.
- ➔ Impresión de boletines y documentos institucionales.

Imprimir
Volver

LV M

Alumno: Apellido2670, Nombre2670 Curso: 4 División: A Año lectivo: 2014

CALIFICACIONES

Áreas-Materias	1°T	2°T	3°T	Prom.	Ex.R.	Ex.C.	Ex.P.	Prom.Def.
Materia: 14	8.00	7.00	8.00	7.66				7.66
Materia: 15	8.00	7.00	7.00	7.33				7.33
Materia: 16	4.00	4.00	4.00	4.00	3.00	A	6.00	5.00
Materia: 17	4.00	7.00	10.00	7.00				7.00
Materia: 21	9.00	6.00	7.00	7.33				7.33
Materia: 22	9.00	9.00	7.00	8.33				8.33
Materia: 23	7.00	7.00	7.00	7.00				7.00
Materia: 25	7.00	7.00	7.00	7.00				7.00
Materia: 26	8.00	7.00	8.00	7.66				7.66
Materia: 27	8.00	7.00	8.00	7.66				7.66
Materia: 5	8.00	7.00	7.00	7.33				7.33

TRIMESTRALES

Intro. a la Problemática de:	1°T	Ex.R.	Ex.C.	Ex.P.	Inasistencias	Prom.Def.
Materia: 30	7.00				2	
Materia: 29	7.00				0	7
Materia: 28	7.00				2	

CALIFICACIONES

Áreas-Materias	1°T	2°T	3°T	Prom.	Ex.R.	Ex.C.	Ex.P.	Prom.Def.	Inasist. 1°T	Inasist. 2°T	Inasist. 3°T
Materia: 10	7.00	8.00	8.00	7.66				7.66	0	0	3

PROMEDIO: 7.41

Inasistencias			
1°T	2°T	3°T	Total
3.5	3	2	8.5

Comportamiento		
1°T	2°T	3°T
MB	MB	B

Sanciones		
1°T	2°T	3°T
0	0	2

Observaciones | Firma del/la Responsable | Firma de la Autoridad

Figura 3. Boletín

3.1.1.3. Divisiones y comisiones

Se permite la creación de unidades -las llamadas comisiones y divisiones- que agrupan lógicamente alumnos, materias, profesores y preceptores. De esta forma se permite la posibilidad de manejar en bloque a un conjunto de alumnos, tal como sucede en la realidad.

3.1.1.4. Gestión de usuarios profesores, preceptores, jefes de preceptores y autoridades

El sistema cuenta con los módulos CRUD de quienes representan el personal y autoridades del colegio.

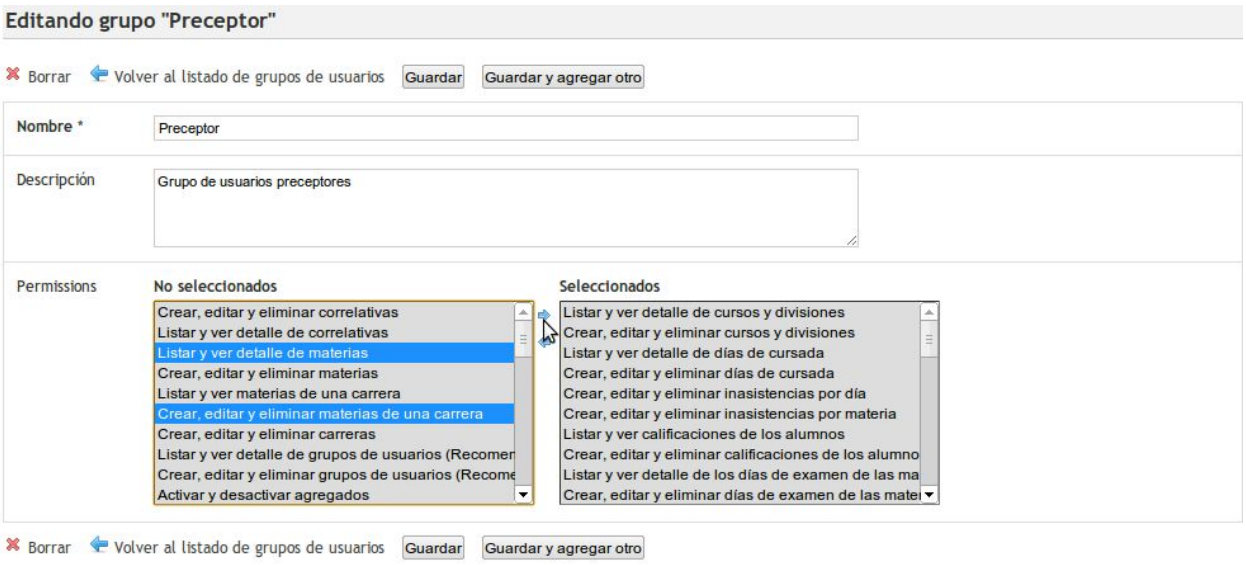

Figura 4. Manejo de grupos de usuarios y permisos

3.1.1.5. Manejo de mesas de examen

Se permite la creación de mesas de exámenes tanto manuales como automáticas (esto es, una vez creada la mesa los alumnos a rendir deberán ser colocados uno a uno en el caso de mesas manuales o serán afectados a las mesas automáticamente por el sistema en el caso de las mesas automáticas). Se manejan los conceptos de instancias (Regulares, Complementarias) y tipos de mesa (Libre, Previa).

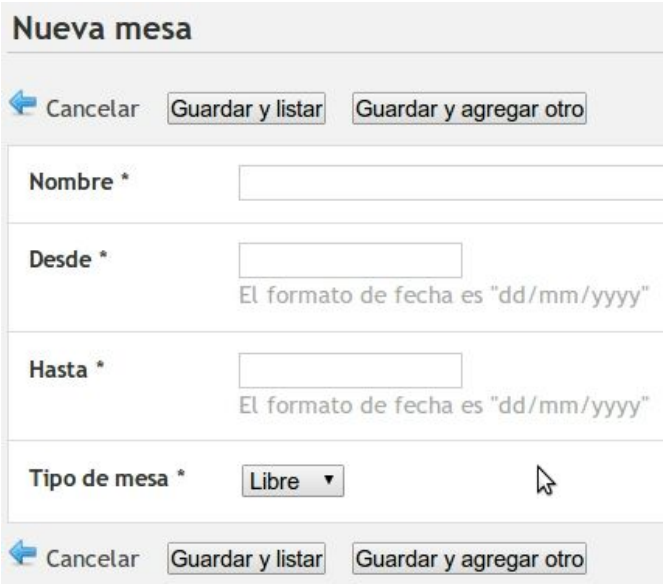

Figura 5. Creación de mesa de examen

3.1.1.6. Información de aulas y horarios

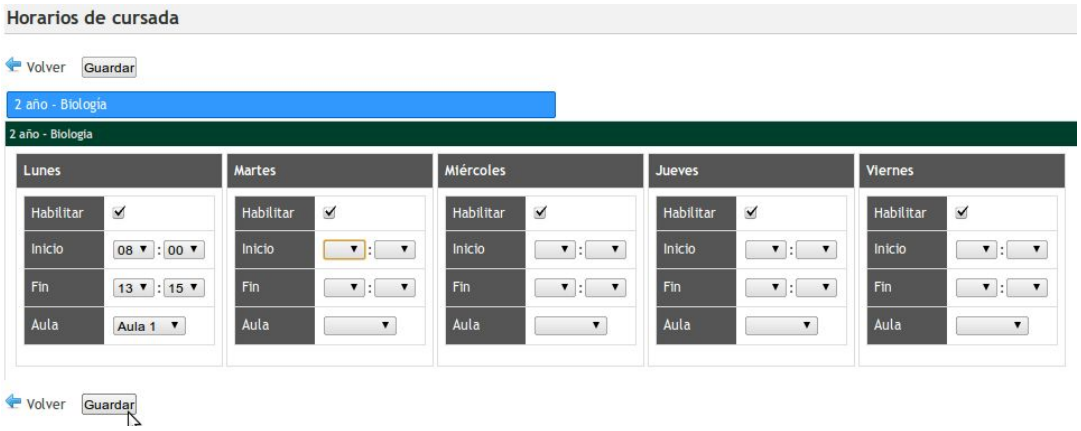

Figura 6. Administración de tabla horaria

3.1.1.7. Personalización

Como cada Institución educativa puede tener su propio esquema de enseñanza, Kimkëlen permite la personalización a través de la creación de *flavors* o sabores. Un sabor responde a los diversos modos de gestionar una institución educativa, y permite definir en forma desacoplada formas de evaluación, seguimiento de inasistencias, sanciones disciplinarias, impresión de boletines, generación de reportes, etcétera.
El sistema hace amplio uso de patrones de diseño como por ejemplo *template method* o método plantilla (Gamma, Helm, Johnson y Vlissides, 2003).

3.1.2. Tecnologías utilizadas y requerimientos mínimos

Kimkëlen está desarrollado bajo el *framework* Symfony 1.2. (Potencier, 2014), escrito en el lenguaje de programación PHP. La instalación, independientemente de la plataforma operativa del servidor (GNU Linux, Windows Server, etc.) requiere de la configuración de un servidor de aplicaciones *web*, como puede ser Apache Server (Apache Software Foundation, 2014), configurado correctamente para ejecutar el lenguaje de programación PHP en una versión de la rama mayor a la 5.2 y menor que PHP 5.4. También se requerirá de un servidor de Base de Datos como MySQL (Oracle, 2014).

3.1.3. Falencias y puntos a mejorar

Actualmente, además de *bugs* menores de funcionamiento, *performance* y carencia de un módulo estadístico para brindar información de carácter gerencial, el sistema no comprende un *frontend* para el acceso de alumnos de la escuela y familiares interesados en ver la evolución académica de sus estudiantes a cargo.
Tampoco permite integración con servicios de terceros o acceso desde tecnologías *mobil*e.

3.2. Fedena

El proyecto Fedena (Foradian Technologies, 2014) -cuyo nombre está inspirado en Atenea, diosa griega del conocimiento- es un sistema web ERP utilizado en escuelas con propósitos administrativos. Fue desarrollado por Foradian Technologies, compañía establecida en Bangalore, India. Comenzó a desarrollarse en Junio de 2009 y la primera versión del sistema fue lanzada en Agosto del 2009.

En la actualidad, Fedena es utilizado por el departamento de educación gubernamental de Kerala para automatizar el manejo de más de quince mil escuelas.

Existe una versión comercial del sistema llamada Fedena Pro, la cual permite incluir módulos y *plugins premium* que requieren abono aparte.

Figura 7

3.2.1. Características

3.2.1.1. Funcionalidades y módulos disponibles

Las funcionalidades y módulos que pueden incorporarse al sistema son los listados a continuación, perteneciendo algunos al *core* y siendo otros pagos.

- Tablas horarias
- Login de profesores
- Hostel/Dormitorios
- Integración con Moodle
- Manejo de eventos
- Login de padres/estudiantes
- Customización de interfaz
- Integración con datos biométricos
- Integración con Google Apps
- Manejo de cursos
- Información de los alumnos
- Manejo de noticias
- Exámenes online
- Admisión de estudiantes
- Login de empleados
- Pago de cuotas
- Sanciones disciplinarias
- Envío de alertas vía SMS
- Asistencia a clases
- Manejo de usuarios
- Exámenes
- Encuestas
- Videoconferencias
- Biblioteca
- Transporte
- Versión móvil
- Calendario escolar
- Blog
- Finanzas
- RRHH

3.2.2. Tecnologías utilizadas y requerimientos mínimos

Fedena fue desarrollado bajo el framework Ruby On Rails en su versión 2. Por ende para su instalación se necesitará tener instalado Ruby, Rails, MySQL y algunas gemas con las cuales tiene dependencia.

3.3. Mi-escuela

Mi-escuela (Technoware, 2014) es un sistema de administración y control escolar que se adapta a diversos tipos de planteles educativos (escuelas primarias, secundarias, bachilleratos, universidades, escuelas de inglés, etc.). Facilita a los usuarios llevar un mejor control escolar centrado en la gestión de calificaciones. Fue desarrollado en México.

Es *software* propietario, que maneja dos opciones de contratación: alquiler y compra.

El servicio de alquiler -*Software as a service* o SaaS- le brinda al cliente la posibilidad de utilizar el sistema conectándose a los servidores externos pagando una cuota mensual fija basada en el número de alumnos. Para la compra el costo del sistema se basa en el número de alumnos inscriptos, por lo que la licencia es por servidor. Será necesario comprar licencias adicionales si se incrementa el total de alumnos inscriptos.

3.3.1. Características

3.3.1.1. Manejo de alumnos

Este módulo actualiza los datos generales, calificaciones, asistencias de los alumnos además de generar consultas de manera sencilla y eficiente las cuales pueden ser exportadas a planillas Excel para graficar o darles una mejor presentación si así se lo requiere. Las opciones del módulo son las siguientes:

- Archivo: contempla el ABM de los datos generales del alumno.
- Expediente: permite consultar la información académica y administrativa del alumno como horarios, tutores, calificaciones, documentos, agrupaciones de alumnos, adeudos, faltas, observaciones, constancias de estudio, actividades,

revalidaciones de materias, impresión de estado de cuenta para el pago en banco, pagos en línea, etc.

- Consultas: Diseñadas para conocer de forma rápida y sencilla total de alumnos clasificados por: grupo, carrera, nivel, grado, etc.

3.3.1.2. Manejo de grupos

Con este módulo se podrán administrar las materias, horarios y alumnos asignados a un grupo, éste se divide en las siguientes opciones:

- Registro: dar de alta los grupos dentro del sistema.
- Alumnos: asigna alumnos inscriptos de forma aleatoria en los grupos creados o los asigna respetando su grupo anterior.
- Expediente: dentro de este se verifica toda la información relacionada a un grupo, se puede consultar lo siguiente: Horarios, Alumnos, Aulas, Materias, Calificaciones, generar listas generales de alumnos, asistencias, promedio general del grupo, promedio por Materia, mejores alumnos del grupo y alumnos con el promedio más bajo del grupo.
- Aprovechamiento: genera los promedios por evaluaciones parciales o mensuales y finales de todos los grupos a nivel general, por materia, por carrera y por alumno.
- Evaluaciones: se imprimen boletas de calificaciones en formato definido por la escuela clasificado en orden variables como puede ser: apellido paterno, nombre, domicilio, etc.

3.3.1.3. Manejo de materias

Administra las materias que se imparten en cada uno los diferentes niveles de educación de la escuela.

- Archivo: se agregan, se dan de baja o se actualizan los datos de las materias.

- Expediente: dentro del expediente de la materia se puede consultar o actualizar lo siguiente: nivel, grado y carreras en la que se dicta, temario, cronología, materias que el alumno debió haber cursado y aprobado para poder cursar la materia y docentes que la dictan.
- Planes de estudio: permite asignar o eliminar materias en los diferentes planes existentes y vigentes. Estos planes de estudio están clasificados por nivel, grado y carrera.
- Evaluaciones: el administrador del sistema podrá crear cuestionarios que evalúen el temario y cronología del curso al final del mismo para que sean contestadas por los alumnos, ya sea por Internet o que se impriman para que sean contestadas en un día u hora determinada.

