

**UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO**

**CARRERA:
INGENIERÍA DE SISTEMAS**

**Trabajo de titulación previo a la obtención del título de:
Ingenieros de Sistemas**

**TEMA:
SISTEMA WEB Y APLICACIÓN ANDROID PARA LA GESTIÓN DE LAS
ACTIVIDADES DE VOLUNTARIADO DE LA UNIVERSIDAD
POLITÉCNICA SALESIANA, SEDE QUITO.**


**AUTORES:
RODDY VINICIO GAHONA DELGADO
DIEGO FERNANDO SALGUERO TORRES**

**TUTOR:
FRANKLIN EDMUNDO HURTADO LARREA**

Quito, noviembre del 2017

CESIÓN DE DERECHOS DE AUTOR

Nosotros, RODDY VINICIO GAHONA DELGADO, con documento de identificación N° 1500844905, y DIEGO FERNANDO SALGUERO TORRES, con documento de identificación N° 1718978966, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación con el tema: “SISTEMA WEB Y APLICACIÓN ANDROID PARA LA GESTIÓN DE LAS ACTIVIDADES DE VOLUNTARIADO DE LA UNIVERSIDAD POLITÉCNICA SALESIANA, SEDE QUITO”, mismo que ha sido desarrollado para optar por el título de INGENIEROS DE SISTEMAS en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente. En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.



.....
RODDY VINICIO
GAHONA DELGADO
CI: 1500844905



.....
DIEGO FERNANDO
SALGUERO TORRES
CI: 1718978966

Quito, noviembre del 2017

DECLARATORIA DE COAUTORÍA DEL TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el Proyecto Técnico, con el tema: “SISTEMA WEB Y APLICACIÓN ANDROID PARA LA GESTIÓN DE LAS ACTIVIDADES DE VOLUNTARIADO DE LA UNIVERSIDAD POLITÉCNICA SALESIANA, SEDE QUITO” realizado por los estudiantes RODDY VINICIO GAHONA DELGADO y DIEGO FERNANDO SALGUERO TORRES, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana, para ser considerados como trabajo final de titulación.

Quito, noviembre del 2017



.....
FRANKLIN EDMUNDO HURTADO LARREA

CI: 1713382016

Dedicatoria

Se lo dedico a mis padres, y en general a mi corta familia, quienes incondicionalmente han estado en cada uno de los momentos previos a la entrega de este trabajo, son ellos mi fuerza y el motor que me impulsa para seguir adelante, y no dejarme vencer por la adversidad. Este trabajo es un logro más, que en parte les pertenece por no dejar que me rinda jamás, y darme el ejemplo para poder convertirme en un gran ser humano.

Roddy Vinicio Gahona Delgado

Se lo dedico a mis padres y a mi hermana quienes a lo largo de este camino me han apoyado incondicionalmente, gracias a sus consejos que me brindaron día a día consiguieron que yo nunca me rindiera y continúe adelante hasta lograr el objetivo que me propuse hace algunos años atrás, gracias a ellos he conseguido convertirme en la persona que soy ahora.

Diego Fernando Salguero Torres

Agradecimiento

Nosotros agradecemos a la Universidad Politécnica Salesiana que ha contribuido en nuestra formación profesional, a la Dirección Técnica de Pastoral por brindarnos las facilidades necesarias para realizar nuestro proyecto de titulación, a nuestro tutor el Ingeniero Franklin Hurtado Larrea por guiarnos y apoyarnos durante todo el desarrollo del trabajo.

Roddy Vinicio Gahona Delgado

Diego Fernando Salguero Torres

ÍNDICE

INTRODUCCIÓN	1
Antecedentes	1
Problema	2
Justificación.....	3
Objetivo General.....	5
Objetivos Específicos.....	5
Alcance del proyecto.....	5
Marco Metodológico	7
Scrum.	7
Roles utilizados.....	7
Sprints.	8
Reuniones.....	8
Prototipos.	9
Capítulo 1.....	10
1.1 Voluntariado.....	10
1.1.1 Seguimiento.	10
1.1.2 Verificación.....	10
1.1.3 Validación.	11
1.1.4 Registro de asistencia.....	11
1.2 Metodología	12
1.3 Ingeniería de software	12
1.4 Proyectos de software	13
1.4.1 Desarrollo adaptativo.....	13
1.4.1.1 Scrum.	14
1.4.2 Desarrollo ágil.....	15
1.4.3 Especificación requerimientos.....	17
1.4.4 Casos de uso.....	18
1.4.4.1 Include (Casos de Uso UML).	19
1.4.5 Diagrama de proceso AS IS.....	19
1.4.3 Diagrama de proceso TO BE.....	20
1.5 Arquitectura.....	20
1.5.1 Arquitectura cliente.....	20
1.5.2 Arquitectura servidor.....	21
1.5.3 MVC.....	21
1.5.4 MVP.....	22
1.5.5 Api rest.....	23
1.6 Lenguajes de programación	23
1.6.1 Java.....	24
1.6.2 JavaScript (JS).....	24
1.6.3 PHP.....	25
1.6.4 JSON.....	25
1.6.5 XML.....	26
1.7 Herramientas	26
1.7.1 Sublime Text 3.....	26
1.7.2 Apache 2.....	27
1.7.3 PhpMyAdmin.....	27
1.7.4 Android Studio.....	28
1.7.5 Firebase.....	28

1.7.6 JMeter.....	29
1.8 Prototipo.....	29
1.9 Pruebas de caja negra	30
Capítulo 2.....	31
2.1 Análisis del proceso	31
2.1.1 Proceso manual (AS IS).	31
2.1.2 Proceso optimizado con software (TO BE).	33
2.2 Especificación de requerimientos	34
2.3 Casos de uso.....	35
2.3.1 Caso de uso perfil estudiante sistema web.....	36
2.3.2 Caso de uso perfil administrativo sistema web.	37
2.3.3 Caso de uso perfil administrador sistema web.	38
2.3.4 Caso de uso perfil estudiante aplicación móvil.....	40
2.4 Diagrama de clases.....	41
2.4.1 Diagrama de clases sistema web.	41
2.4.2 Diagrama de clases aplicación móvil.	42
2.5 Diagrama de base de datos	42
2.6 Diseño de interfaces	46
Capítulo 3.....	47
3.1 Arquitectura sistema web	47
3.2 Estándares de programación sistema web.....	48
3.2.1 Estándares Front-End.	48
3.2.2 Estándares Back-End	49
3.3 Listado de componentes sistema web	49
3.4 Código relevante sistema web.....	50
3.4.1 Modelo genérico Front-End.	50
3.4.2 Módulo general Font-End.	52
3.4.3 Notificaciones dispositivo móvil Back-End.	53
3.5 Arquitectura aplicación móvil.....	54
3.6 Estándares de programación aplicación móvil.....	58
3.6.1 Programación Android.	58
3.6.2 Programación de servicios web.....	58
3.6.3 Base de datos.....	58
3.7 Listado de componentes aplicación móvil	59
3.8 Código relevante aplicación móvil.....	59
3.8.1 AndroidManifests.xml.	59
3.8.2 Google-services.json.	62
3.8.3 Estructura de envío de datos con post (asincrónico).....	63
3.8.4 Estructura básica para la vista de instituciones UPS.....	65
3.9 Pruebas de la solución.....	67
3.9.1 Plan de pruebas.	67
3.9.1.1 Pruebas funcionales.....	67
3.9.1.2 Pruebas de rendimiento sistema web.	78
3.9.1.3 Pruebas de rendimiento aplicación móvil.	82
3.9.1.4 Pruebas de entendibilidad sistema web.....	89
3.9.1.5 Pruebas de entendibilidad aplicación móvil.....	91
CONCLUSIONES.....	94
RECOMENDACIONES.....	95
GLOSARIO DE TÉRMINOS.....	96
LISTA DE REFERENCIAS	97

ÍNDICE DE TABLAS

Tabla 1. Tablas de la base de datos versión final	43
Tabla 2. Resumen de pruebas funcionales	68
Tabla 3. Resultados generales de las peticiones get inscripción	79
Tabla 4. Resultados de las peticiones post inscripción	80
Tabla 5. Resultados de las peticiones get postulados.....	80
Tabla 6. Resultados de las peticiones get reportes	81
Tabla 7. Características Samsung Note 5.....	82
Tabla 8. Características Nexus 5X.....	83
Tabla 9. Características Pixel.....	83