3.3.1.4. Manejo de docentes

El módulo nos permite registrar los datos generales de docentes que imparten o no clases dentro del plantel, además de llevar la administración de los mismos como pueden ser: asistencias, horario, grupos, etc.

- Archivo: esta opción da la posibilidad de agregar, dar de baja y actualizar los datos de los docentes.
- Expediente: se consultan los datos generales del docente y se pueden actualizar datos de sus grupos relacionados. Lo que el docente o administrador del sistema puede actualizar es: calificaciones, asistencias, tareas o actividades de los alumnos registrados en los grupos que imparte clases.
- Evaluaciones: el administrador del sistema podrá crear cuestionarios que evalúen el desempeño del docente durante el curso para que sean contestadas por los alumnos al concluir el mismo, ya sea por Internet o que se impriman para que sean contestadas en un día determinado.

- Consultas: permiten visualizar el total del plantel docente ya sea global, por nivel, por grado. Además de visualizar su horas laboradas, faltas, ingreso en diferentes intervalos de tiempo y realizar comparaciones entre meses.

3.3.1.5. Documentos digitales

Mediante esta opción se pueden anexar los documentos digitalizados de alumnos, aspirantes y docentes, así en lugar de recibir copias y organizarlas en el archivo de modo manual, sólo se escanean los documentos originales y se anexan a su expediente digital, lo que hace más fácil la búsqueda y organización. Con esto se reduce espacio y costos. A su vez los alumnos, aspirantes y docentes pueden consultar estos documentos además de imprimirlos.

3.3.2. Tecnologías utilizadas y requerimientos mínimos

Si se opta por la opción de alquiler, sólo se necesitará de un navegador *web* para poder utilizar el sistema. Si se elige la opción de compra se necesitará instalar SQL Server para la base de datos.

3.4. Proyecto ALBA

Este proyecto (Open Computación, 2014) fue presentado y seleccionado en la convocatoria 2005 de los Proyectos Federales de Innovación Productiva. Está licenciado bajo GNU/GPL, y su código está disponible en Github.

3.4.1. Características

El software contempla un conjunto de módulos que pueden clasificarse en:

- Gestión de las unidades educacionales: establecimientos, ciclos, calendarios.
- Gestión de alumnos: legajos, seguimientos, consultas.
- Gestión de docentes: legajos, horarios, etcétera.

3.4.2. Tecnologías utilizadas y requerimientos mínimos

Al igual que Kimkëlen, ALBA está desarrollado bajo el framework Symfony (Potencier, 2014). Para su instalación se requerirá de un Servidor Web Apache 2.x, PHP (Versión 5) y un servidor de base de datos MySQL 4.1.x ó superior. Corre en sistemas operativos GNU/Linux y en Windows.

3.5. Por qué Kimkëlen

Entre la amplia oferta de SIS, ¿por qué se elige Kimkëlen como base y no se decide extender otro sistema? ¿Por qué no se elige utilizar un sistema que ya posea la funcionalidad de aplicaciones *frontend*? Se escogió el sistema Kimkëlen como objetivo ya que el mismo, a pesar de poseer menos funcionalidades que otros de los sistemas anteriormente descritos, se cree presenta ventajas por sobre los demás. A continuación se describirán dichas ventajas.

3.5.1. Mejor adaptación al manejo escolar en Argentina y Latinoamérica

Ninguno de los sistemas anteriormente descritos fue desarrollado pensado para ser utilizado en escuelas de enseñanza pre-universitaria de Argentina.
Los sistemas educativos están sujetos a múltiples variables que pueden hacer que uno se diferencie de otro en forma radical. Los pilares y esquemas de enseñanza están sujetos a variables que comprenden (entre otras tantas) y/o están influenciadas por:

- País
- Región
- Economía
- Cultura
- Sociedad
- Religión

Así, la terminología, el circuito de administración y comunicación, el énfasis que se hace en los datos que se guardan y en cómo se muestran al usuario final pueden variar mucho de un lugar a otro.

Como ejemplo conciso y para esclarecer la discusión, podemos ver que *software* como Fedena presenta módulos para el manejo de *campus* y abono de matrículas y cuotas. En Argentina, y en particular para las escuelas propósito que fueron propulsoras de la problemática a resolver, dichos módulos carecen de sentido; aquí la educación es pública y los *campus* universitarios no son lo que se acostumbra.

Partir de un sistema base que presenta funcionalidades que no se adaptan correctamente al esquema necesario no sería fructífero.

3.5.2. Mejor adaptación al manejo de escuelas de enseñanza pre-universitaria

El concepto de colegio pre-universitario es relativo e intrínseco a las Universidades Nacionales de nuestro país. En el presente trabajo no se ha realizado una investigación exhaustiva acerca de la existencia de establecimientos con comportamiento similar en otros países y/o regiones pero, luego de la de evaluación de los SIS anteriormente descritos, salta a la vista que ninguno excepto Kimkëlen se adecúa al manejo complejo y diferencial que presentan los colegios de esta modalidad.

A diferencia de un colegio primario o secundario convencional, las escuelas de enseñanza pre-universitaria comprenden el manejo de inasistencias por día y por materia, cursos con múltiples profesores y alumnos que cursan diferentes años

lectivos, materias que califican en conjunto con otras, régimen de cursada anuales, cuatrimestrales y bimestrales. Estas características y muchas otras distinguen un colegio de nivel pre-universitario de otros y requieren que un SIS para su gestión sea capaz de adaptarse a estas medidas.

3.5.3. Código liberado

Kimkëlen vela por el *software* libre. Su código se encuentra disponible en Github en su totalidad para todo aquél que quiera descargarlo (Desarrollo-CeSPI, 2014). La extensión de Kimkëlen representaría entonces un importante aporte a la comunidad de *software* libre y al desarrollo de competencias tecnológicas.

El no tener que pagar por el costo de las licencias de uso es un aspecto importante para nuestra realidad, y favorece avanzar en el concepto de legalidad.

Además, aportará a la promoción de la libertad como valor social, es decir, la posibilidad de sentir libertad como usuario, como ciudadano, en una sociedad que nos restringe y nos clasifica. Poder participar de las comunidades virtuales en torno al *software*, acceder a los contenidos de forma libre, y ser parte de las discusiones y de la co-creación de estos contenidos, permite desarrollar en la práctica, importantes valores sociales.

3.5.4. *Frontend apps* con tecnología de punta

Se planteará construir parcialmente las aplicaciones *frontend* de forma totalmente desacoplada del sistema *backend*. De esta forma, las aplicaciones *frontend* podrán ser implementadas con tecnología de punta, sin tener la obligación de que estén escritas en Symfony 1.2 como lo está Kimkëlen *backend*, versión del *framework* que ya no consta de mantenimiento por parte de sus desarrolladores -*legacy code*-.

4. Marco teórico

4.1. TIC en el contexto actual de gestión escolar

La informatización ha significado para la sociedad contemporánea un cambio de paradigma global y radical en referencia a las nuevas tecnologías. No resulta posible referirnos al mundo de hoy, hablar de la sociedad en su totalidad, sin tener en cuenta la mediación que las nuevas tecnologías llevan a cabo en nuestra vida cotidiana. Más allá del uso que podamos darle a la informática, del papel del individuo delante de la computadora, hay una metafísica de esta nueva ciencia que alcanza más allá de la conceptualización de la informática como herramienta; que habla del cambio inevitable en consecuencia de esta informatización (Gudiño, 2014).

En el universo de discurso de este trabajo, se puede denotar a la informática como una herramienta fundamental para la gestión académica de los alumnos, que brindará la digitalización de sus datos, de su entorno familiar y aportará a la comunicación e interacción de las distintas partes involucradas.

Actualmente, las instituciones educativas están siendo seducidas a migrar en forma paulatina sus actividades diarias a sistemas de gestión como solución a los problemas planteados en el capítulo anterior. De esta forma se traslada toda la actividad de procesos manuales hacia procesos informatizados y auditados. Por ejemplo, la tarea de registrar todas las calificaciones de las materias y calcular las promedios de cada una y el promedio general del estudiante se verá ampliamente simplificada, ya que el preceptor solo deberá encargarse de cargar las inasistencias; los promedios parciales y global serán calculados automáticamente por el sistema -siguiendo parámetros de comportamiento específico de la escuela-.

4.2. Las aplicaciones *web*

4.2.1. Evolución histórica

Al hablar del avance de la tecnología y puntualmente de los sistemas de gestión debemos destacar que hace algunos años éstos estaban basados exclusivamente en desarrollos sobre clientes locales, las llamadas aplicaciones *desktop.* En algunos casos no se utilizaban servidores, es decir el DBMS corría sobre el mismo equipo cliente, y en otros casos se montaba un servicio de red para que más de un cliente pudiera acceder a la misma base de datos. Esta red era local, por lo cual sólo podían acceder los equipos conectados a la misma.

Con el advenimiento de Internet, este tipo de desarrollos resulta poco práctico para el acceso a la información, justamente porque lo que se pretende es acceder a esa información desde cualquier oficina, hogar o espacio físico con conexión.

No obstante, estos desarrollos tradicionales tienen una gran ventaja que es la no exposición a otros equipos o a otras redes. Esta no exposición garantiza gran seguridad de los datos y brinda una estabilidad considerable, debido a que no existe un posible ataque al servidor desde el exterior.

En la actualidad, y más allá de las ventajas mencionadas anteriormente, todos los desarrollos de sistemas están mutando a entornos *web*. Las tecnologías más utilizadas en este tipo de desarrollos son normalmente lenguajes de programación como PHP, Ruby, Java o .Net, con DBMS como MySQL, Oracle, MongoDB, SQL Server o PostgresSQL.

Las aplicaciones *web* son aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor *web* a través de Internet o de una intranet mediante un *user agent* como lo es por ejemplo un navegador *web*.

En su libro, Luján Mora (2002) habla de las tres generaciones establecidas en la historia de de las aplicaciones *web*. No obstante, desde la fecha en que se ha establecido esta clasificación ha evolucionado notablemente la tecnología utilizada en la *web*, por lo que se puede añadir una cuarta categoría a la clasificación.

En la actualidad conviven las cuatro generaciones, aunque ya nadie desarrolla sitios *web* que caigan bajo las clasificaciones de primera o segunda generación.

4.2.1.1. Primera generación (1992-1994)

La creación de páginas web de esta generación estaba limitada por cuestiones tecnológicas tales como anchos de banda acotado, navegadores poco desarrollados, monitores monocromo, etcétera. Internet se empleaba como si fuera un medio de comunicación igual a los impresos (revistas, libros). Entre las características principales de este tipo de aplicaciones podemos citar:

- ➔ Tiempo de carga rápida: son páginas basadas en texto, con muy pocas imágenes y ningún recurso multimedia.
- ➔ Navegación poco estructurada, carente de coherencia.
- ➔ Empleo de saltos de línea como separadores y de líneas horizontales para separar secciones en una misma página.
- ➔ Poco uso de enlaces de páginas en un mismo sitio web.
- ➔ Poco o nulo empleo de elementos gráficos.
- ➔ No presentan problemas y discrepancias en la visualización en distintos navegadores.
- ➔ Están destinadas a alojar contenido educativo o científico en su mayoría; son pocas las empresas que poseen un sitio *web*.
- ➔ Las páginas son estáticas. Recién a fines de este período aparece CGI, que permite la generación dinámica de páginas.

4.2.1.2. Segunda generación (1995 - actualidad)

La diferencia distintiva para con las aplicaciones *web* de primera generación es la utilización masiva de elementos gráficos. En esta generación prima el uso de la

tecnología en lugar de tenerse en cuenta el propósito del sitio *web*. A criterio personal, esta tendencia no fue beneficiosa. Entre las características más resaltantes de las aplicaciones de esta generación encontramos:

- ➔ Tiempo de carga lento debido al uso excesivo de imágenes y animaciones, que encandilan al desarrollador y diseñador por su novedad y lo distraen, no teniendo así en cuenta al usuario final. Esta generación de sitios web resulta de pésimo rendimiento para conexiones lentas.
- ➔ Los íconos sustituyen a las palabras.
- ➔ El color de fondo deja de ser el clásico blanco o gris. Incluso se emplean imágenes como fondo.
- ➔ Los banners sustituyen a los títulos y encabezados de las páginas.
- ➔ La navegación suele ser jerárquica, no obstante no existe la filosofía de planificar la navegación del sitio.
- ➔ Aparecen tecnologías multimedia propietarias que requieren la instalación de un *plug-in* para su visualización.
- ➔ La tasa de utilización de CGI aumenta, lo que brinda la posibilidad de desarrollar aplicaciones con acceso a bases de datos.

4.2.1.3 Tercera generación (1996 - actualidad)

La páginas pertenecientes a esta generación son las más comunes en la actualidad. Se distinguen por:

- ➔ Tiempo de carga rápido: los desarrolladores se centran en el contenido en lugar de en la presentación. Se minimiza el tiempo de carga mediante el uso de CSS, uso minimalista de recursos gráficos y la optimización del código HTML. El rendimiento de las páginas se verifica empleando conexiones de distintas velocidades.

- ➔ Las páginas se limitan para que se puedan visualizar completas en una pantalla, sin necesidad de hacer *scroll*.
- ➔ Se simplifica la navegación. Se organiza la información a partir de una página inicial hasta una página final, ofreciendo distintos caminos.
- ➔ Los sitios *web* se crean teniendo en cuenta al usuario y sin perder de vista el objetivo del sitio.
- ➔ Se tienen en cuenta principios tipográficos y de organización visual de la información, como así también se emplean de forma coherente las imágenes, los colores, los íconos para crear una identidad corporativa.
- ➔ Hay planificación: la estructura del sitio *web* cobra gran importancia.
- ➔ Se consolidan las páginas dinámicas. Aparecen nuevas tecnologías como ASP que supone una verdadera revolución en la creación de páginas *web* dinámicas, y a partir de allí más tecnologías como JSP basada en Java y PHP.

4.2.1.4. Cuarta generación (1999 - actualidad)

Esta última generación de aplicaciones cuenta con las siguientes características principales:

- ➔ Se vuelven a emplear en exceso los recursos gráficos.
- ➔ Se extiende el uso de tecnologías poco empleadas hasta el momento, como CSS, y aparecen nuevas tecnologías como DHTML que permiten un mayor control sobre la visualización de las páginas *web*, pero a costa de incompatibilidades entre distintos navegadores.
- ➔ Uso de nuevas tecnologías multimedia que permiten crear un sitio *web* sin necesidad de emplear HTML.
- ➔ El desarrollo de sitios *web* está a cargo de un equipo multidisciplinario, compuesto por informáticos, diseñadores gráficos , expertos en contenido, etc.

- ➔ El aumento de ancho de banda permite *streaming* de video y audio en tiempo real.
- ➔ El objetivo al desarrollar es crear una experiencia completa desde que el usuario visualiza la primera página hasta que abandona el sitio *web.*
- ➔ La mayoría de las páginas *web* se crean a partir de información almacenada en bases de datos.

Estas aplicaciones son populares debido a la facilidad que presentan para actualizarlas y mantenerlas, por lo práctico que resulta un navegador *web* como cliente liviano, como así también porque son independientes de la plataforma, es decir, del sistema operativo subyacente. Otra gran ventaja es que no ocupan espacio en el cliente; no es necesario instalar ningún *softwar*e.
Aquí se introducen entonces el concepto de arquitectura cliente-servidor.

4.2.2. Arquitectura cliente - servidor

El modelo cliente-servidor se basa en mantener las aplicaciones y los datos en el servidor. Implica la existencia de una relación entre los procesos que solicitan servicios (clientes) y procesos que responden a estos servicios (servidores). De esta forma es el cliente quien realiza peticiones al servidor, que es quien le brindará las respuestas. Estos dos tipos de procesos pueden ejecutarse en el mismo procesador o en distintos.
Las ventajas que esta arquitectura permite son varias, entre ellas la centralización de la gestión de la información y la separación de responsabilidades, lo que simplifica y clarifica el diseño del sistema (Pérez, 2011).
La arquitectura cliente-servidor reemplaza a la arquitectura monolítica en la que no hay distribución, tanto a nivel físico como a nivel lógico. De esta forma, se puede obtener acceso a la información desde cualquier ubicación y el cliente no requiere una gran cantidad de almacenamiento.
Las aplicaciones *web* son un tipo de especial de aplicaciones cliente-servidor.

4.2.2.1. Separación de funciones

La arquitectura cliente/servidor nos permite la separación de funciones en tres niveles tal como se muestra en la figura 8.

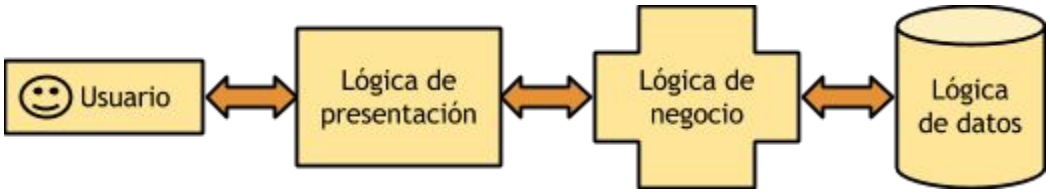

Figura 8. Separación de funciones arquitectura cliente-servidor

- **Lógica de presentación**: Es la encargada de la entrada y salida de aplicación con el usuario. Sus actividades principales se centran en obtener información del usuario a la lógica de negocio para su procesamiento, recibir los resultados del procesamiento de la lógica de negocio y presentar los resultados al usuario.
- **Lógica de negocio**: También llamada “lógica de aplicación”. Se encarga de gestionar los datos a nivel de procesamiento. Su acción es actuar como puente entre el usuario y los datos. Recibe la entrada de nivel de presentación, interactúa con la lógica de datos para ejecutar las reglas de negocio con las que debe cumplir la aplicación y envía el resultado del procesamiento al nivel de presentación.
- **Lógica de datos**: Se encarga de gestionar los datos a nivel de almacenamiento. Sus principales tareas consisten en almacenar, recuperar, mantener y asegurar la integridad de los datos.

Si un sistema distribuido está correctamente diseñado, los niveles anteriores pueden distribuirse y redistribuirse independientemente sin afectar al funcionamiento de la aplicación.

4.2.2.2. Descripción de una aplicación *web*

En las aplicaciones *web* suelen distinguirse tres niveles (arquitectura de tres capas). Aunque en verdad una arquitectura cliente-servidor puede presentar N capas, N capas con objetos, N capas con objetos y sistemas heredados.

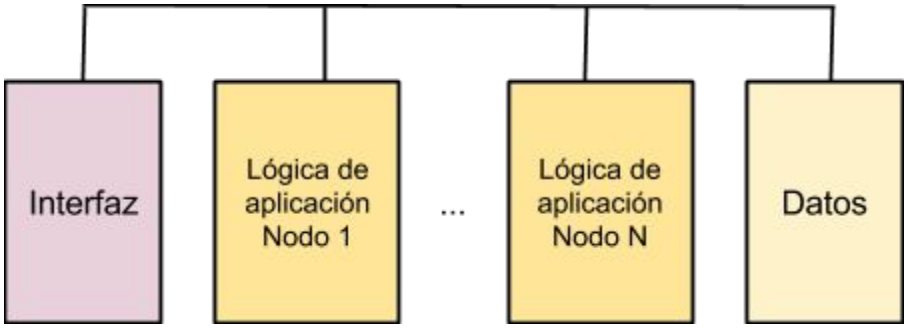

Figura 9

Arquitectura en tres capas:

- El nivel superior que interacciona con el usuario. Normalmente es un cliente *web*, es decir un navegador.
- El nivel inferior que es quien proporciona los datos.
- El nivel intermedio que es el encargado de procesar los datos a través de alguna tecnología web dinámica.

El nivel superior entonces, conformado por el cliente *web,* es un programa con el que interacciona con el usuario para solicitar a un servidor *web* el envío de los recursos que

desea obtener mediante el protocolo de comunicación, que usualmente es HTTP o HTTPS. El navegador *web* manda peticiones a la capa intermedia que ofrece servicios valiéndose de consultas y actualizaciones a la base de datos.
Tanto el cliente como el servidor y el protocolo mediante el cual éstos se comunican están estandarizados y no deben ser programados por el programador de aplicaciones.

4.2.2.3. Descripción de un sistema cliente/servidor

Un sistema cliente/servidor presenta por lo general las siguientes características:

1. Una combinación de la parte cliente, que interactúa con el usuario -hace de interfaz entre el usuario y el resto de la aplicación- y la parte servidor que interactúa con los recursos compartidos (por ejemplo, bases de datos).
2. La parte cliente y servidor tienen distintas necesidades al momento de ejecutarse: velocidad de procesador, memoria, velocidad y capacidad del almacenamiento secundario (discos duros), dispositivos de entrada/salida, etc.
3. El entorno suele ser heterogéneo, esto es, que se emplea *software* y *hardware* de distintos fabricantes. El *hardware* y sistema operativo del cliente y del servidor suelen diferir. Cliente y servidor se suelen comunicar a través de una API y RPC conocidas (por ejemplo ODBC para acceder a bases de datos).
4. La parte cliente se implementa normalmente haciendo uso de una GUI.

4.2.3. Arquitectura REST

REST es un tipo de arquitectura de desarrollo *web* que se apoya totalmente en el estándar HTTP. Dicha arquitectura permite la creación de “servicios” RESTful y aplicaciones que pueden ser usadas por cualquier dispositivo o cliente que entienda

HTTP, por lo que es increíblemente más simple y convencional que otras alternativas que se han usado en la última década, como SOAP y XML-RPC.

REST se definió en el 2000 por Roy Fielding, coautor principal también de la especificación HTTP.

REST podría considerarse como un *framework* para construir aplicaciones *web* respetando el protocolo HTTP. Es el tipo de arquitectura más natural y estándar para crear APIs para servicios orientados a Internet.

Las URLs permiten acceder a cada una de las páginas, secciones o documentos de un sitio *web*. Cada página, información en una sección, archivo, al hablar de REST, se nombran como recursos.

El recurso, por lo tanto, es la información a la que se quiere acceder, modificar o borrar, independientemente de su formato. Las URL son un tipo de URI que además de permitir identificar de forma única el recurso, permiten localizarlo para poder acceder a él o compartir su ubicación. Para manipular los recursos, el protocolo HTTP dota al programador de los siguientes métodos con los cuales operar:

- GET: Para consultar y leer recursos.
- POST: Para crear recursos.
- PUT: Para editar recursos.
- DELETE: Para eliminar recursos.
- PATCH: Para editar partes concretas de un recurso.

4.2.3.1. Servicio

Un servicio es una pieza de funcionalidad de negocio autocontenida. Puede ser simple (almacenar o recuperar datos de cliente) o complejo (un proceso de negocio para un pedido de cliente). Un servicio *web* permitirá el acceso a recursos. En una API de arquitectura REST, la idea de servicio como tal desaparece. Lo que se tendrá son recursos, accesibles por identificadores (URIs). Sobre esos recursos se puede realizar

acciones, generalmente diferenciadas a través de verbos HTTP distintos como se describió en la sección anterior.

4.2.3.2. Acople débil

El acople débil es el concepto de reducir las dependencias del sistema. Puesto que los procesos de negocio están distribuidos, es importante minimizar los efectos de modificaciones y fallos. De otra forma, las modificaciones se vuelven muy arriesgadas y los fallos del sistema pueden derrumbar el completo andamiaje del sistema. No obstante, existe un precio para el acople débil: la complejidad. Los sistemas distribuidos débilmente acoplados son más difíciles de desarrollar, mantener y depurar.

4.3. Desarrollo propuesto

Con lo investigado y partiendo de la base del sistema Kimkëlen *backend* y su base de datos, se propone el desarrollo de:

- API RESTful de Kimkëlen *backend*.

y el planeamiento y diseño de:

- Aplicación *frontend* para acceso de estudiantes.
- Aplicación *frontend* para acceso de tutores y/o padres de estudiantes.

Las aplicaciones *frontend* van a permitir consultar a quien corresponda la información pertinente acerca del progreso académico en la Institución educativa de un estudiante dado. Serán aplicaciones *web* y accesibles desde dispositivos móviles para aportar flexibilidad de uso. Su desarrollo se plantea de forma desacoplada al sistema Kimkëlen

backend, es decir, las aplicaciones no compartirán código y se comunicarán mediante una API REST que proveerá el acceso a los recursos que alimentarán a las aplicaciones *frontend*.

5. Marco práctico

5.1. Trabajo de campo

En la presente investigación se utilizaron encuestas por muestreo como instrumento de medición. Dichas encuestas fueron destinadas a directivos, preceptores, personal administrativo y profesores de ambos sexos de tres de los colegios de enseñanza pre-universitaria de la UNLP -Liceo Víctor Mercante, Bachillerato de Bellas Artes y Colegio Nacional Rafael Hernández-.

Algunas preguntas consistieron en una serie prefijada de alternativas, entre las cuales el encuestado escogió la que creyó conveniente, y otras consistieron en un campo de texto libre para que el encuestado argumente sus respuestas.

Dichas encuestas (ver Anexo I), además de basarse en obtener *feedback* de los usuarios sobre el sistema en general, se centraron en recolectar opiniones sobre el nivel de deseo de los usuarios de extender el sistema Kimkëlen con aplicaciones de acceso para alumnos y tutores, y si los usuarios creen que esto beneficiaría al circuito comunicacional escolar y cómo o, caso contrario, por qué creen que no sería beneficiosa la implementación de aplicaciones *frontend*.

Luego de cotejar, interpretar y analizar las respuestas obtenidas, los resultados obtenidos con respecto al pensamiento de los usuarios sobre las nuevas aplicaciones fueron los siguientes.

Tabla N°1. ¿Le gustaría contar con una aplicación para alumnos?

	Valor absoluto	Valor relativo
Sí	32	**82%**
No	7	**18%**

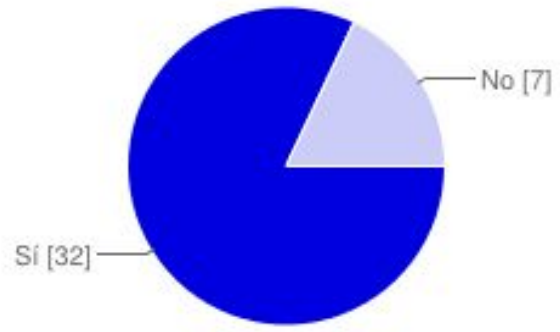

Figura 10

Tabla N°2. ¿Qué ventajas cree que aportaría contar con un acceso al sistema para estudiantes?

Entre las respuestas más populares:

Aporta al seguimiento de los estudiantes sobre sus trayectos.
Independencia para los alumnos, sobre todo los de años superiores.
El alumno verá reflejada su situación.
Ventajas a la comunicación alumnos - profesor - preceptores.
Les permitirá a los alumnos tener mayor manejo y organización.
Mayor seguimiento de calificaciones por parte de los alumnos.
Que los alumnos puedan acceder en el momento a información y favorezca cierta autonomía.
La posibilidad de consultar inasistencias.

Agilidad, ahorro de tiempo e insumos (tinta y papel).
Comunicación instantánea y autocontrol.
Poder tener el control diario y estricto de sus inasistencias a clase y calificaciones desde su hogar.
Posibilidad de que el alumno solicite pedidos *online*.
Inscripción a mesas *online*.
Verificación de datos personales por el usuario.
Los alumnos podrían organizarse mejor en relación al sistema de faltas, ver sus calificaciones y su desempeño académico en cualquier momento del año (y no sólo cuando se entrega el boletín).
Sería una introducción a un manejo más independiente, algo que luego en la vida universitaria se puede asemejar con el SIU Guaraní.
Organización.
Versatilidad.

Tabla N°3. ¿Qué funcionalidades cree que se les debería permitir realizar a los alumnos desde el sistema?

	Valor absoluto	Valor relativo
Ver calificaciones, asistencias y sanciones propias	28	**72%**
Integración con entornos virtuales de enseñanza y	12	**31%**

aprendizaje (como Moodle)		
Posibilidad de envío de mensajes a preceptores y docentes	8	**21%**
Ver notificaciones que sean cargadas por preceptores o autoridades de la institución (notas sobre asuetos escolares, paros, actos escolares, días de recreación, desinfección, duelo, etc)	19	**49%**
Imprimir y/o descargar documentos (boletines, analíticos)	9	**23%**
Otros	1	**3%**

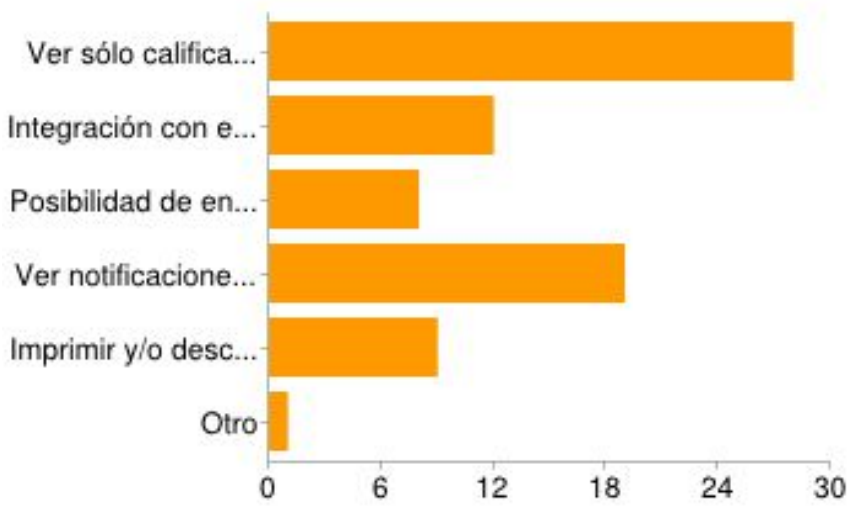

Figura 11

Tabla N°4. ¿Por qué no le gustaría que la institución cuente con un acceso al sistema para alumnos?