ÍNDICE DE FIGURAS

Figura 1. Módulos del sistema web.	6
Figura 2. Módulos de la aplicación móvil.....	7
Figura 3. Diagrama proceso normal versión final.....	31
Figura 4. Diagrama proceso mejorado versión final.....	33
Figura 5. Caso de uso perfil estudiante sistema web versión final.	36
Figura 6. Caso de uso perfil administrativo sistema web versión final.	37
Figura 7. Caso de uso perfil administrador para configuraciones sistema web versión final.	38
Figura 8. Caso de uso perfil administrador para registro y seguimiento sistema web versión final.....	39
Figura 9. Caso de uso perfil administrador para reportes y perfil sistema web versión final.	40
Figura 10. Caso de uso perfil estudiante aplicación móvil versión final.	40
Figura 11. Diagrama de clases sistema web versión final.	41
Figura 12. Diagrama de clases aplicación móvil versión final.	42
Figura 13. Diagrama de la base de datos versión final.	45
Figura 14. Diseño de interfaces.....	46
Figura 15. Diseño arquitectónico sistema web.	47
Figura 16. Código modelo genérico parte 1 sistema web.	50
Figura 17. Código modelo genérico parte 2 sistema web.	51
Figura 18. Código módulo general sistema web.....	52
Figura 19. Código individual_model sistema web.....	53
Figura 20. Diagrama de la arquitectura de la aplicación móvil.	55
Figura 21. Diagrama de la arquitectura aplicación móvil con firebase.....	56
Figura 22. Código AndroidManifest.xml con la configuración de los servicios parte 1.....	60
Figura 23. Código AndroidManifest.xml con su configuración parte 2.	61
Figura 24. Código Google-services.json.....	62
Figura 25. Código con datos enviados desde la aplicación móvil.	64
Figura 26. Código con datos enviados desde la aplicación móvil.	65
Figura 27. Fragmento de código del adaptador.....	66
Figura 28. Tiempo de respuesta petición get inscripción.....	78
Figura 29. Tiempo de respuesta para la petición post inscripción.	79
Figura 30. Tiempo de respuesta petición get estudiantes postulados.....	80
Figura 31. Gráfico que muestra el tiempo de respuesta petición get reporte general.....	81
Figura 32. Prueba 1 de rendimiento aplicación móvil.	83
Figura 33. Prueba 2 de rendimiento aplicación móvil.	84
Figura 34. Prueba 3 de rendimiento aplicación móvil.	84
Figura 35. Prueba 4 de rendimiento aplicación móvil.	85
Figura 36. Prueba 5 de rendimiento aplicación móvil.	85
Figura 37. Prueba 6 de rendimiento aplicación móvil.	86
Figura 38. Prueba 7 de rendimiento aplicación móvil.	87
Figura 39. Gráfico prueba 8 de rendimiento aplicación móvil.	87
Figura 40. Prueba 9 de rendimiento aplicación móvil.	88
Figura 41. Resultado de la pregunta 1 del sistema web.	89

Figura 42. Resultado de la pregunta 2 del sistema web.	89
Figura 43. Resultado de la pregunta 3 del sistema web.	90
Figura 44. Resultado de la pregunta 4 del sistema web.	90
Figura 45. Resultado de la pregunta 1 de la aplicación móvil.	91
Figura 46. Resultado de la pregunta 2 de la aplicación móvil.	92
Figura 47. Resultado de la pregunta 3 de la aplicación móvil.	92
Figura 48. Resultado de la pregunta 4 de la aplicación móvil.	93

Resumen

Con el desarrollo del proyecto, cuyo detalle se describe en el presente documento, se ha buscado aplicar todos los conocimientos adquiridos en la carrera de Ingeniería de Sistemas, desarrollando un sistema web y una aplicación móvil que le sirva a la Dirección Técnica de Pastoral sede Quito como herramientas de automatización para el Voluntariado de Acción Social, permitiéndoles contar con dos plataformas en las que se pueda acceder desde cualquier lugar y en cualquier momento, facilitando el proceso de inscripción y seguimiento de los estudiantes para cada periodo académico. Por otro lado, la Dirección Técnica de Pastoral contará con valiosa información en tiempo real que facilite la comunicación entre todos los campus de la sede Quito.

Abstract

With the development of the project, the detail of which is described in this document, it has been sought to apply all the knowledge acquired in the engineering career of systems, developing a web system and a mobile application that serves the technical direction of Pastoral headquarters Quito as automation tools for Social action volunteering, allowing them to have two platforms that can be accessed from anywhere and at any time, facilitating the process of registration and monitoring of Students for each academic period. On the other hand, the Technical Directorate of Pastoral will have valuable information in real time that facilitates the communication between all the campuses of the Quito headquarters.

INTRODUCCIÓN

En el presente documento se describen todas aquellas herramientas, técnicas, metodologías y conocimientos aplicados para la realización del proyecto que tiene como propósito el desarrollo de un sistema web y una aplicación móvil que funciona sobre el sistema operativo Android (en el texto donde se cite aplicación móvil se refiere a una aplicación móvil Android). Sin duda, muchos de los conocimientos y experiencias adquiridas durante el proceso de formación en la carrera de Ingeniería de Sistemas fueron clave para llevar a cabo con éxito este proyecto y dar solución al problema presentado con el VAS.

En el capítulo uno “Marco teórico” se describen los principales conceptos que sustentan el proyecto, en el capítulo dos “Análisis y diseño” se analizan los diferentes componentes con los que cuenta el software y en el capítulo tres “Construcción y pruebas” se abarca como se construye el sistema web y la aplicación móvil y las pruebas realizadas a cada uno de estos.

Antecedentes

La UPS con su misión de formar “honrados ciudadanos y buenos cristianos” (Universidad Politécnica Salesiana, s.f.) posee un proyecto de voluntariado que, aproximadamente, beneficia a 60 instituciones (sin fines de lucro), que ofrecen un servicio a la comunidad más vulnerable como: “Los adultos mayores, niñas, niños y adolescentes, mujeres embarazadas, personas con discapacidad, gente privada de su libertad y quienes adolezcan de enfermedades catastróficas o de alta complejidad, recibirán atención prioritaria y especializada en los ámbitos público y privado, dentro de la nueva Constitución.” (Pardo, 2008).

Problema

La UPS brinda a la comunidad el apoyo mediante los jóvenes estudiantes que desean realizar esta actividad que tiene como recompensa cuantificada 10 puntos a su nota final en la materia de Pensamiento Social de la Iglesia. Los estudiantes de la UPS, en la sede Quito, cuentan con la posibilidad de adquirir el puntaje mencionado mediante la realización del VAS en cualquiera de las instituciones con las que la universidad ha celebrado convenios o en instituciones de su comunidad.

El proceso inicia cuando el estudiante se acerca a una de las oficinas de la Dirección Técnica de Pastoral de su campus respectivo y de manera verbal, expresa que desea realizar el proceso VAS para la materia de Pensamiento Social de la Iglesia, el estudiante puede escoger una de las instituciones con las con las que la universidad tenga un acuerdo o una intuición de su comunidad, este proceso se lo realiza con hojas impresas de listas de los cursos para poder validar que el estudiante esté inscrito en la materia, las listas no tienen ningún tipo de validación y pagos del estudiante puesto que son listas que las adquieren previo a las verificaciones que se hace con el pago de matrícula. Los datos son ingresados de manera manual en hojas de Excel que no cuentan con ningún tipo de seguridad, respaldo, ni tampoco con su respectiva replicación a los otros campus.

Por otro lado, la Dirección Técnica de Pastoral mantiene el seguimiento a las actividades realizadas en el VAS por los estudiantes con una estrategia para realizar la validación de que los estudiantes estén acudiendo a la institución, se rigen mediante el número de estudiantes que tengan asignados:

Más de 3 estudiantes: visita por parte de pastoral.

Menos de 2 estudiantes: llamada telefónica a la institución.

Los estudiantes poseen una hoja impresa en la cual deben colocar la actividad realizada y se lo valida con la firma de la persona encargada en la institución y una firma del estudiante, esta hoja es el comprobante que posee el estudiante para la validación de que el VAS se realizó de la mejor manera, a esta hoja se adjunta a la documentación impresa que dispone el estudiante y es entregada en la Dirección Técnica de Pastoral.