Entre las respuestas más populares:

Considero que a nivel de organización resultaría engorroso el tema de usuarios, contraseñas y privacidad.
Porque el vínculo docente-alumno no debe verse mediatizado por lo virtual. Todo lo que los alumnos necesitan saber lo consultan con nosotros. No debe descuidarse esta socialización.
Los alumnos reciben la información en papel y está bien que así sea.

Tabla N°5. ¿Le gustaría contar con un acceso al sistema para tutores/padres de los alumnos?

	Valor absoluto	Valor relativo
Sí	32	**82%**
No	7	**18%**

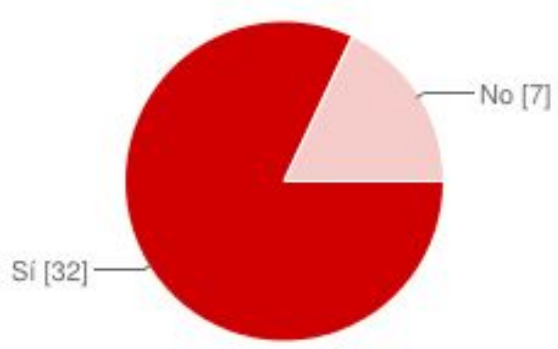

Figura 12

Tabla N°6. ¿Qué ventajas cree que aportaría contar con un acceso al sistema para tutores/padres de estudiantes?

Entre las respuestas más populares:

Le permitirá a los padres y/o tutores estar en mayor conocimiento del desempeño académico de sus hijos/as.
Tener acceso a la información no garantiza un seguimiento cercano y compromiso por parte de los padres u otros adultos responsables pero lo hace factible.
Podría aligerar los tiempos para algunos trámites; los responsables de los alumnos tendrían una vía más de comunicación con el colegio, sin perjuicio de lo cual podrían seguir usando las modalidades que hasta el día de hoy manejamos.
Ventajas en la comunicación padre-hijo y padre-escuela.
Facilidad de saber al instante el progreso de sus hijos.
Organización.
Versatilidad.
Agilización y fluidez de la comunicación padre/escuela.
Acceso de los padres a información que evite tener que acercarse a la escuela.
Mejorará el rendimiento de los alumnos indirectamente al verse los tutores comprometidos y enterados del progreso de su hijo en la institución.
Mejoraría el acceso a la información por parte de los tutores, pudiendo realizar un seguimiento más minucioso del desarrollo académico de los alumnos. Sin embargo, creo que su implementación requeriría una buena disposición por parte de los

tutores, como así también una capacitación para facilitar la accesibilidad al sistema.
El padre verá reflejada la situación de su hijo.
Considero que aportaría a los tutores un acceso más rápido a cierta información respecto del seguimiento de su hijo y favorece la comunicación escuela-familia en el aspecto informativo.
Los tutores van a tener sin duda mayor responsabilidad sobre el desempeño académico y actitudinal de los chicos, crearía una mayor comunicación entre todas las partes que hacen a una mejor escuela.
La comunicación lograría que mejore el compromiso de los padres con la educación de los hijos. Estarían al tanto del rendimiento del hijo con una inmediatez increíble y contribuirá en el mejoramiento escolar de sus hijos.
Ayudará en situaciones de excepción donde la familia no puede llegar a la escuela.
Practicidad.
Ahorro de insumos una vez que se logre una dinámica.
Le permitirá contar con información instantánea sobre calificaciones, ausentismo, sanciones, notificaciones.

Tabla N°7. ¿Qué funcionalidades cree que se les debería permitir realizar a los tutores?

	Valor absoluto	Valor relativo
Ver calificaciones, asistencias y sanciones de su	31	**79%**

hijo/estudiante a cargo únicamente.		
Descargar boletines, analíticos y otros documentos de su hijo/estudiante a cargo.	13	**33%**
Ver notificaciones que sean cargadas por preceptores o autoridades de la institución (por ej, citaciones, notas sobre asuetos escolares, paros, actos escolares, autorizaciones para paseos educativos, etc).	19	**49%**
Enviar mensajes a docentes, preceptores, autoridades.	8	**21%**
Otro	0	**0%**

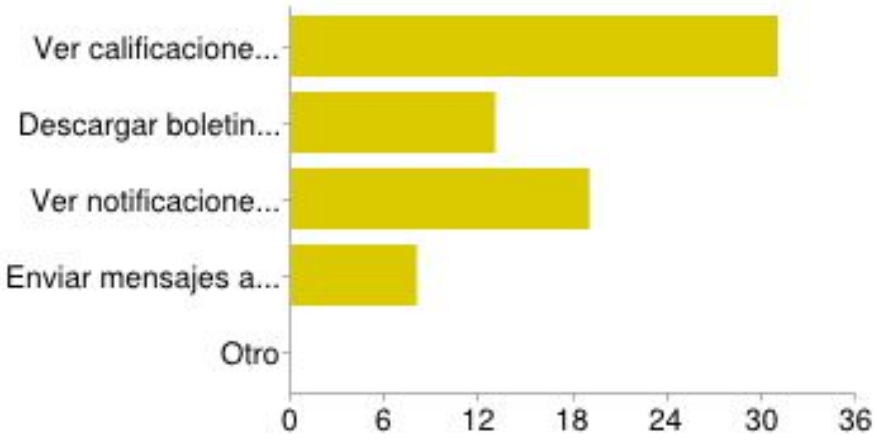

Figura 13

Tabla N°8. Justifique por qué cree que no sería beneficioso un acceso al sistema para tutores de estudiantes.

Entre las respuestas más populares:

No aporta a la socialización.
Porque los alumnos deben manejar su documentación. Los tutores deben tener acceso si el alumno o la institución los consideran necesario.
Porque los tutores reciben la información por escrito y está bien que así sea.
No todos los padres sabrían utilizarlo y complicaría la comunicación.
No aportaría a la independencia de los adolescentes.

Tabla N°9. Rol que desempeña Ud. en la institución.

	Valor absoluto	Valor relativo
Profesor/Docente	12	**31%**
Preceptor	17	**44%**
Jefe de preceptores	1	**3%**
Regente/Vicerregente	2	**5%**
Dpto. de alumnos/Alumnado	6	**15%**
Director/Vicedirector	1	**3%**

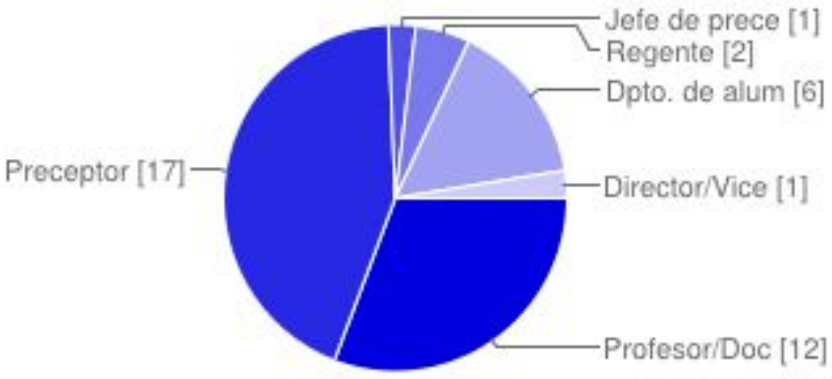

Figura 14

Tabla N°10. Edad

	Valor absoluto	Valor relativo
18 - 35 años	16	**41%**
36 - 50 años	14	**36%**
Mayor a 50 años	9	**23%**

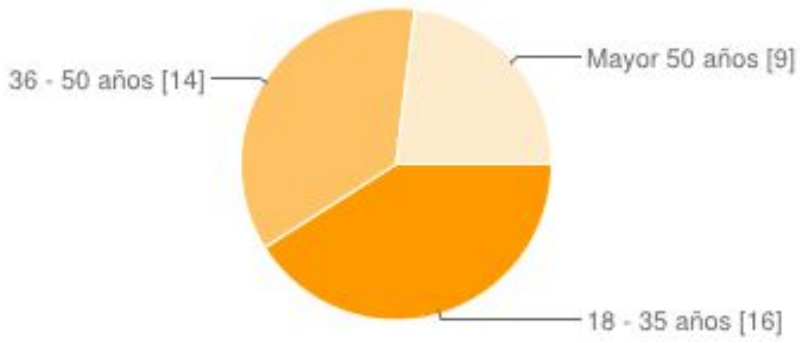

Figura 15

Tabla N°11. Sexo

	Valor absoluto	Valor relativo
Femenino	23	**59%**
Masculino	14	**36%**
No especifica	2	**5%**

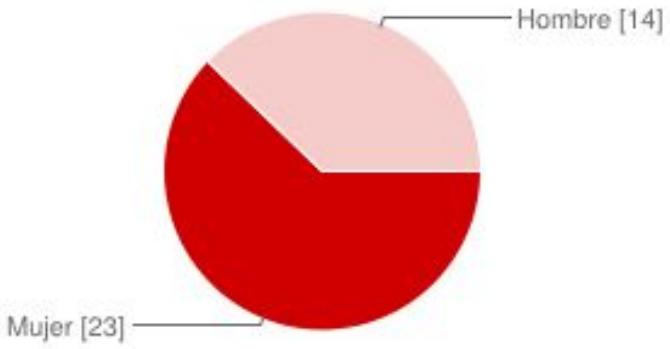

Figura 16

5.2. Aplicaciones *frontend*

En esta sección se detallarán los prototipos de aplicaciones *frontend* para estudiantes y para tutores de los estudiantes.

5.2.1. Casos de uso

5.2.1.1. Actores

UE: Usuario estudiante. Representará a un estudiante del colegio.

UT: Usuario tutor. Representará a un tutor o padre de uno o más estudiantes del colegio.

5.2.1.2. CU01 - Inicio de sesión

Versión	1
Descripción	Acceso del estudiante o tutor al sistema.
Actores involucrados	UE, UT
Precondición	El usuario debe existir en la base de datos de Kimkëlen y debe estar activo.
Flujo normal	- El usuario completa los campos del formulario de inicio de sesión, que serán e-mail y contraseña, y luego hace click en el botón Iniciar sesión. - Si la búsqueda de un usuario UE o UT por e-mail en la en la base de datos es exitosa y la contraseña ingresada es correcta, se devuelve código HTTP 200 OK, una *api key* y un objeto JSON con la información básica del usuario que será guardada en la sesión del cliente *web*. - El usuario es redirigido a su *dashboard* principal.
Postcondición	El usuario está autenticado o logueado en el sistema.

5.2.1.3. CU02 - Acceso al *dashboard* principal

Versión	1
Descripción	Visualización del tablero principal del sistema.

Actores involucrados	UE, UT
Precondición	El usuario debe estar logueado en el sistema.
Flujo normal	- El usuario acaba de ser redirigido luego del inicio de sesión o bien hizo *click* en el ícono de *Home*. - El sistema muestra una pantalla donde se visualiza toda la información del usuario y vínculos para navegar por la aplicación.
Postcondición	El usuario tiene acceso a su tablero principal y a todas las facilidades que éste le provee.

5.2.1.4. CU03 - Elección de estudiante

Versión	1
Descripción	Elección del estudiante de quien se quiere ver la información disponible.
Actores involucrados	UT
Precondición	El usuario debe estar logueado en el sistema. Se asume que el usuario tiene al menos un estudiante a cargo.
Flujo normal	- El sistema muestra en el *dashboard* del usuario un sector con la lista de los estudiantes que el UT tiene a cargo. - El usuario hace *click* en alguno de los nombres de estudiantes que se listan.
Postcondición	El usuario eligió un estudiante para poder consultar sus datos.

5.2.1.5. CU04 - Acceso a historial de calificaciones

Versión	1
Descripción	Acceso a las calificaciones del estudiante por materia y periodo, ordenadas por año lectivo.
Actores involucrados	UE, UT
Precondición	El usuario debe estar logueado en el sistema.
Flujo normal	- El usuario hace *click* en el *link* Calificaciones. - El sistema muestra un listado donde cada pestaña representa un año lectivo cursado por el alumno. El año seleccionado por defecto será el año lectivo actual. Cada pestaña muestra en una tabla, ordenada por materia y períodos, las notas obtenidas en cada instancia de cursada y examen y el promedio definitivo de cada materia. Al pie de dicha tabla se muestra el promedio anual del estudiante si el año lectivo está completo.
Postcondición	El usuario ha podido visualizar las calificaciones del estudiante.

5.2.1.6. CU05 - Acceso a listado de sanciones disciplinarias

Versión	1
Descripción	Acceso a las sanciones disciplinarias del estudiante de un año lectivo dado.
Actores involucrados	UE, UT
Precondición	El usuario debe estar logueado en el sistema.

Flujo normal	- El usuario hace *click* en el *link* Sanciones disciplinarias. - El sistema muestra un listado ordenado por fecha de las sanciones disciplinarias obtenidas por el alumno en el año lectivo elegido.
Postcondición	El usuario ha podido visualizar las sanciones disciplinarias del estudiante.

5.2.1.7. CU06 - Acceso a listado de inasistencias a clases

Versión	1
Descripción	Acceso a las inasistencias del estudiante de un año lectivo.
Actores involucrados	UE, UT
Precondición	El usuario debe estar logueado en el sistema.
Flujo normal	- El usuario hace *click* en el *link* Inasistencias. - El sistema muestra un listado ordenado por fecha de las inasistencias a clases y retrasos computados al alumno en el año lectivo dado.
Postcondición	El usuario ha podido visualizar las inasistencias computadas para ese estudiante.

5.2.1.8. CU07 - Cierre de sesión

Versión	1
Descripción	Finalización de la sesión de usuario.
Actores involucrados	UE, UT

Precondición	El usuario debe estar previamente logueado en el sistema.
Flujo normal	- El usuario hace *click* en el *link* de Cerrar sesión. - El sistema elimina todos los datos del usuario almacenados en el navegador o cliente *web.* - El usuario es redirigido a la pantalla de inicio de sesión.
Postcondición	El usuario ya no está logueado en el sistema.

5.2.2. Prototipos

Los prototipos de ambas aplicaciones -para estudiantes y tutores- fueron realizados utilizando el lenguaje de programación Ruby bajo el *framework* de desarrollo Ruby On Rails -RoR- (Heinemeier H., 2015).

El código fuente del *frontend* de estudiantes (ecorrons/kimkelen_students_frontend, 2015) y del *frontend* de tutores (ecorrons/kimkelen_tutors_frontend, 2015) están disponibles en Github para ser descargados y modificados por quien quiera contribuir al proyecto.