Como ya se ha mencionado, este documento no posee ningún tipo de seguridad ni respaldo, la documentación impresa genera mucho papeleo que se vuelve muy burocrático para el personal de la Dirección Técnica de Pastoral, al no contar con información replicada en tiempo real los demás campus no cuentan con una manera óptima de visualizar vacantes disponibles, estudiantes asignados, ni ningún tipo de información oportuna que sirva para mejorar el proceso de inscripción y seguimiento en el VAS. Por otro lado, los estudiantes no cuentan con ningún tipo de respaldo en caso de que las hojas impresas puedan degradarse o perderse y esto genera que cuando se suscita este tipo de inconvenientes exista malestar por parte de los estudiantes y también de la parte operativa de la Dirección Técnica de Pastoral.

Justificación

La Dirección Técnica de Pastoral al gestionar su información en hojas impresas, sin ningún tipo de respaldo digital que pueda garantizar una acción de contingencia, en caso de que algún inconveniente se suscite con la información de estudiantes o instituciones (sin fines de lucro), se ve en la necesidad de contar con un sistema que, además de facilitar el ingreso de la información, permita contar en tiempo real con información actualizada en todos los campus de la sede Quito de la UPS.

La Dirección Técnica de Pastoral no cuenta con un medio moderno y eficiente para publicar las vacantes disponibles en cada una de las instituciones que brindan el servicio a la comunidad y donde los estudiantes de la UPS pueden realizar el VAS, es

por ello que se ve la necesidad de implementar una sistema web donde los colaboradores de la Dirección Técnica de Pastoral de cada campus, puedan administrar y gestionar dichas vacantes, y una aplicación móvil para los estudiantes, donde puedan consultar lugares, cantidad de vacantes y tipo de actividades que van a desempeñar en la institución.

El automatizar los procesos de una entidad puede resultar en un incremento en la eficiencia y el servicio que reciben los usuarios, las TI pueden generar tiempos de respuesta significativamente pequeños a los tiempos que se disponga con el manejo manual de los datos. Otra de las cualidades de las plataformas tecnológicas, específicamente plataformas web, es contar con un nivel mayor de seguridad en el manejo de la información con respecto al modo actual de gestionarla, y la posibilidad de contar con replicación en los distintos lugares donde se la requiera. “Las Tecnologías de la Información, son una excelente herramienta de gestión empresarial, que ayuda positivamente para el desarrollo y viabilidad de las organizaciones.” (REYES, 2013).

Contar con un sistema con las características que se presentan, normalmente, genera las siguientes ventajas:

Mayor Visibilidad.

Accesibilidad las 24 horas del día.

Presentar una imagen profesional y creíble.

Promover sus servicios.

La tecnología requerida para poder ejecutar este proyecto, por un lado, las plataformas web que por su versatilidad de usos en distintas ubicaciones geográficas son idóneas para tipo de proyecto a emprender, y por otro lado el uso de aplicaciones móviles orientadas para dispositivos con sistema operativo Android que por la popularidad de

este sistema operativo y por la gran cantidad de dispositivos que cuentan con dicho sistema puede convertirse en la forma más óptima para el acercamiento con los usuarios de dispositivos inteligentes como smartphone. Estas tecnologías están en vigencia, por lo tanto, se coincidiera que tecnológicamente el proyecto es viable.

El sistema puede, a futuro, ser aplicado en otras sedes en las que opera la UPS, y con esto lograr que el proceso VAS sea más eficiente, seguro y con respuesta más oportuna, mejorando la experiencia de usuarios en cuanto al proceso, y a su vez también facilitando el manejo de la información a los colaboradores de la Dirección Técnica de Pastoral.

Objetivo General.

Desarrollar un sistema web y una aplicación Android para la gestión de las actividades del voluntariado (VAS) de los estudiantes de la Universidad Politécnica Salesiana.

Objetivos Específicos.

Desarrollar una aplicación Android que facilite la consulta de vacantes en las distintas instituciones (sin fines de lucro) que brinden un servicio a la comunidad más vulnerable, y de igual manera facilite el registro de asistencia de los estudiantes.

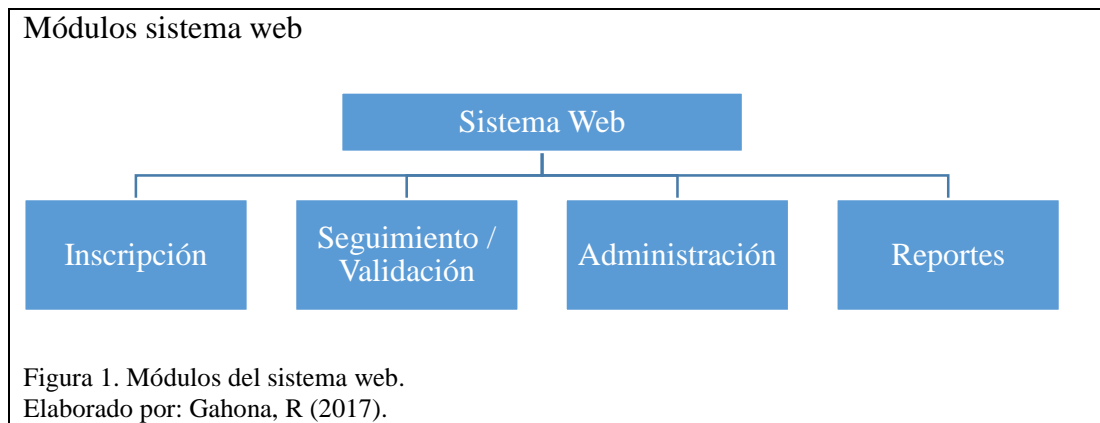
Crear un sistema web para la gestión de los procesos de inscripción y seguimiento de las actividades del voluntariado que realizan los estudiantes de la UPS.

Hacer un seguimiento continuo de las actividades del voluntariado a fin de verificar su cumplimiento.

Alcance del proyecto

Desarrollar un sistema web para el uso del personal de pastoral, y una aplicación móvil para los estudiantes.

El sistema web consta de los siguientes módulos:



Desglose de usuarios del sistema:

Estudiante realiza las siguientes tareas:

Visualizar las vacantes ofertadas por parte de pastoral de las instituciones del voluntariado.

Llevar acabo el registro de actividad en el voluntariado.

El colaborador de Dirección Técnica de Pastoral realiza las siguientes tareas:

Informar sobre las vacantes generadas con las instituciones.

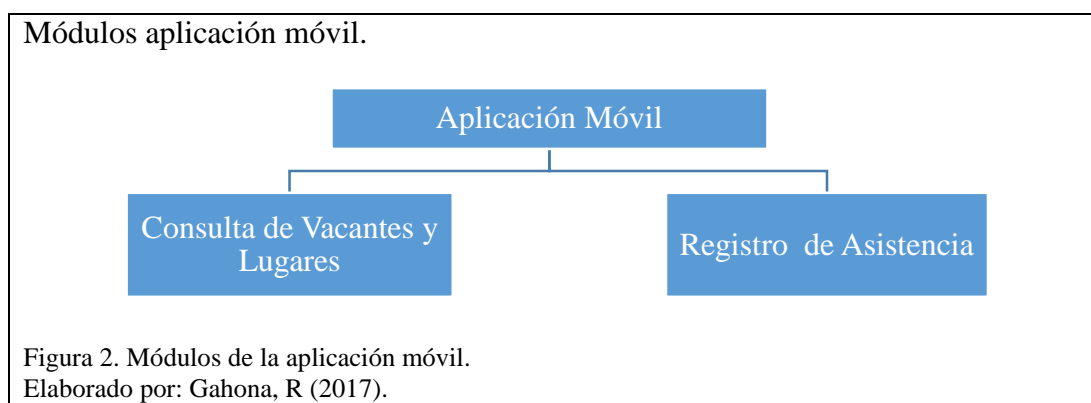
Ingresar y validar información de los estudiantes que apliquen a realizar el voluntariado.

Ingresar la información de nuevas instituciones al proyecto de Voluntariado de Acción Social.

Gestionar la documentación de los estudiantes que realicen el voluntariado.

Generar reportes.