5.2.2.1. La aplicación *frontend* de estudiantes

Luego de iniciar sesión la aplicación muestra un *dashboard* principal en donde el estudiante podrá ver su información personal registrada en el sistema junto con su foto de perfil.

Luego, se le presentará al alumno un menú con posibles acciones:

- Calificaciones
- Sanciones disciplinarias
- Inasistencias a clase

Dentro de cada opción del menú, el alumno podrá ver la información correspondiente para todos aquellos años lectivos para los cuales presente historial académico.

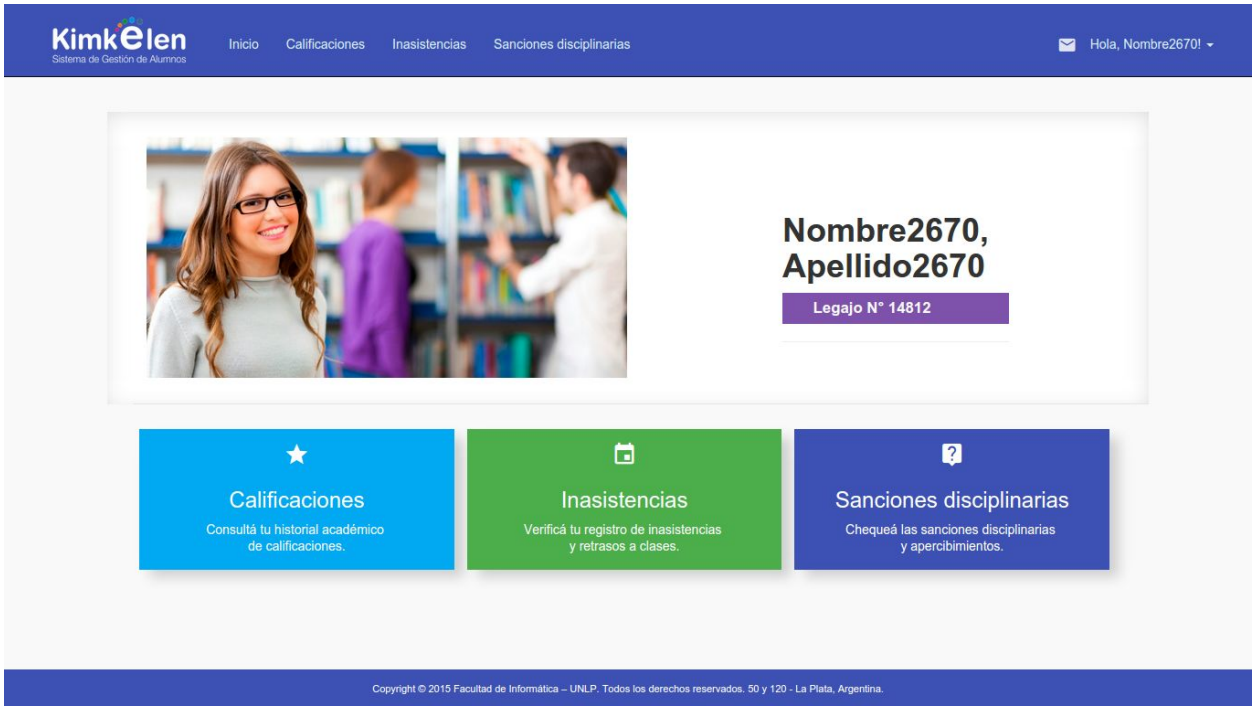

Figura 17. *Dashboard* de estudiantes

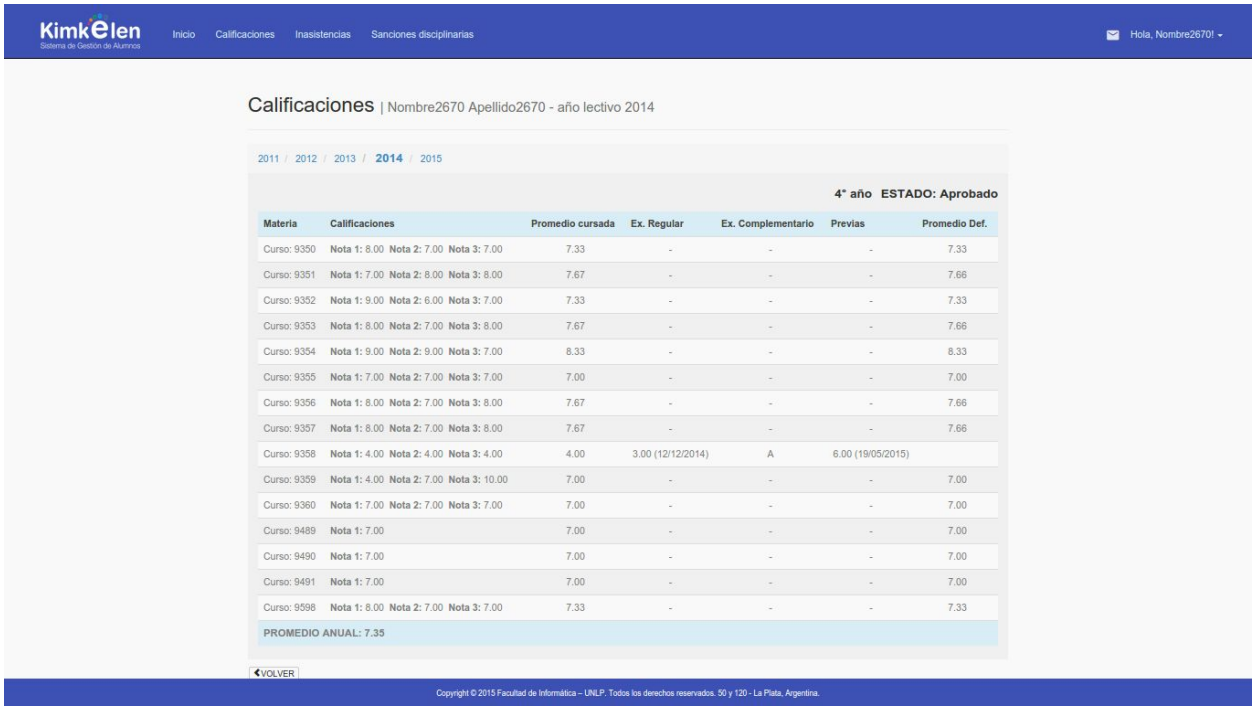

Figura 18. Calificaciones por año lectivo del estudiante

Figura 19. Sanciones disciplinarias para el estudiante en un año lectivo

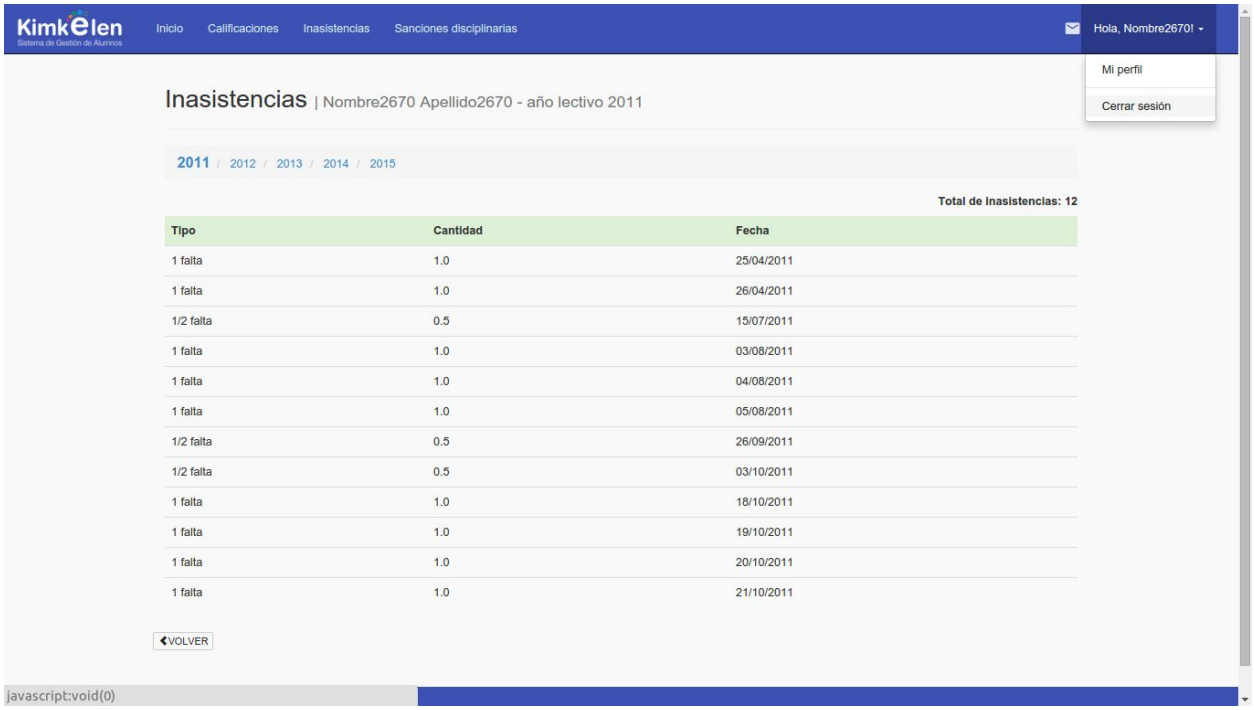

Figura 20. Inasistencias de un estudiante para un año lectivo dado.

5.2.2.2. La aplicación *frontend* de tutores

La aplicación de tutores será muy similar a la aplicación de estudiantes con la diferencia de que el *dashboard* principal, además de mostrar la información personal del tutor, mostrará una lista de los estudiantes que éste posee a cargo. Luego, para cada estudiante la información disponible será la misma que en el *frontend* para estudiantes.

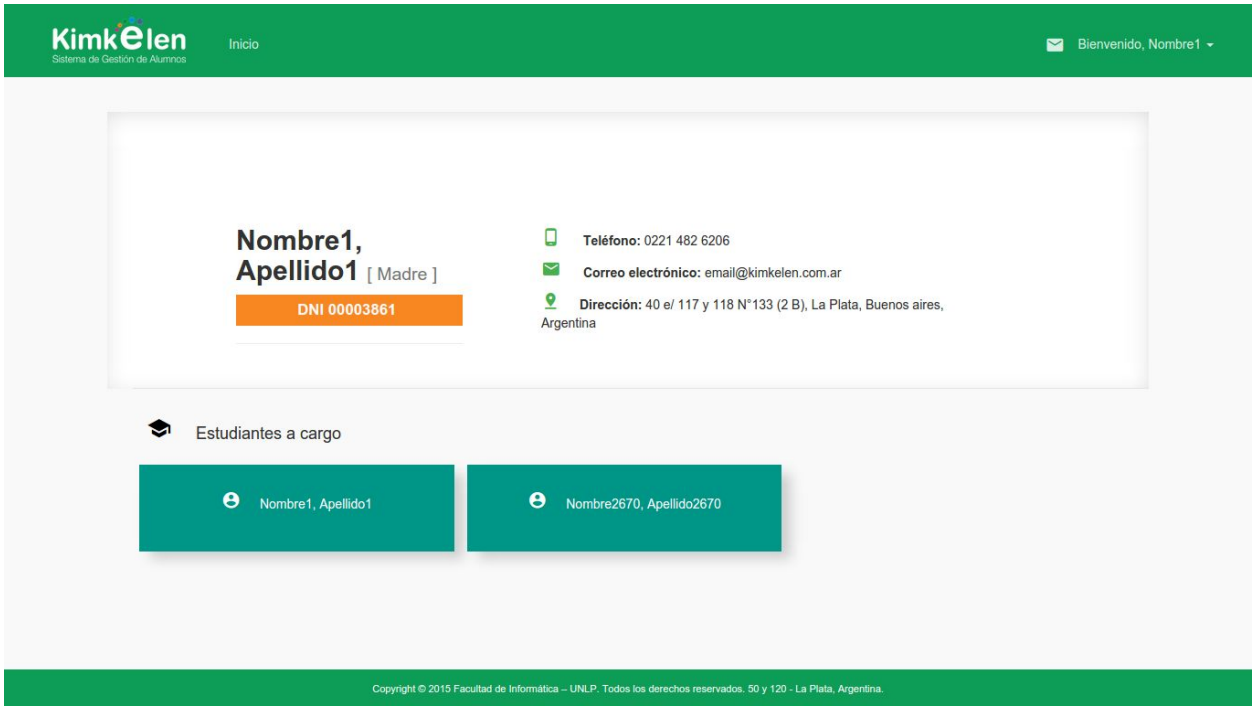

Figura 21. Dashboard de tutores.

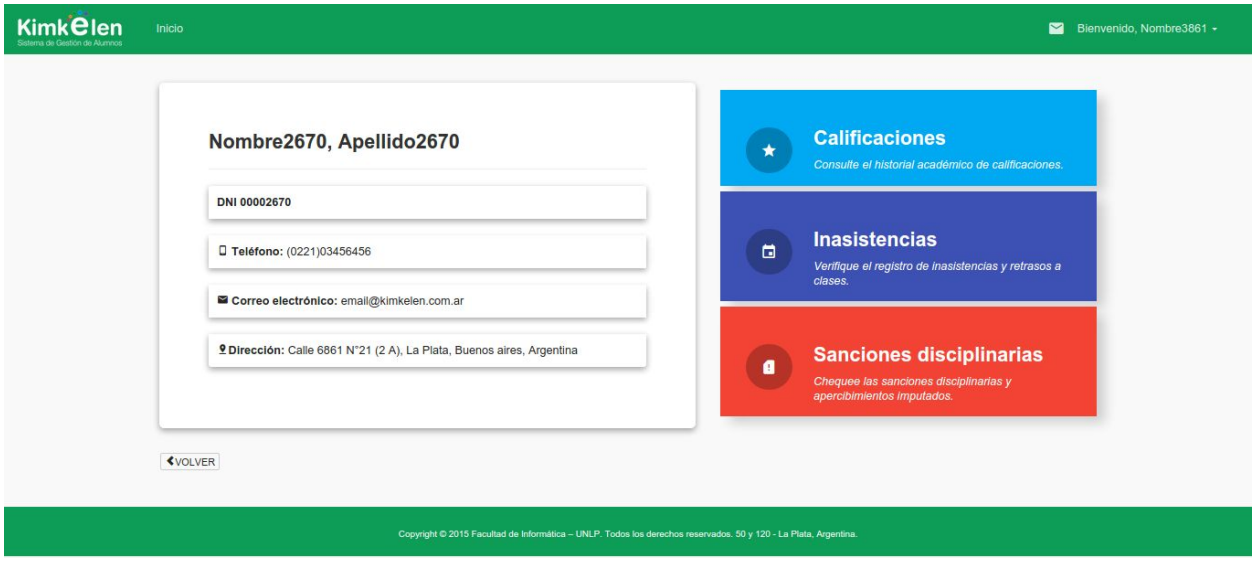

Figura 22. Vista información personal y acciones disponibles para uno de los estudiantes elegidos a cargo del tutor.