La aplicación móvil consta de los siguientes módulos:



El estudiante realiza las siguientes tareas:

Visualización de vacantes ofertadas por parte de la Dirección Técnica de Pastoral o de las instituciones del voluntariado.

Registrar la actividad en el voluntariado.

Marco Metodológico

En las siguientes líneas se detalla cómo se utiliza la metodología de desarrollo adaptativo de software, de manera iterativa e incremental, a través de sprints a lo largo del proyecto tanto del sistema web como el de la aplicación móvil.

Scrum.

En vista de que el cliente (llamado así por temas didácticos) deseaba llevar un continuo control sobre los avances del proyecto, usando Scrum como metodología de desarrollo de software a lo largo del proyecto la cual permite adoptar una estrategia de desarrollo incremental encubriendo así diferentes fases del desarrollo, para no realizar una tras otra en un ciclo secuencial o en cascada.

Roles utilizados.

Product owner: Al inicio del proyecto se designa a una colaboradora de la Dirección Técnica de Pastoral para que desempeñe este papel y proporcione toda la información necesaria para identificar cada requerimiento funcional del sistema web y de la

aplicación móvil y adicionalmente ser la persona encargada de la aprobación de cada uno de los sprints completados.

Equipo Scrum: El equipo fue conformado por los integrantes del proyecto de titulación.

Scrum master: El papel de Scrum master fue designado a cada integrante del equipo Scrum de manera rotativa en cada sprint, para que se pueda proporcionar a cada uno la oportunidad para planificar actividades del proyecto.

Sprints.

Se establecieron 2 horas de trabajo diarias de lunes a viernes y 8 horas los fines de semana para el proyecto teniendo como resultado 5 sprints:

Sprint1: Módulo de Inscripción y Consulta de vacantes.

Sprint2: Módulo de Administración y Postulación.

Sprint3: Módulo de Seguimiento / Validación y Registro de Actividades.

Sprint4: Módulo de Reportes.

Sprint5: Pruebas Finales.

Reuniones.

Cada levantamiento de los requerimientos se lleva a cabo con las personas en la Dirección Técnica de Pastoral, a través de reuniones conjuntas en el transcurso de cada sprint, se pudieron identificar todas las necesidades y problemas que el cliente poseía, de tal forma que, tanto para el cliente como para los desarrolladores, los requerimientos y las funcionalidades solicitadas estén completamente, las reuniones se realizaron cada 15 días dejando así, al finalizar cada reunión, un documento de especificación de requerimientos que se debían cumplir al finalizar cada sprint, de tal forma que el cliente se encuentre satisfecho con los requerimientos solicitados.

Prototipos.

Se desarrollaron los prototipos del sistema web y la aplicación móvil, los cuales fueron presentando al inicio del proyecto con el objetivo de proporcionar al Product owner una visión de lo que se tenía en mente para lograr que se plasmen sus ideas y para que se aprueben las interfaces que se pensaron una vez obtenido el objetivo del proyecto. Estos prototipos se utilizaron al inicio del proyecto puesto que después de cada sprint, se finalizaba con una parte funcional, y en la mayoría de los incrementos cambiaba completamente a los prototipos iniciales.

Capítulo 1

1.1 Voluntariado

El voluntariado es la oportunidad que poseen los jóvenes al entregar un poco de su tiempo de servicio a la misión salesiana. Esta experiencia es muy relevante, ya que los jóvenes se sumergen en la vida comunitaria y junto a los salesianos trabajan por los más pobres y necesitados (Montenegro, 2016).

Los jóvenes experimentan un cambio de vida ya que pasan de ser estudiantes a ser colaboradores de la comunidad y ya no a preocuparse únicamente de sus necesidades sino también a ver las necesidades de sus destinatarios y entender que la realidad es subjetiva al lugar. “El voluntariado es una manera de demostrar la solidaridad organizada, una intervención de la comunidad en la resolución de sus problemas. La acción voluntaria expresa la responsabilidad e implicación de los ciudadanos en el desarrollo comunitario, en definitiva, el voluntariado es una herramienta para ejercer el derecho a participar y transformar la realidad que nos rodea” (Diputacion Foral de Bizkaia, 2006).

1.1.1 Seguimiento.

“El proceso de seguimiento académico establece la definición de los resultados de las actividades en términos de asistencia de los estudiantes, calificaciones, pertinencia de la formación recibida, promoción y recuperación de problemas de aprendizaje” (PEI INSTITUCION EDUCATIVA LA LEONA, s.f.).

1.1.2 Verificación.

La verificación suele ser el proceso que se realiza para revisar si una determinada cosa está cumpliendo con los requisitos y normas previstos (Merino, 2014).

1.1.3 Validación.

Es la acción y efecto de validar (convertir algo en válido, darle fuerza o firmeza). El adjetivo válido, por otra parte, hace referencia a aquello que tiene un peso legal o que es rígido y subsistente (Gardey, 2013).

1.1.4 Registro de asistencia.

Dentro del proceso de aprendizaje de los estudiantes, se puede aseverar que al pedir que se brinde algo de retroalimentación de la información, experiencia o práctica mostrada, es un mejor indicador de que el estudiante estuvo presente en el proceso. Cabe indicar que a pesar de citar cursos en línea y que el proyecto no sea de esa índole, la información presentada respalda las palabras y los conceptos aplicados a este documento. “Los cursos online que se imparten utilizando plataformas formativas cuentan con herramientas que permiten hacer un seguimiento automático de las conexiones y participación diaria de los alumnos. El cuadro permite comprobar que alumnos han entrado en el aula cada día. Para conocer con más detalle qué actividades ha consultado o incluso realizado el alumno se puede pinchar sobre el nombre del alumno” (Pavón & Pérez, 2003). Un concepto que valida el utilizar para el proyecto en cuestión el registro de experiencia como un registro de asistencia, es el utilizar este apartado para obtener una clara muestra que el estudiante fue y realizó su voluntariado, puesto que comparte datos relevantes únicos para él. “Control de Clases Es un proceso académico que consta de las siguientes actividades dentro del desarrollo de las clases: a. Verificar el estado actual de las sesiones de clase desarrolladas con lo planificado b. Verificar las evaluaciones programadas en el curso en desarrollo c. Verificar la asistencia de los alumnos a las sesiones de clase” (Lisset, 2017). Se procura la automatización del proceso de registro de experiencia para generar una forma útil óptima y fácil de acceder para que la información suministrada sea lo más legible

posible para los entes de control que tiene este proceso. “Frente a esta problemática, se automatizaron los diversos procesos de registro de evaluación y asistencia, diseñando e implementando una solución informática; y con la dirección adecuada de las TIC’s que posee la institución educativa, se mejoró, hasta cierto punto, la eficacia al tener información clasificada y oportuna para una mejor toma de decisiones, superando de esta manera en gran medida las consecuencias negativas mencionadas líneas arriba, haciendo de lado la forma manual de hacer las cosas, como era en un principio” (Cordova, 2016).

1.2 Metodología

Una metodología es considerada como el uso de buenas prácticas para la ejecución de un proyecto, la metodología otorga un marco de trabajo específico, que ayuda a la ejecución de tareas desglosadas de las fases dentro del ciclo de vida de un proyecto, “El término metodología se define como el grupo de mecanismos o procedimientos racionales, empleados para el logro de un objetivo, o serie de objetivos que dirige una investigación científica. Este término se encuentra vinculado directamente con la ciencia, sin embargo, la metodología puede presentarse en otras áreas como la educativa, en donde se encuentra la metodología didáctica o la jurídica en el derecho” (CONCEPTODEFINICION.DE, s.f.).

1.3 Ingeniería de software

La ingeniería de software es la ciencia aplicada que se encarga de gestionar todas las actividades realizadas dentro del ciclo de vida de un proyecto de software, desde la recopilación de la información necesaria para los requerimientos, hasta su mantenimiento post implementación. “Ingeniería de software es la aplicación práctica del conocimiento científico al diseño y construcción de programas de computadora y a la documentación asociada requerida para desarrollar, operar y mantenerlos. Se

conoce también como desarrollo de software o producción de software” (TechNet, 2012), también se puede decir que “Ingeniería de software trata del establecimiento de los principios y métodos de la ingeniería a fin de obtener software de modo rentable, que sea fiable y trabaje en máquinas reales” (Bauer, s.f.).