5.3. API RESTful

Para lograr la comunicación con Kimkëlen *backend* se diseñó una API REST con los servicios necesarios a ser consumidos por las aplicaciones *frontend*.
La situación actual, tal cual fue descrita en secciones anteriores, consiste en el sistema Kimkëlen *backend* construido sobre el *framework web* Symfony 1.2 (Potencier, 2014). El flujo de interacción se ilustra en la Figura 23.

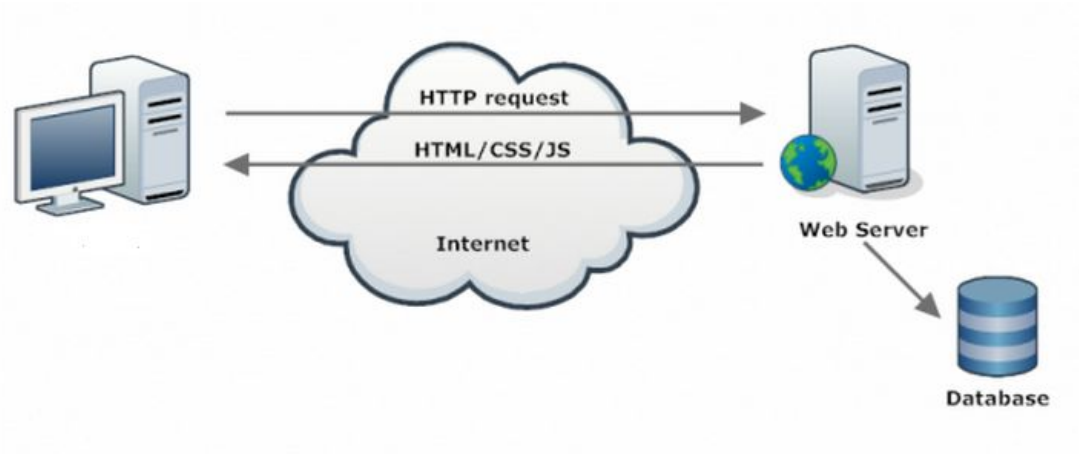

Figura 23

Para alimentar las aplicaciones *frontend* y para acelerar el desarrollo, la solución más viable será construir una API que aloje una serie de servicios REST necesarios para la interacción. Así se evitará la duplicación de código y se seguirá la filosofía del principio DRY. Aún se tendrá, como se muestra en la Figura 24, un servidor *web* que producirá no solo HTML y código javascript sino también recursos XML o JSON que seguirán el estándar REST.

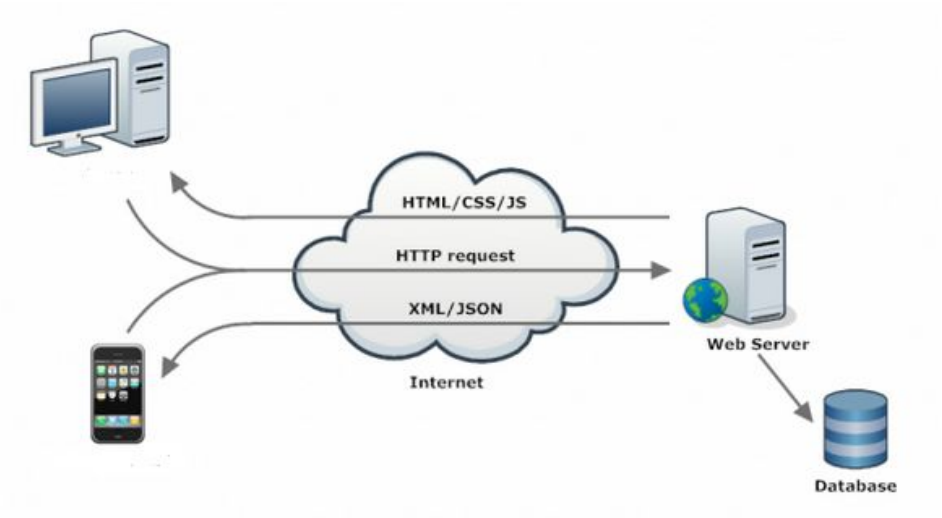

Figura 24

5.3.1. Descripción de los servicios REST

Los principales servicios a desarrollar, especificando su función, parámetros, forma de invocación y valores de retorno para todos los casos están descritos en el Anexo II.

5.3.2. Ejemplo de interacción

A continuación se ilustra con un ejemplo la interacción Kimkëlen *backend* - BBDD - API - Kimkëlen *frontend*. En la figura 25 se observa el diagrama de comunicación de las distintas partes involucradas.

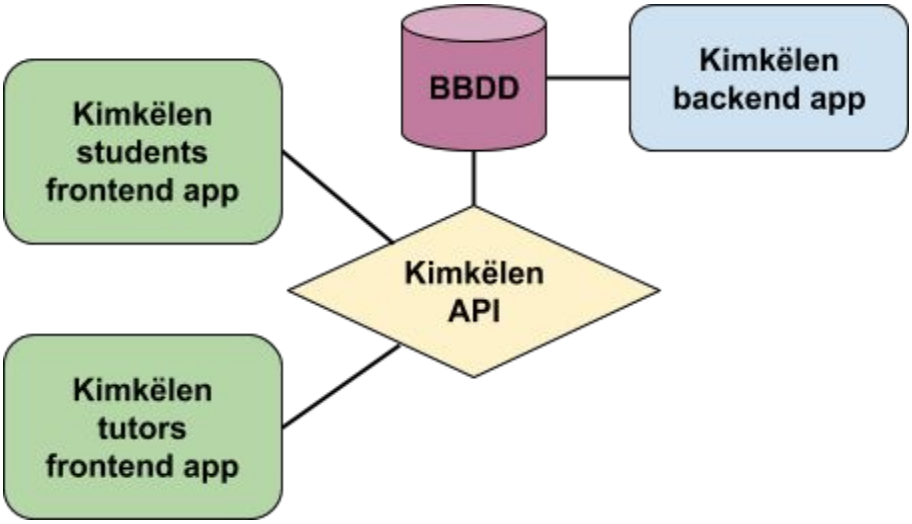

Figura 25

Supongamos que un profesor carga las notas trimestrales al curso que dicta desde Kimkëlen *backend*. Al enviar el formulario de carga de calificaciones las nuevas calificaciones serán guardadas en la base de datos del sistema.
Luego, un alumno inicia sesión desde su dispositivo móvil a Kimkëlen *frontend* y hace click en el vínculo “Calificaciones” para el año lectivo actual. Para poder mostrar el listado de calificaciones, el *frontend* dialogará con la API haciendo la consulta correspondiente a la URI:

api/v1/students/:id/marks/2015

Al recibir la petición HTTP GET, la API se conectará a la base de datos de Kimkëlen, hará la consulta SQL correspondiente, recuperará la información, armará el JSON correspondiente y lo devolverá en el cuerpo del el HTTP *Response*. Así, el *frontend* de estudiantes desde el cual el estudiante hizo el HTTP *Request* recibe la respuesta, procesa el JSON y dibuja la vista, permitiéndole ver la nueva calificación recientemente cargada por el profesor.

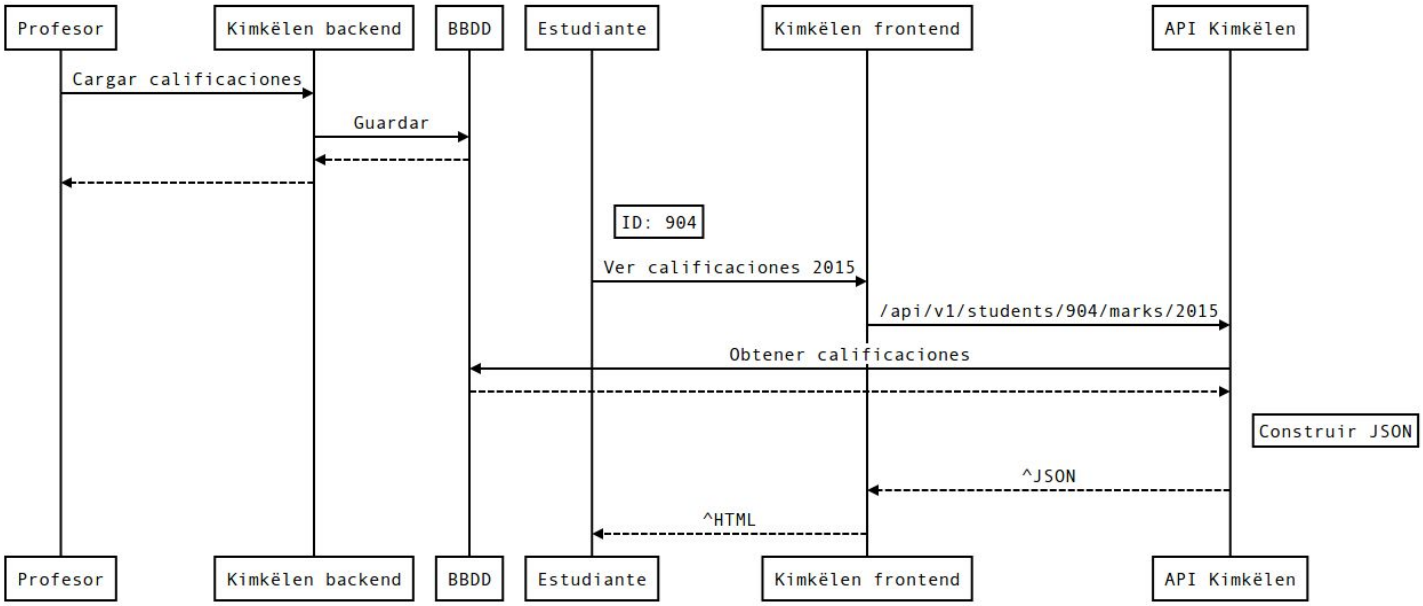

Figura 26. Diagrama de secuencia de interacción

5.3.3. Tecnología utilizada

La API RESTful fue programada bajo el lenguaje de programación Ruby. Más específicamente se desarrolló sobre Rails::API (Pastorino, da Silva y Klabnik, 2015), que conforma un subconjunto de una aplicación sobre el *framework* de desarrollo Ruby on Rails -RoR- (Heinemeier H., 2015). Rails::API fue creado para aplicaciones que no requieren todas las funcionalidades que una aplicación completa RoR ofrece. Es un poco más ligero, y por lo tanto un poco más rápido que una aplicación normal RoR.

En la construcción de la API se tuvieron en cuenta innovadores estilos de diseño como lo es *Hypermedia Linking* (Zazueta, 2015). En una API REST, este concepto aplica de manera que ésta sea capaz de funcionar como lo hace una página *web*, proporcionando al usuario una guía sobre qué tipo de contenido se puede recuperar, o qué acciones puede realizar, así como los vínculos adecuados para hacerlo. La idea es

proporcionar a través de vínculos o *links* información para dirigir al usuario hacia las próximas acciones posibles que pueden tomar basado en el objeto o "página" que están navegando.

Para los valores de retorno que devuelve cada uno de los servicios se utilizó Jbuilder (Rails - JBuilder, 2015), ya que a través de un DSL provee una forma fácil y ordenada de generar grandes estructuras JSON.

El código fuente de la API se encuentra disponible en Github (ecorrons/kimkelen_api, 2015) para que cualquiera pueda descargarlo y aportar a su mejora.

6. Conclusiones y trabajo a futuro

En lo sucesivo, la implementación completa y puesta en producción de las aplicaciones Kimkëlen *frontend* -que lograrán la informatización completa del circuito comunicacional en escuelas de enseñanza pre universitaria- traerá consigo diversos cambios no solo en la gestión académica sino también en la percepción de todos los usuarios activos de la comunidad educativa -autoridades, docentes, alumnos y tutores- sobre la importancia del dato certero, la detección temprana y en tiempo real de situaciones e incidentes que suceden dentro del recinto escolar y la concepción de una base sólida de información histórica.

Es posible considerar que se han alcanzado los objetivos planteados en la introducción del proyecto. A continuación se describen en detalle los aportes realizados en esta tesis.

6.1. Aportes realizados

- Se logró tener una sólida base de conocimiento sobre la gestión de la comunicación de todos los involucrados en el ámbito educativo escolar y un resultado positivo del estudio sobre la mejoras que la implantación de TIC supone.
- Se desarrolló una API RESTful del sistema Kimkëlen con los servicios necesarios para poder alimentar a las aplicaciones *frontend*.
- Se desarrollaron prototipos de aplicaciones *frontend* para alumnos y sus tutores, considerando la situación problemática estudiada y partiendo del proyecto Kimkëlen *backend*. De esta forma, quedó demostrado el potencial que se encuentra en el uso de sistemas *web*, habiendo pasado éstos de ser una revolución tecnológica a una revolución social, tras convertirse en medios libres y descentralizados de creación y distribución de contenidos.

- Se planteó fomentar, a través de la cultura tecnológica, el desarrollo y bienestar estudiantil y de su círculo familiar, como así también de los docentes, administrativos y directivos de las escuelas.
- Se realizó un aporte a la comunidad de *software* libre y a la práctica de *Green Computing* proponiendo incorporar en la totalidad de la gestión de escuelas un mecanismo de flujo electrónico -*paperless*- de información.

6.2. Trabajos futuros

- Incorporar autenticación a API RESTful.
- Extensión de los prototipos de aplicaciones *frontend* propuestos considerando:
    - Aprovechar servicios de terceros para maximizar la fortaleza de los servicios, esto es, proveer *login* a través de redes sociales.
    - Adaptación a dispositivos móviles.
    - Menú de notificaciones en ambas aplicaciones, de manera que padres y estudiantes puedan recibir notificaciones enviadas por las autoridades del colegio ya sea en forma global o a un alumno o tutor en particular.
- Integración de la aplicación *frontend* de estudiantes con entornos virtuales de enseñanza y aprendizaje tales como *Moodle*.
- Realizar pruebas de aceptación y usabilidad de las aplicaciones *frontend* con el usuario final.

Acrónimos

ABM *Alta, Baja y Modificación*

Denominan las operaciones básicas del almacenamiento persistente. Se utiliza el acrónimo CRUD para lo mismo, obviando la operación de Obtener.

API *Application Programming Interface*

Una Interfaz de Programación de Aplicaciones es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro *software* como una capa de abstracción.

BBDD *Base de Datos*.

Una base de datos o banco de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

CeSPI *Centro Superior para el Procesamiento de la Información*

Es el centro de cómputos de la Universidad Nacional de La Plata en donde se realizan las tareas relacionadas con los distintos sistemas que brindan servicios a dicha universidad.

CGI *Common Gateway Interface*

Estándar que permite el intercambio de información entre un servidor y un programa externo al servidor.