1.4 Proyectos de software

Es el conjunto de actividades que se realizan para transferir la información de un proceso manual a un modelo automatizado con la ayuda de herramientas informáticas. Los proyectos de software como todo proyecto deben contar con una planificación tanto de recursos como de tiempos de entrega. “Es el Proceso de gestión para la creación de un Sistema o software, la cual encierra un conjunto de actividades, una de las cuales es la estimación, estimar es echar un vistazo al futuro y aceptamos resignados cierto grado de incertidumbre. Aunque la estimación, es más un arte que una Ciencia, es una actividad importante que no debe llevarse a cabo de forma descuidada” (Universitario, s.f.). Todos los proyectos deben contar con un inicio y un fin. “Un proyecto es una asociación de esfuerzos, limitado en el tiempo, con un objetivo definido, que requiere del acuerdo de un conjunto de especialidades y recursos. También puede definirse como una organización temporal con el fin de lograr un propósito específico. Cuando los objetivos de un proyecto son alcanzados se entiende que el proyecto está completo.” (González).

1.4.1 Desarrollo adaptativo.

“El desarrollo adaptativo de software (DAS) lo propuso Jim Highsmith 1998 como una técnica para construir software y sistemas complejos. Los apoyos filosóficos del DAS se enfocan en la colaboración humana y la organización propia del equipo” (Flores, 2016).

El desarrollo de software adaptativo es un concepto que ayuda que se estructure el proyecto en forma iterativa e incremental, pudiendo abordar proyectos de cualquier envergadura, también cabe mencionar que fomenta el uso de prototipos, (DAS) resalta la utilización de un método de construcción de software tolerante a fallos, esto se debe a que el software no es tangible y está sujeto a varios cambios dentro de todo su ciclo de vida. “El desarrollo de software adaptable (ASD) es una metodología de desarrollo que hace énfasis en aplicar las ideas que se originaron en el mundo de los sistemas complejos, adaptación continua del proceso al trabajo” (Wikispace, s.f.).

Sus principales características son:

Iterativo, (Se aplican todas las fases del análisis del desarrollo en cada una de las repeticiones “iteraciones”).

Orientado a los componentes de software más que a las tareas en las que se va a alcanzar dicho objetivo (se centra en los procesos y no en los resultados).

Tolerante a los cambios (flexible a cambios inesperados por su cualidad de iterar “repetir” cada una de las fases).

Guiado por los riesgos (Se analizan riesgos constantemente durante el desarrollo).

La revisión de los componentes sirve para aprender de los errores y volver a iniciar el ciclo de desarrollo (los errores sirven de retroalimentación para nuevas iteraciones).

1.4.1.1 Scrum.

Es un proceso de la metodología de desarrollo ágil para proyectos de software, que brinda ciertas cualidades de gran ayuda para los desarrolladores, entre sus principales bondades se resaltan la mitigación de los riesgos por el gran nivel de involucramiento que se mantiene con el cliente final. “Scrum es un marco de trabajo que nos permite

encontrar prácticas emergentes en dominios complejos, como la gestión de proyectos de innovación. No es un proceso completo, y mucho menos, una metodología” (Alaimo, 2013). Es recomendable utilizar este conjunto de buenas prácticas siempre y cuando se pueda tener una visión global del sistema a construir y además de ello se pueda fragmentar el sistema global en módulos para el desarrollo, “Scrum se basa en iteraciones de tiempo fijo. Puedes elegir la duración de la iteración, pero la idea general es mantener la misma longitud de la iteración durante un período de tiempo, determinando así una cadencia” (Skarin, 2010). Scrum coloca roles a los involucrados de un proyecto, a continuación, se cita su nombre y su principal funcionalidad.

Product owner: es la persona responsable del éxito de producto desde el punto de vista de los involucrados (stakeholders).

Equipo de Desarrollo: El equipo de desarrollo está formado por todos los individuos necesarios para la construcción del producto en cuestión. Es el único responsable por la construcción y calidad del producto.

Scrum master: Es el Coach del equipo y es quien puede ayudar a alcanzar su máximo nivel de productividad posible. Es un facilitador, líder del proyecto, persona encargada de motivar al equipo.

1.4.2 Desarrollo ágil.

“Desarrollo ágil de software, una tendencia en alza en la que el desarrollo iterativo e incremental se impone a los trámites habituales en esta industria. O, dicho de otro modo, una metodología en la que el desarrollador va adaptando sus soluciones a unos requisitos también cambiantes a lo largo del tiempo” (Fraga, 2016).

El desarrollo ágil del software es un conjunto de buenas prácticas que consisten en cambiar la ideología de un desarrollo predictivo y secuencial que implantaba como

única estructura de ejecución de un proyecto, y cambiar la forma en la que se desarrollaban los proyectos de software en especial, implantando un desarrollo iterativo e incremental, que facilita el retomar puntos críticos de un proyecto y a su vez realizar una retroalimentación al aplicar todas las disciplinas del desarrollo de software al estado actual del proyecto en cuestión.

Para que una metodología de desarrollo de software se pueda considerar como ágil debe cumplir con cuatro disposiciones indispensables que están en el Manifiesto Ágil:

Los individuos e interacciones, por encima de los procesos y las herramientas.

Software en funcionamiento, frente a la documentación exhaustiva.

La colaboración con el cliente sobre la negociación contractual.

Respuesta al cambio, mejor que el seguimiento de un plan.

Explicando un poco cada uno de los manifiestos se puede obtener los siguientes conceptos:

Las metodologías de desarrollo ágil, ayudan a integrar al equipo de trabajo en el proyecto sin tanta burocracia ni protocolos que puedan quitar tiempo valioso de desarrollo.

Promocionan la utilización de prototipos funcionales para las presentaciones con los clientes en lugar de documentación extensa que la mayoría de veces es confusa para clientes sin conocimiento técnico.

Las metodologías de desarrollo ágil involucran al cliente de una forma directa y constante dentro de todo el ciclo de vida del proyecto, mitigando la aparición de requerimientos críticos que puedan aparecer en vísperas de que el sistema esté en producción.

Con las iteraciones y el constante involucramiento del cliente el mismo puede obtener una métrica visible de satisfacción con el producto a entregar, las metodologías ágiles están sujetas a cambios dentro de la planificación inicial de un proyecto de software, lo que ayuda a retomar caminos de ejecución durante todo el ciclo de vida del proyecto.

1.4.3 Especificación requerimientos.

Los requerimientos de un sistema describen los servicios (funcionalidades) que ha de ofrecer el sistema y las restricciones asociadas a su desempeño dentro del entorno donde se lo vaya a implementar. Los requerimientos de un proyecto de software están divididos en dos grandes grupos, el primer grupo es los requerimientos funcionales, que son los que definen funcionalidades con las que va a constar el sistema. Los requerimientos funcionales de un sistema, “Los requerimientos funcionales de un sistema, son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.

Por lo general, estos deben incluir funciones desempeñadas por pantallas específicas, descripciones de los flujos de trabajo a ser desempeñados por el sistema y otros requerimientos de negocio, cumplimiento, seguridad u otra índole” (PMOinformatica.com, 2017). En sus características principales están:

Son requerimientos que muestran funcionalidades claras que requiere el usuario (Ejemplo: un formulario).

Afectan de manera directa al alcance de un proyecto y deben ser tomados con una planificación específica.

Y por el otro lado existen los requerimientos no funcionales, que están orientados en su gran mayoría a la parte estética y no comprometen de gran manera al

funcionamiento del sistema, “Requisitos No Funcionales, son requisitos que imponen restricciones en el diseño o la implementación como restricciones en el diseño o Estándares de Calidad. Son propiedades o cualidades que el producto debe tener” (EcuRed, s.f.), entre sus principales atributos se pueden recalcar los siguientes:

Son características que hacen al producto atractivo, usable, rápido y eficiente.

No modifican las funcionalidades que el sistema tiene, pero si pueden otorgar más funcionalidad.

Son necesarios para que el sistema funcione de manera óptima.