CRUD *Create, Read, Update, Delete*

Crear, Obtener, Actualizar y Borrar. Se usa para referirse a las funciones básicas en bases de datos o la capa de persistencia en un software.

CSS *Cascading Style Sheets*

Tecnología desarrollada por la W3C empleada en la creación de páginas *web* que permite un mayor control sobre el lenguaje HTML. Permite crear hojas de estilo que definen cómo se tiene que mostrar cada elemento. El término “en cascada” indica que varias hojas de estilo pueden aplicar sobre una misma página.

DBMS *Database Management System*

Es un conjunto de programas que se encargan de manejar la creación y todos los accesos a las bases de datos. Los ejemplos más populares son MySQL, PostgreSQL, Oracle, Microsoft SQLServer, SQLite.

DHTML *Dynamic HTML*

Designa el conjunto de técnicas que permiten crear sitios web interactivos utilizando una combinación de lenguaje HTML estático, un lenguaje interpretado en el lado del cliente (como JavaScript), el lenguaje de CSS y la jerarquía de objetos de un DOM.

DRY *Don't Repeat Yourself*

Es un principio de desarrollo de *software*, destinado a reducir la repetición de información de todo tipo, especialmente útil en los sistemas de arquitecturas de múltiples niveles. El principio DRY sostiene que: "Cada pieza de conocimiento debe tener una única e inequívoca forma de representación dentro de un sistema."

DSL *Domain Specific Language*

Es un lenguaje de especificación dedicado a resolver un problema en particular, representar un problema específico y proveer una técnica para solucionar una situación particular.

ERP *Enterprise Resource Planning*

Los sistemas de planificación de recursos empresariales son sistemas de información gerenciales que integran y manejan muchos de los negocios asociados con las operaciones de producción y de los aspectos de distribución de una compañía en la producción de bienes o servicios.

GNU GPL v2.0 *GNU General Public License*

Licencia Pública General de GNU. Es la licencia más usada en el mundo del *software* y garantiza a los usuarios finales la libertad de usar, estudiar, compartir (copiar) y modificar el *software*. Su propósito es declarar que el software cubierto

por esta licencia es libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

GUI *Graphic User Interface*

Es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual una interacción amigable con un sistema informático.

HTML *Hypertext Markup Language*

Lenguaje compuesto por una serie de etiquetas o marcas que permiten definir el contenido y la apariencia de las páginas *web.*

HTTP *HyperText Transfer Protocol*

Es el protocolo de aplicación usado en cada transacción de la World Wide Web.

HTTPS *HyperText Transfer Protocol Secure*

Es un protocolo basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de HTTP.

JSP *JavaServer Pages*

Tecnología que ayuda a los desarrolladores de software a crear páginas web dinámicas basadas en HTML, XML, entre otros tipos de documentos. JSP es similar a PHP, pero usa el lenguaje de programación Java.

ODBC *Open DataBase Connectivity*

Es un estándar de acceso a las bases de datos cuyo objetivo es hacer posible el acceder a cualquier dato desde cualquier aplicación, sin importar qué DBMS almacene los datos.

PHP *Hypertext Preprocessor*

Lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos.

REST *Representational State Transfer*

Es una técnica de arquitectura software para sistemas hipermedia distribuidos.

RPC *Remote Procedure Call*

Protocolo que permite a un programa ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos. Las RPC son muy utilizadas dentro de la comunicación cliente-servidor. Siendo el cliente el que inicia el proceso solicitando al servidor que ejecute cierto procedimiento o función y enviando éste de vuelta el resultado de dicha operación al cliente.

RRHH *Recursos Humanos*

SOAP *Simple Object Access Protocol*

Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

TIC *Tecnologías de la Información y de la Comunicación*

Las Tecnologías de la Información y la Comunicación son un conjunto de servicios, redes, software y hardware que tienen como fin la mejora de la calidad de vida de las personas dentro de un entorno, y que se integran a un sistema de información interconectado y complementario.

URI *Uniform Resource Identifier*

Un identificador de recursos uniforme es una cadena de caracteres que identifica los recursos de una red de forma unívoca.

URL *Uniform Resource Locator*

Un localizador de recursos uniforme es un identificador de recursos uniforme (URI) cuyos recursos referidos pueden cambiar, esto es, la dirección puede apuntar a recursos variables en el tiempo. Están formados por una secuencia de caracteres, de acuerdo a un formato modélico y estándar, que designa recursos en una red, como Internet.

Glosario

aplicación backend: hace referencia a la visualización del administrador del sitio o sistema.

aplicación frontend: Hace referencia a la visualización del usuario que navega el sistema.

bug: (del inglés, «bicho») Es un error o fallo en un programa o sistema de software que desencadena resultados indeseados.

core: Núcleo.

dashboard: tablero.

desktop: Escritorio. En el contexto de aplicaciones, una aplicación desktop es aquélla en la cual normalmente no iniciamos sesión, sólo se inicia sesión al iniciar el sistema operativo, no requieren conexión a internet para funcionar.

home: Hogar. Usualmente el *homepage* de una aplicación hace referencia a la página de inicio. Representa la página por defecto del sitio.

intranet: Es una red informática que utiliza la tecnología del protocolo de Internet para compartir información, sistemas operativos o servicios de computación dentro de una organización.

javascript: Comúnmente abreviado “JS”, es un lenguaje de programación interpretado.

mobile: (de "*mobile phone*") Significa teléfono celular en ingles.

open source: Significa código abierto. Es la expresión con la que se conoce al software distribuido y desarrollado libremente.

plug-in: (del inglés, «enchufable» o «inserción») Es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica. Esta aplicación adicional es ejecutada por la aplicación principal. También se conocen como *add-on* («añadido»), conector o extensión.

paperless: Desmaterialización, digitalización o virtualización. Se refiere, en relación a una empresa, una organización o una institución cualquiera, al reemplazo de los soportes tradicionales de información por ficheros informáticos y computadoras.

scroll: Se denomina así al movimiento en 2D de los contenidos que conforman la ventana que se muestra en una aplicación informática (por ejemplo, una página *web* visualizada en un navegador web).

streaming: También denominado transmisión, lectura en continuo, difusión en continuo, descarga continua. Es la distribución digital de multimedia a través de una red de computadoras de manera que el usuario consume el producto, generalmente archivo de video o audio, en paralelo mientras se descarga.

template method: Patrón de comportamiento cuyo propósito es definir en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de sus pasos. Permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar su estructura.

user agent: Agente de usuario. Es una aplicación informática que funciona como cliente en un protocolo de red; el nombre se aplica generalmente para referirse a aquellas aplicaciones que acceden a la World Wide Web.

Referencias

Apache Software Foundation. (2014). *Apache HTTP Server Project*. Recuperado de http://www.apache.org/

Buch, T. (2013). *Desarrollo y ecopolítica. Los grandes debates de la tecnología, el ambiente y la sociedad*. 1ra edición. Carapachay, Bueno Aires, Argentina: Lenguaje claro Editora.

Brampton, A. (2015). *js-sequence-diagrams.* Recuperado de: http://bramp.github.io/js-sequence-diagrams/

Castells, M. (2005). *La era de la información: Economía, Sociedad y Cultura*. *La Sociedad Red. Vol. 1*. Madrid, España: Alianza Editorial.

CeSPI. (2014). *CeSPI - UNLP - Kimkëlen*. Recuperado de http://www.cespi.unlp.edu.ar/kimkelen

Desarrollo-CeSPI. (2014). *Desarrollo-CeSPI/kimkelen*. Recuperado de https://github.com/Desarrollo-CeSPI/kimkelen

Dougiamas, M. (2014). *Moodle - Open Source Learning Plataform*. Recuperado de https://moodle.org/?lang=es

ecorrons/kimkelen_api. (2015). ecorrons/kimkelen_api. Recuperado de https://github.com/ecorrons/kimkelen_api

ecorrons/kimkelen_students_frontend. (2015). ecorrons/kimkelen_students_frontend. Recuperado de https://github.com/ecorrons/kimkelen_students_frontend

ecorrons/kimkelen_tutors_frontend. (2015). ecorrons/kimkelen_tutors_frontend. Recuperado de https://github.com/ecorrons/kimkelen_students_frontend

Elizondo Martínez, J. O. (2009). El individuo ante el tiempo atemporal. *Argumentos (México)*, *22*(60), 81-90. Recuperado de http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S0187-57952009000200005&lng=es&tlng=es

Facebook. (2014). *Facebook*. Recuperado de https://www.facebook.com

Fielding R., Gettys J., Mogul J., Frystyk H., Masinter L., Leach P., Berners-Lee T. *RFC 2616 - Hypertext Transfer Protocol -- HTTP 1.1.* Recuperado de https://tools.ietf.org/html/rfc2616#section-10

Foradian Technologies. (2014). *Open Source Student Information System - Fedena*. Recuperado de http://www.projectfedena.org/

Fredrich, T. (2014). *What is REST?*. Recuperado de http://www.restapitutorial.com/lessons/whatisrest.html

Gamma, E., Helm, R., Johnson, R., Vlissides, J. (2003). *Patrones de diseño. Elementos de software orientado a objetos reutilizable*. Madrid, España: Pearson Educación.

Giner de la Fuente, F., Gil Estallo M. A. (2004). *Los sistemas de información en la sociedad del conocimiento*. Madrid, España: ESIC.

Gudiño, M. (2014). *Conceptualización de la información.* Recuperado de http://www.slideshare.net/jagp2704/conceptualizacion-de-la-informatizacion

Hamiduzzaman, M. (2012). E-governance in management of education system in Bangladesh: Innovations for next generation level. *Universal Journal of Education and General Studies*, *1*(7), 195-209. Recuperado de http://universalresearchjournals.org/ujegs/pdf/2012/July/Hamiduzzaman.pdf

Heinemeier H., D. (2015). *Ruby on Rails*. Recuperado de http://rubyonrails.org

Joshi, G. (2013). *Management Information Systems*. Nueva Delhi, India: Oxford University Press.

Kendall, K. E., Kendall, J. (2005). *Análisis y diseño de sistemas*. 6a. Edición. México: Pearson Educación.

Kumar, P. R. y K. Murugadoss. (2014). *E-Governance and Impact in Educational Sector*. Recuperado de http://www.indianmba.com/Faculty_Column/FC1231/fc1231.html

Luján Mora, S. (2002). *Programación de aplicaciones web: Historia, Principios básicos y Clientes web*. Alicante, España: Club Universitario.

Marqués, A. (2014). *Conceptos sobre APIs REST*. Recuperado de http://asiermarques.com/2013/conceptos-sobre-apis-rest/

Open Computación. (2014). *Proyecto ALBA*. Recuperado de http://www.proyectoalba.com.ar/

Oracle Corporation. (2014). MySQL::The world's most popular open source database. Recuperado de https://www.mysql.com/

Pastorino, S., da Silva C. A. & Klabnik S. (2015). *rails-api*. Recuperado de https://github.com/rails-api/rails-api

Pérez, J. M. (2011). *Virtualización y Green IT*. Tesis Lic, Univ. Nac. De La Plata, La Plata, Argentina.

Ponce Regalado, F. y Rojas Sifuentes, W. (2014). *Promoción y desarrollo de las TIC en América Latina*. Recuperado de http://www.slideshare.net/ACORN-REDECOM/promocin-y-desarrollo-de-las-tic-en-amrica-latina-ponce-2010

Potencier, F. (2014). *Symfony*. Recuperado de http://symfony.com

Rails - JBuilder. (2015). *Jbuilder*. Recuperado de https://github.com/rails/jbuilder

Readsoft. *Almacenamiento digital*. (2014). Recuperado de http://cl.readsoft.com/terminologia/digitalizacion-de-documentos/almacenamiento-digital

Rodríguez C., Pérez J.P. & Corrons M. E. (2014). *Contribuciones de software de código abierto*. Recuperado de https://speakerdeck.com/chrodriguez/cisl-choiquecms-kimkelem-meran-software-liberado-por-la-unlp

Rodríguez, Christian; Pérez J. P.; Corrons M. E. (2014). *Driving schools to automation through Kimkëlen software*. Procedia - Social and Behavioral Sciences, 189, 367–370. doi:10.1016/j.sbspro.2015.04.003

Selwyn, N. (2011). *Schools and Schooling in the Digital Age; A critical analysis*. Nueva York, NY, Estados Unidos: Routledge.

Technoware. (2014). *mi-escuela.com*. Recuperado de http://www.mi-escuela.com/

Twitter. (2014). *Twitter*. Recuperado de https://twitter.com

Zazueta, R. (2015). *Hypermedia linking*. Recuperado de http://www.narwhl.com/hypermedia-linking

Anexo I. Sistema de alumnos Kimkëlen

Encuesta destinada a preceptores, profesores y autoridades de la Institución

*Obligatorio

1. **Rol que desempeña Ud. en la Institución** *
   Marca solo un óvalo.
   - ◯ Profesor/Docente
   - ◯ Preceptor
   - ◯ Jefe de preceptores
   - ◯ Regente
   - ◯ Dpto. de alumnos/Alumnado
   - ◯ Director/Vicedirector

2. **Edad**
   Marca solo un óvalo.
   - ◯ 18 - 35 años
   - ◯ 36 - 50 años
   - ◯ Mayor 50 años

3. **Sexo**
   Marca solo un óvalo.
   - ◯ Mujer
   - ◯ Hombre

4. **¿En qué aspectos cree que Kimkëlen ayudó a su institución?** *
   Marque una o más opciones
   Selecciona todas las opciones que correspondan.
   - ☐ Organización
   - ☐ Estandarización de documentación
   - ☐ Facilidad de archivo, recuperación y búsqueda de información
   - ☐ Formalización de procesos
   - ☐ Digitalización de datos que a futuro podrían ser utilizados para realizar estadísticas
   - ☐ Ninguno
   - ☐ Otros: .....

5. **¿En qué aspectos lo ayudó a usted en su labor dentro de la institución?** *
   Marque una o más opciones
   Selecciona todas las opciones que correspondan.

   - [ ] Le brindó la posibilidad de trabajar desde el hogar
   - [ ] Eficiencia (Agiliza el trabajo)
   - [ ] Mejor conocimiento y seguimiento de los alumnos
   - [ ] Ninguno
   - [ ] Otros: .....

6. **¿En qué aspectos encuentra que Kimkëlen NO es útil?** *
   Marque una o más opciones
   Selecciona todas las opciones que correspondan.

   - [ ] Realentiza los procesos. La vieja usanza (hacer todo en papel) resultaba más rápido
   - [ ] Le es dificil de utilizar (incómodo, poco intuitivo)
   - [ ] Le provoca miedo utilizarlo (si hago click podré volver hacia atrás? La acción que estoy ejecutando tendrá los resultados que espero?)
   - [ ] Aunque le es útil, no posee todas las funcionalidades que necesita para trabajar
   - [ ] Otros: .....

7. **¿Qué funcionalidades opina que faltan considerar en el sistema?**
   Describa brevemente si encuentra falencias en el sistema

   .....