1.4.4 Casos de uso.

Es una herramienta de modelado para clarificar los procesos que son llevados a cabo por los distintos tipos de usuarios de los que dispone el sistema, el utilizar UML en el caso específico de diagramas de casos de uso, brinda la apertura para que cualquier persona ajena al equipo de desarrollo del proyecto, y que disponga de conocimientos técnicos, pueda entender el funcionamiento desde el lado del usuario final del sistema. “Los casos de uso se crean para refinar un conjunto de requisitos de acuerdo con una función o tarea. En lugar de la tradicional lista de requisitos que quizá no trate de forma directa el uso de la solución, los casos de uso reúnen requisitos comunes basados en el tipo de función u objetivo. Los casos de uso definen qué harán los usuarios o funciones en la solución y un proceso empresarial define cómo realizarán esas funciones” (IBM, 2011).

Sin embargo, cabe denotar que los casos de uso no muestran una completa visión de todas las funcionalidades, puesto que se los utiliza para tener una visión general del funcionamiento sin entrar en detalle, y con ello provocar confusión o incluso temor en los encargados de traducir estos diagramas a los clientes finales. “En un diagrama de

casos de uso no se muestran los casos de uso en detalle; solamente se resumen algunas de las relaciones entre los casos de uso, los actores y los sistemas. En concreto, en el diagrama no se muestra el orden en que se llevan a cabo los pasos para lograr los objetivos de cada caso de uso. Esos detalles pueden describirse en otros diagramas y documentos, que pueden vincularse a cada caso de uso” (Microsoft, 2015).

1.4.4.1 Include (Casos de Uso UML).

“En términos muy simples, cuando relacionamos dos casos de uso con un “include”, estamos diciendo que el primero (el caso de uso base) incluye al segundo (el caso de uso incluido). Es decir, el segundo es parte esencial del primero. Sin el segundo, el primero no podría funcionar bien; pues no podría cumplir su objetivo” (Abiztar, s.f.). Se utiliza esta función para indicar la utilización de un caso de uso que no tiene contacto directo con el usuario, pero que en el funcionamiento está inmerso su utilidad puesto que es parte vital del caso de uso principal que tiene el contacto directo con el usuario.

1.4.5 Diagrama de proceso AS IS.

Es la forma en la cual se están llevando a cabo los procesos actualmente, teniendo en cuenta toda la posible interacción que estos llevan a cabo. “Los procesos Tal como Está (As is) constituye la primera fase después de haber identificado el alcance del trabajo. Las medidas para racionalizar, armonizar, simplificar y automatizar los procedimientos y los documentos de comercio dentro de una región o un país solo pueden identificarse y llevarse a cabo una vez que se cuenta con una imagen clara de los procesos y procedimientos actuales” (CEPE/ONU, 2012). Para llevar a cabo este proceso es necesario realizar estos cuestionamientos.

¿Cómo se está trabajando actualmente?

¿Quiénes participan en los procesos?

¿Qué tiempo se requiere para cada una de las tareas?

¿Qué tipo de sistema de información se está utilizando?

1.4.3 Diagrama de proceso TO BE.

El proceso To Be por otro lado brinda un enfoque del proceso ya operativo una vez que este fue sometido a la optimización que se plantea como requerimiento. En síntesis, la etapa de To Be es donde se quiere llegar una vez aplicado todo el proceso de refinamiento del proceso. “El desarrollo del modelo To-Be permite establecer Indicadores de Performance –KPI que apoyaran el mejoramiento del negocio y el accountability” (Saffirio, 2009).

1.5 Arquitectura

1.5.1 Arquitectura cliente.

Se denomina cliente al equipo informático que utilizan los usuarios para gestionar la información que brinda el sistema. “Es el que inicia un requerimiento de servicio. El requerimiento inicial puede convertirse en múltiples requerimientos de trabajo a través de redes LAN o WAN. La ubicación de los datos o de las aplicaciones es totalmente transparente para el cliente” (VALLE, 2005). El cliente de este proyecto de titulación realiza la petición de requerimientos tanto de páginas web como de una aplicación móvil, puesto que el sistema debe interactuar, cuenta con muchos factores comunes entre ellas, a continuación, se citara otra definición de cliente, “rol que desempeña un equipo demandando servicios de los servidores, pero también puede realizar procesamiento local, tales como desplegar páginas web, mostrar ventanas y generar correo electrónico.” (Cubillos).

1.5.2 Arquitectura servidor.

Se denomina servidor al equipo informático encargado de atender los requerimientos que envían los clientes, en el caso de este proyecto de titulación el servidor es único e independiente tanto para el sistema web como para la aplicación móvil, se puede citar un concepto de servidor, “rol que desempeña un equipo ofreciendo un conjunto de servicios a los clientes, tales como manejo de archivos, impresión, páginas web, direccionamiento de correo electrónico, actualización de BD y control de acceso” (Cubillos).

1.5.3 MVC.

Es un patrón arquitectónico para desarrollo de aplicaciones, muy popular para el desarrollo de sistemas en entornos web por su versatilidad y la principal cualidad de separar la lógica de negocio de las interfaces de usuario. “El patrón MVC es un patrón de arquitectura de software encargado de separar la lógica de negocio de la interfaz del usuario y es el más utilizado en aplicaciones Web, ya que facilita la funcionalidad, mantenibilidad y escalabilidad del sistema, de forma simple y sencilla, a la vez que permite “no mezclar lenguajes de programación en el mismo código”” (Bahit, 2014).

Su fundamento es la separación del código en tres capas diferentes, acotadas por su funcionalidad, las capas son: modelo, vista y controlador, cada una de ellas cumple un rol vital en el desarrollo de aplicaciones, a continuación, se muestra un breve detalle de su utilidad.

Modelos. Los objetos de modelo son las partes de la aplicación que implementan la lógica del dominio de datos. A menudo, los objetos de modelo recuperan y almacenan la información en una base de datos.

Vistas. Son los componentes que muestra la interfaz de usuario de la aplicación. Normalmente, esta interfaz de usuario se crea a partir de los datos de modelo o la información a ser mostrada.

Controladores. Son los componentes que controlan la interacción del usuario, trabajan con el modelo y por último seleccionan una vista para representar la interfaz de usuario.

1.5.4 MVP.

(Modelo Vista Presentador), el cual permite el desarrollo de aplicaciones robustas y la utilización de servicios web de una manera óptima. “El SDK de Android no nos ofrece de forma nativa un patrón MVP. Aun así, podemos llevar a cabo alguna aproximación que nos permita desarrollar aplicaciones robustas, que sean más fáciles de mantener, escalar y testear” (develapps, s.f.). El patrón MVP permite separar la capa de presentación de la lógica de la misma, de tal forma que todo lo relacionado con cómo funciona la interfaz queda separado del cómo representarlo en pantalla. Idealmente el patrón MVP permitiría conseguir que una misma lógica pudiera tener vistas totalmente diferentes e intercambiables, en cualquier caso, siempre es mejor usarlo para la arquitectura que no usarlo en absoluto, aplicando este patrón en una aplicación se obtiene al menos tres capas:

Modelo: Esta capa gestiona los datos. Son las clases que se denominan de lógica de negocio.

Vista: Se encarga de mostrar los datos. Aquí se encontrarían los fragmentos y vistas.

Presentador: Se sitúa entre el modelo y la vista, permitiendo conectar la interfaz gráfica con los datos.

1.5.5 Api rest.

Tecnología aplicada a la mayoría de las aplicaciones desarrolladas en los últimos años, rest permite la creación de servicios y aplicaciones que pueden ser usadas por cualquier cliente o dispositivo que entienda el protocolo http. “REST es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON. Es una alternativa en auge a otros protocolos estándar de intercambio de datos como SOAP (Simple Object Access Protocol), que disponen de una gran capacidad, pero también mucha complejidad” (BBVAOPEN4U, 2016). Cuenta con el siguiente detalle de métodos.

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.

1.6 Lenguajes de programación

“Los lenguajes de programación facilitan la tarea de programación, ya que disponen de formas adecuadas que permiten ser leídas y escritas por personas, a su vez resultan independientes del modelo de computador a utilizar” (Lenguajes de Programación, 2016). En el desarrollo de este proyecto de titulación se utilizaron algunos de los lenguajes de programación más empleados a nivel mundial, un lenguaje de programación no es ni bueno ni malo cada uno tiene su utilidad y posee ventajas y desventajas al momento de desarrollar una aplicación, es cualidad de los desarrolladores escoger la mejor tecnología aplicable a su proyecto previo a un estudio que se debe realizar para verificar su utilidad,

“Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Por lo tanto, un lenguaje de programación es un modo práctico para que los seres humanos puedan dar instrucciones a un equipo” (CCM, s.f.). Para el proyecto en cuestión se establecen los siguientes lenguajes de programación: Java, JavaScript (JS) y PHP, a continuación, se da un breve resumen de cada uno de ellos.

1.6.1 Java.

Java es uno de los principales lenguajes de programación a nivel mundial, una de sus principales ventajas al momento de escogerlo, es su cualidad de ser multiplataforma, lo que facilita la utilización de aplicaciones desarrolladas con este lenguaje, “Java es un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por Sun Microsystems. Hay muchas aplicaciones y sitios web que no funcionarán a menos que tenga Java instalado y cada día se crean más. Java es rápido, seguro y fiable. Desde portátiles hasta centros de datos, desde consolas para juegos hasta súper computadoras, desde teléfonos móviles hasta Internet, Java está en todas partes” (Oracle, s.f.), una de sus debilidades cabe en la necesidad de instalar la máquina virtual de Java que sirve para poder compilar las aplicaciones en cada uno de los ordenadores, y también mencionar que las aplicaciones deben ser desarrolladas en la versión actual más estable para mitigar futuros errores.

1.6.2 JavaScript (JS).

JS es un lenguaje de programación que se lo utiliza principalmente para el desarrollo de páginas web, aunque está presente en la actualidad en la construcción de aplicaciones móviles híbridas, “Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar

directamente en cualquier navegador sin necesidad de procesos intermedios” (LIBROSWEB, s.f.), este lenguaje de programación es particularmente útil en el proyecto en cuestión puesto que permite obtener datos de forma dinámica para que los usuarios dispongan de información relevante y oportuna.

1.6.3 PHP.

Es uno de los lenguajes más populares de la actualidad, su versatilidad y facilidad para compilar es uno de los motivos por los cuales la mayoría de los desarrolladores alguna vez han optado por aprenderlo y o utilizarlo en un entorno real de trabajo, “Lo mejor de utilizar PHP es su extrema simplicidad para el principiante, pero a su vez ofrece muchas características avanzadas para los programadores profesionales” (PHP, s.f.). La facultad de ser incrustado directamente en códigos HTML es una de las principales ventajas de utilizarlo, a pesar de que las buenas prácticas de la programación moderna es estructurar el código en capas lo que hace de esta cualidad obsoleta, incluso en ambientes reales de producción puede ser motivo de solución parcial, temporal o definitiva al momento de ocurrir un error en el sistema, otras de sus ventajas es que soporta POO (Programación Orientada a Objetos).

1.6.4 JSON.

“JSON (JavaScript Object Notation - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo” (JSON, s.f.). Es un lenguaje de intercambio de datos legible y entendible para los seres humanos, está comprendido por la colección de pares de nombre y valor. “JSON nació como una alternativa a XML, el fácil uso en JavaScript ha generado un gran número de seguidores de esta alternativa. Una de las mayores ventajas que tiene el uso de JSON es que puede ser leído por cualquier lenguaje de programación” (RODRÍGUEZ, s.f.).

1.6.5 XML.

“XML es un meta-lenguaje que nos permite definir lenguajes de marcado adecuados a usos determinados” (Albasanz, s.f.). XML es un estándar reconocido internacionalmente, puesto que no pertenece a ninguna empresa gracias a su poca dificultad es un lenguaje que es utilizado en algunos sistemas. Su principal funcionalidad es representar información estructurada en la web. “Extensible Markup Language (XML) es un formato de texto simple, muy flexible derivado de SGML (ISO 8879). Originalmente diseñado para satisfacer los desafíos de la publicación electrónica a gran escala, XML también está desempeñando un papel cada vez más importante en el intercambio de una amplia variedad de datos en la Web y en otros lugares” (Quin, 2016).

1.7 Herramientas

El proyecto de titulación consta de 2 componentes, por un lado, se tiene el sistema web y por el otro lado la aplicación móvil, por lo tanto, se citan las herramientas utilizadas para cada una de ellas dentro del desarrollo del proyecto.

Para el desarrollo del sistema web se utiliza Sublime Text 3, Apache 2 y PhpMyAdmin, y para el desarrollo de la aplicación móvil se utiliza Android Studio conjuntamente con herramientas ya citadas. A continuación, se muestra un breve resumen con información relevante de las herramientas utilizadas.

1.7.1 Sublime Text 3.

Es un editor de texto muy útil y con una variedad de herramientas que facilitan el trabajo de los desarrolladores, su portabilidad de la al usuario una de las principales funcionalidades al momento de realizar proyectos desde zonas geográficamente distantes, o en lugares donde no se pueda acceder a una maquina con mucho recurso o

editores más robustos. “Sublime Text es un editor de código multiplataforma, ligero y con pocas concesiones a las florituras. Es una herramienta concebida para programar sin distracciones. Su interfaz de color oscuro y la riqueza de coloreado de la sintaxis, centra nuestra atención completamente” (F.Manuel, 2012).

1.7.2 Apache 2.

“Apache es el servidor web utilizado con más frecuencia en los sistemas Linux. Los servidores web se emplean para entregar las páginas que los equipos cliente solicitan. Los clientes normalmente solicitan y visualizan las páginas web mediante aplicaciones de navegación como Firefox, Opera, Chromium e Internet Explorer” (ubuntu Documentation., s.f.), en el caso puntual de este proyecto se utiliza el servidor de aplicaciones Apache para montar el sistema web y también los servicios web con los que cuenta la aplicación móvil.

1.7.3 PhpMyAdmin.

Es el gestor de base de datos utilizado para el desarrollo de los módulos antes citados, el sistema consta con una sola base de datos única que está alojada en el mismo servidor que se encuentra la configuración del servidor de aplicaciones. Es una de los gestores de base de datos más utilizados en el mundo para el desarrollo de páginas web, esto se debe a su compatibilidad con sistemas operativos, navegadores, dispositivos finales, “PhpMyAdmin es una herramienta de software libre escrita en PHP, destinada a manejar la administración de MySQL a través de la Web. PhpMyAdmin soporta una amplia gama de operaciones en MySQL y MariaDB. Las operaciones de uso frecuente (gestión de bases de datos, tablas, columnas, relaciones, índices, usuarios, permisos, etc.) se pueden realizar a través de la interfaz de usuario, mientras que todavía tiene la capacidad de ejecutar directamente cualquier sentencia SQL” (phpMyAdmin, s.f.).

1.7.4 Android Studio.

Un IDE de desarrollo diseñado específicamente para la construcción de aplicaciones Android, por su gran cantidad de herramientas útiles para el desarrollo de aplicación entre las que se pueden destacar, que posee su propia máquina virtual configurable para poder probar las aplicaciones desarrolladas, otra de las grandes ayudas que brinda a los desarrolladores de aplicaciones móviles es una estadística en tiempo real reflejada con información desde los servidores de Google Play con el porcentaje de dispositivos registrados con sus sistema operativo actual, punto muy importante a tomar en cuentas por la velocidad con la que evoluciona las versiones de sistemas operativo Android y evitar que las aplicaciones tengan problemas de compatibilidad, “Android Studio proporciona las herramientas más rápidas para crear apps en todas las clases de dispositivos Android. La edición de códigos de primer nivel, la depuración, las herramientas de rendimiento, un sistema de compilación flexible y un sistema instantáneo de compilación e implementación te permiten concentrarte en la creación de aplicaciones únicas y de alta calidad” (Android Studio, s.f.).

1.7.5 Firebase.

Es la plataforma de desarrollo en la nube de Google la cual está disponible para varias plataformas (Android, IOS, y plataformas Web). “Los productos de Firebase funcionan bien por sí solos, comparten datos y estadísticas, y se puede acceder a ellos desde un panel unificado” (Firebase), es una herramienta poderosa para el desarrollo de aplicaciones por su gran variedad de servicios ofertados de forma gratuita. “Firebase se trata de una plataforma móvil creada por Google, cuya principal función es desarrollar y facilitar la creación de apps de elevada calidad de una forma rápida, con el fin de que se pueda aumentar la base de usuarios y ganar más dinero. La plataforma está subida en la nube y está disponible para diferentes plataformas como

iOS, Android y web. Contiene diversas funciones para que cualquier desarrollador pueda combinar y adaptar la plataforma a medida de sus necesidades” (Cardona, 2016).