   .....

   .....

   .....

   .....

8. **¿Le gustaría contar con un acceso al sistema para alumnos?** *
   Marca solo un óvalo.

   - ◯ Sí *Pasa a la pregunta 9.*
   - ◯ No *Pasa a la pregunta 11.*

Sistema de alumnos Kimkëlen (Cont.)

9. **¿Qué ventajas cree que aportaría contar con un acceso para estudiantes?**
   (por ejemplo: ventajas a la comunicación, ahorro de insumos tales como papel o tinta de impresión, etc.)

   .....

   .....

   .....

   .....

   .....

10. **¿Qué funcionalidades cree que se les debería permitir realizar a los alumnos desde el sistema?**
Selecciona todas las opciones que correspondan.

- [ ] Ver sólo calificaciones, asistencias y sanciones propias
- [ ] Integración con entornos virtuales de enseñanza y aprendizaje (como Moodle)
- [ ] Posibilidad de envío de mensajes a preceptores y docentes
- [ ] Ver notificaciones que sean cargadas al sistema por preceptores o autoridades de la institución (por ej, notas sobre asuetos escolares, paros, actos escolares, días de recreación, desinfección, duelo, etc)
- [ ] Imprimir y/o descargar documentos (boletines, analíticos)
- [ ] Otros: .....

Pasa a la pregunta 12.

11. **¿Por qué no le gustaría que la institución cuente con un acceso al sistema para alumnos?**
(Justifique brevemente)

.....

.....

.....

.....

.....

Sistema de alumnos Kimkëlen (Cont.)

12. **¿Le gustaría contar con un acceso al sistema para tutores/padres de los alumnos?** *
Marca solo un óvalo.

- ◯ Sí *Pasa a la pregunta 13.*
- ◯ No *Pasa a la pregunta 15.*

Pasa a la pregunta 16.

Sistema de alumnos Kimkëlen (Cont.)

13. **¿Qué ventajas cree que aportaría el tener un acceso al sistema para tutores?**
(por ejemplo: ventajas en la comunicación padre-hijo y padre-escuela, mejorará el compromiso de los padres, mejorará el rendimiento de los alumnos indirectamente al verse los tutores comprometidos y enterados del progreso de su hijo en la institución, etc.)

.....

.....

.....

.....

.....

14. **¿Qué funcionalidades cree que se les debería permitir realizar a los tutores?**
Elija una o más opciones
Selecciona todas las opciones que correspondan.

- [ ] Ver calificaciones, asistencias y sanciones de su hijo/estudiante a cargo únicamente.
- [ ] Descargar boletines, analíticos y otros documentos de su hijo/estudiante a cargo.
- [ ] Ver notificaciones que sean cargadas al sistema por preceptores o autoridades de la institución (por ej, citaciones, notas sobre asuetos escolares, paros, actos escolares, autorizaciones para paseos educativos, etc).
- [ ] Enviar mensajes a docentes, preceptores, autoridades.
- [ ] Otros: .....

Pasa a la pregunta 16.

Sistema de alumnos Kimkëlen (Cont.)

15. **Justifique por qué cree que NO sería beneficioso un acceso al sistema para tutores.**

.....

.....

.....

.....

.....

Sistema de alumnos Kimkelen (cont.)

16. **¿En dónde se encuentra cuando accede al sistema normalmente?** *
Selecciona todas las opciones que correspondan.

- [ ] En la escuela desde un dispositivo propio.
- [ ] En la escuela, desde un dispositivo de la propiedad de la escuela.
- [ ] Desde su hogar.
- [ ] Otros: .....

17. **¿Le gustaría que el sistema sea accesible desde dispositivos móviles (celulares/tablets)?** *
Elija una o más opciones
Selecciona todas las opciones que correspondan.

- [ ] Sí, le gustaría que todas las funcionalidades sean accesibles en dispositivos móviles.
- [ ] Sí, sería útil pero que se pueda acceder de forma acotada, con un perfil de "Sólo consulta". Así se podría consultar el sistema desde las aulas mismas a través un dispositivo móvil.
- [ ] No le encuentra utilidad.
- [ ] Le da lo mismo.
- [ ] Las aplicaciones para tutores/alumnos deberían ser accesibes desde celulares.

Anexo II. Especificación API Kimkëlen

Versión	Fecha	Autor	Descripción
v1	01.07.2015	Corrons María Emilia	Borrador inicial

Métodos

1. login

Autenticar el usuario contra el sistema.

Requerimiento

Método	URL
POST	/api/v1/login

Tipo	Parámetros	Valores
HEAD POST POST	api_key username password	string string string

api_key
Debe ser enviada en todos los requerimientos del cliente ya que ayuda al servidor a validar la fuente del requerimiento.

username
Nombre del usuario.

password
Contraseña del usuario.

Respuesta

Estado	Respuesta
200	{ "auth_key": <auth_key> }
403	{"error":"API key is missing."}
400	{"error":"Please provide username."}

400	{"error":"Please provide password."}
401	{"error":"Invalid API key."}
401	{"error":"Incorrect username or password."}
500	{"error":"Something went wrong. Please try again later."}

2. obtener estudiante

Recupera la información personal de un estudiante.

Requerimiento

Método	**URL**
GET	/api/v1/students/:id

Tipo	**Parámetros**	**Valores**
HEAD GET	auth_key id	string string

auth_key

El auth_key que fue otorgado en la respuesta de /api/v1/login

id

El id del estudiante del cual se quiere obtener información.

Respuesta

Estado	**Respuesta**

200	La respuesta será un objeto que contenga la información del estudiante. El objeto tendrá la siguiente estructura:
	{
	"student": {
	"id": 3,

	"global_file_number": "14548",
	"person": {
	"firstname": "Milagros",
	"lastname": "ALMADA",
	"identification_type": “DNI”,
	"identification_number": "40489861",
	"phone": "4227843",
	"email": "",
	"address": {
	"street": "4 e/ 524 y 525",
	"number": "780",
	"floor": "",
	"flat": "",
	"city": {
	"name": "La Plata",
	"state": {
	"name": "Buenos Aires",
	"country": {
	"name": "Argentina"
	}
	}
	}
	}
	},
	"_links": [
	{
	"ref": "self",
	"href": "/students/3/image",
	"method": "GET"
	}
	]
	}
	}

403	{"error":"API key is missing."}
400	{"error":"Please provide student id."}
401	{"error":"Invalid API key."}
404	{"error":"Invalid student id."}
500	{"error":"Something went wrong. Please try again later."}

3. obtener tutor

Recupera la información personal de un tutor y los estudiantes que éste tiene a cargo (uno o más).

Requerimiento

Método	URL
GET	/api/v1/tutors/:id

Tipo	Parámetros	Valores
HEAD GET	auth_key id	string string

auth_key

El auth_key que fue otorgado en la respuesta de /api/v1/login

id

El id del tutor del cual se quiere obtener información.

Respuesta

Estado	Respuesta

200	La respuesta será un objeto que contenga la información del tutor. El objeto tendrá la siguiente estructura:
	{
	"tutor": {
	"id": 5,
	"tutor_type": {
	"name": "Padre"
	},
	"person": {
	"firstname": "Daniel Alejandro",
	"lastname": "ALMADA",
	"identification_type": "DNI",
	"identification_number": "40789861",
	"phone": null,
	"email": null,
	"address": {

	"street": "10",
	"number": "2035",
	"floor": "",
	"flat": "",
	"city": {
	"name": "La Plata",
	"state": {
	"name": "Buenos aires",
	"country": {
	"name": "Argentina"
	}
	}
	}
	}
	},
	"students": [
	{
	"id": 3,
	"name": "Milagros, ALMADA",
	"_links": "/students/3"
	},
	{
	"id": 5,
	"name": "Micaela Elizabeth, ALMADA",
	"_links": "/students/5"
	}
	]
	}
	}

403	{"error":"API key is missing."}
400	{"error":"Please provide tutor id."}
401	{"error":"Invalid API key."}
404	{"error":"Invalid tutor id."}
500	{"error":"Something went wrong. Please try again later."}

4. obtener listado de años lectivos

Recupera el listado de años lectivos que están cargados en el sistema.

Requerimiento

Método	URL
GET	/api/v1/school_years

Tipo	Parámetros	Valores
HEAD	auth_key	string

auth_key
El auth_key que fue otorgado en la respuesta de /api/v1/login

Respuesta

Estado	Respuesta
200	[2011, 2012, 2013, 2014]
403	{"error":"API key is missing."}
401	{"error":"Invalid API key."}
500	{"error":"Something went wrong. Please try again later."}

5. obtener calificaciones de un estudiante

Recupera el listado de calificaciones para el año lectivo recibido como parámetro del estudiante que también es recibido como parámetro. El listado se ordena por materia.

Requerimiento

Método	URL
GET	/api/v1/students/:id/marks/:school_year

Tipo	Parámetros	Valores
HEAD GET GET	auth_key id school_year	string number number

auth_key
El auth_key que fue otorgado en la respuesta de /api/v1/login

id
El id del estudiante.

school_year
El año lectivo.

Respuesta

Estado	Respuesta

200	{
	"marks": {
	"school_year": 2012,
	"year": 6,
	"status": "Aprobado",
	"course_subjects": [
	{
	"name": "Inglés - Nivel III",
	"average": 9.33,
	"marks": [
	{
	"mark_number": 1,
	"mark": "9.0",
	"is_free": 0
	},
	{
	"mark_number": 2,

```
                        "mark": "10.0",
                        "is_free": 0
                    },
                    {
                        "mark_number": 3,
                        "mark": "9.0",
                        "is_free": 0
                    }
                ],
                "examinations": [],
                "repproveds": [],
                "final_average": "9.33"
            },

            {
                "name": "6 E Filosofía",
                "average": 2.67,
                "marks": [
                    {
                        "mark_number": 1,
                        "mark": "8.0",
                        "is_free": 0
                    },
                    {
                        "mark_number": 2,
                        "mark": "0.0",
                        "is_free": 1
                    },
                    {
                        "mark_number": 3,
                        "mark": "0.0",
                        "is_free": 1
                    }
                ],
                "examinations": [
                    {
                        "mark": null,
                        "is_absent": 0,
                        "examination_number": 2,
                        "date": null
                    }
                ],
                "repproveds": [
                    {
                        "mark": "9.0",
                        "is_absent": 0,
                        "date": null
                    }
                ],
                "final_average": null
            },
    ...
    ...
```

	... "global_average": null } }
403	{"error":"API key is missing."}
400	{"error":"Please provide student id."}
401	{"error":"Invalid API key."}
404	{"error":"Invalid student id."}
404	{"error":"Invalid school year."}
400	{"error":"Please provide school year."}
500	{"error":"Something went wrong. Please try again later."}

6. obtener inasistencias de un estudiante

Recupera el listado de inasistencias para el año lectivo recibido como parámetro del estudiante que también es recibido como parámetro. El listado se ordena por fecha ascendente.

Requerimiento

Método	**URL**
GET	/api/v1/students/:id/absences/:school_year

Tipo	**Parámetros**	**Valores**
HEAD GET GET	auth_key id school_year	string number number

auth_key
El auth_key que fue otorgado en la respuesta de /api/v1/login

id
El id del estudiante.

school_year
El año lectivo.

Respuesta

Estado	Respuesta

200	<pre>{{
	    "absences": {
	        "school_year": 2012,
	        "absences": [
	            {
	                "absence_type": "1 falta",
	                "value": "1.0",
	                "day": "2012-04-11"
	            },
	            {
	                "absence_type": "1/2 falta",
	                "value": "0.5",
	                "day": "2012-05-02"
	            },
	            …
	            …
	            …
	        ],
	        "total_absences": 21
	    }
	}</pre>

403	{"error":"API key is missing."}
400	{"error":"Please provide student id."}
401	{"error":"Invalid API key."}
404	{"error":"Invalid student id."}
404	{"error":"Invalid school year."}
400	{"error":"Please provide school year."}
500	{"error":"Something went wrong. Please try again later."}

7. obtener sanciones de un estudiante

Recupera el listado de sanciones disciplinarias para el año lectivo recibido como parámetro del estudiante que también es recibido como parámetro. El listado se ordena por fecha ascendente.

Requerimiento

Método	URL
GET	/api/v1/students/:id/disciplinary_sanctions/:school_year

Tipo	Parámetros	Valores
HEAD GET GET	auth_key id school_year	string number number

auth_key
El auth_key que fue otorgado en la respuesta de /api/v1/login

id
El id del estudiante.

school_year
El año lectivo.

Respuesta

Estado	Respuesta

200	{
	"disciplinary_sanctions_for_year": {
	"school_year": 2011,
	"disciplinary_sanctions": [
	{
	"disciplinary_sanction_type": "Molestar en clase",
	"value": 3,
	"request_date": "2011-04-28",
	"resolution_date": null,
	"number": null

	<pre> },
	            {
	                "disciplinary_sanction_type": "Ausentarse en hora de clase",
	                "value": 3,
	                "request_date": "2011-11-07",
	                "resolution_date": null,
	                "number": null
	            }
	        ],
	        "total_disciplinary_sanctions": 2
	    }
	}</pre>

403	{"error":"API key is missing."}
400	{"error":"Please provide student id."}
401	{"error":"Invalid API key."}
404	{"error":"Invalid student id."}
404	{"error":"Invalid school year."}
400	{"error":"Please provide school year."}
500	{"error":"Something went wrong. Please try again later."}

8. obtener años lectivos de un estudiante

Recupera el listado de años lectivos para los cuales el estudiante posee historial académico.

Requerimiento

Método	**URL**
GET	/api/v1/students/:id/school_years

Tipo	**Parámetros**	**Valores**
HEAD GET	auth_key id	string number

auth_key
El auth_key que fue otorgado en la respuesta de /api/v1/login

id
El id del estudiante.

Respuesta

Estado	Respuesta

200	{
	"school_years": [
	{
	"year": 2011
	},
	{
	"year": 2012
	}
	]
	}

403	{"error":"API key is missing."}
400	{"error":"Please provide student id."}
401	{"error":"Invalid API key."}
401	{"error":"Invalid student id."}
500	{"error":"Something went wrong. Please try again later."}

Glosario

Convenciones

- **Cliente** - Aplicación cliente.
- **Estado** - Código HTTP de respuesta.
- Todas las respuestas posibles para cada método están listadas en la sección de Respuestas. Solo una de ellas es devuelta por requerimiento.
- Todas las respuestas están en formato JSON.
- Todos los parámetros son obligatorios excepto los marcados como opcionales `[optional]`.

Códigos de estado

Todos los códigos de estado son estándar HTTP. Los que listan a continuación son los utilizados por esta API.

`2XX` - Éxito.
`4XX` - Error en el cliente.
`5XX` - Error en el servidor.

Código de estado	Descripción
200	OK
201	Creado
202	Aceptado
400	Solicitud incorrecta
401	No autorizado
403	Prohibido
404	No encontrado
500	Error interno