1.7.6 JMeter.

“La aplicación JMeter de Apache es un software de código abierto, una aplicación Java 100% pura diseñada para cargar el comportamiento funcional y medir el rendimiento. Originalmente fue diseñado para probar aplicaciones Web, pero desde entonces se ha expandido a otras funciones de prueba” (Apache, s.f.). Herramienta utilizada para ejecutar pruebas de rendimiento en el sistema web, se puede utilizar esta herramienta para simular cargas pesadas para ser procesadas en un servidor o grupo de servidores.

1.8 Prototipo

Un prototipo es un aproximado a un producto real, en software se generaliza que el prototipo debe tener un cierto nivel de funcionalidad para que los clientes puedan observar y delimitar como va a quedar su producto final. “Un prototipo de un modelo de negocio es más bien una herramienta de pensamiento que nos ayuda a explorar las diferentes direcciones que podría tomar nuestro modelo de negocio. ¿Cómo cambiaría nuestro modelo si añadimos otro segmento de clientes? ¿Qué consecuencias tendría desmontamos una estructura de costes?” (EMPRE-RED, s.f.). Manipular un prototipo del sistema a elaborar permite abordar cuestiones de estructura, arquitectura, tecnología, diseño y cuestiones más minuciosas que le dan al desarrollador un mejor entendimiento de los requerimientos brindados por los clientes.

“Esta técnica puede ser utilizada en cualquier etapa del desarrollo. A medida que el proceso progresa y el producto se completa, el prototipo ha de abarcar, cada vez más las características del producto final. Llegados a un punto, la construcción de

prototipos adicionales resultará menos eficiente que usar las construcciones iniciales para el producto” (Hom, 2000).

1.9 Pruebas de caja negra

Las pruebas de caja negra son aquellas que se aplican para verificar la funcionalidad del sistema, sin tomar en cuenta la arquitectura aplicada para la consecución del objetivo planteado, “Las pruebas de caja negra se centran principalmente en lo que “se quiere” de un módulo, charter o sección específica de un software, es decir, es una manera de encontrar casos específicos en ese modulo que atiendan a su especificación” (Globe, 2017). En las pruebas de caja negra únicamente se toma en cuenta las entradas y salidas que se tiene al ejecutarse el sistema, “En las pruebas de caja negra, nos enfocamos solamente en las entradas y salidas del sistema, sin preocuparnos en tener conocimiento de la estructura interna del programa de software” (Terrera, 2016) .

Capítulo 2

2.1 Análisis del proceso

A través de las reuniones conjuntas con los colaboradores de la Dirección Técnica de Pastoral, se establece el flujo del proceso normal que los colaboradores y estudiantes seguían para cumplir a cabalidad con el VAS. Se aborda el proyecto contando como el primer paso el levantamiento del proceso manual que llevan a cabo los colaboradores de la Dirección Técnica de Pastoral, debido a que esto ayuda a identificar a los participantes que están involucrados en el intercambio de información, y toda la información que se necesita para que el proceso se lleve a cabo, y de esta forma determinar los resultados que se generan de la realización del proceso.

2.1.1 Proceso manual (AS IS).

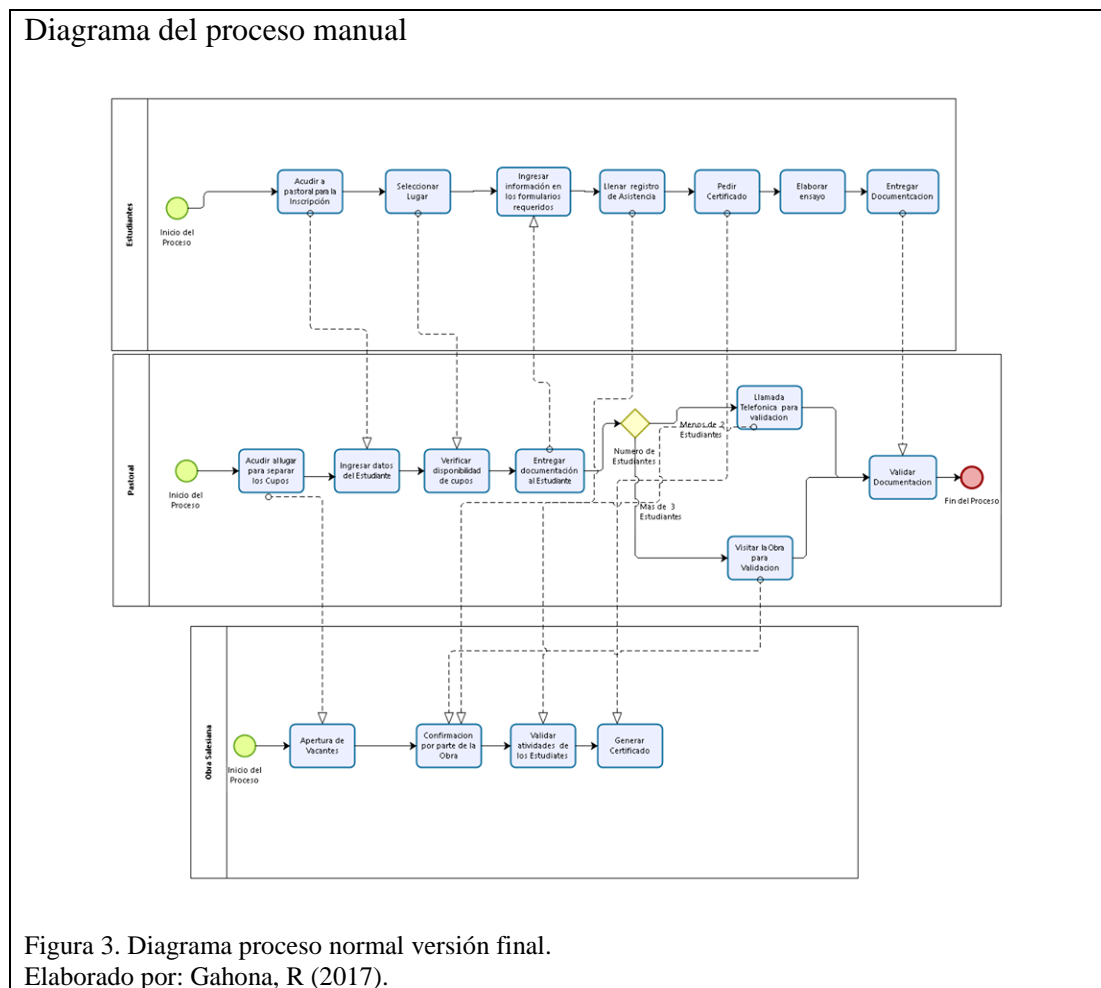


Figura 3. Diagrama proceso normal versión final.
Elaborado por: Gahona, R (2017).

Los estudiantes acuden a la Dirección Técnica de Pastoral para consultar la información de las distintas instituciones UPS, una vez los estudiantes seleccionen una de las instituciones solicitan el ingreso al proceso VAS para ello el colaborador de la Dirección Técnica de Pastoral ingresa la información de manera manual lo que genera cuellos de botella al acumularse los estudiantes para que ingresen la información de en las fichas. Los estudiantes llenan los formatos que son entregados por la Dirección Técnica de Pastoral y no cuentan con ningún tipo de respaldo en caso de pérdida o desgaste de los documentos, los colaboradores de la Dirección Técnica de Pastoral validan el proceso de los estudiantes mediante una llamada o a su vez sí el número de estudiantes es mayor a 3 agendan una visita a la institución, una vez finalice el proceso se vuelve a entregar la documentación conjuntamente con un ensayo y todos los documentos físicos más un CD de la documentación digitalizada.

La figura 3 muestra el diagrama del proceso normal en su versión final luego del análisis de todo el flujo que seguían los colaboradores y estudiantes en el VAS. Con el fin de mejorar y agilizar el proceso normal, se establecieron algunos cambios porque que al ser un proceso el cual no contaba con un seguimiento correcto de la información, al hacer todo manual y sin ningún respaldo, no se podía garantizar una acción de contingencia que les ayude a solventar cualquier problema, teniendo así como resultado:

2.1.2 Proceso optimizado con software (TO BE).

Diagrama del proceso optimizado con software

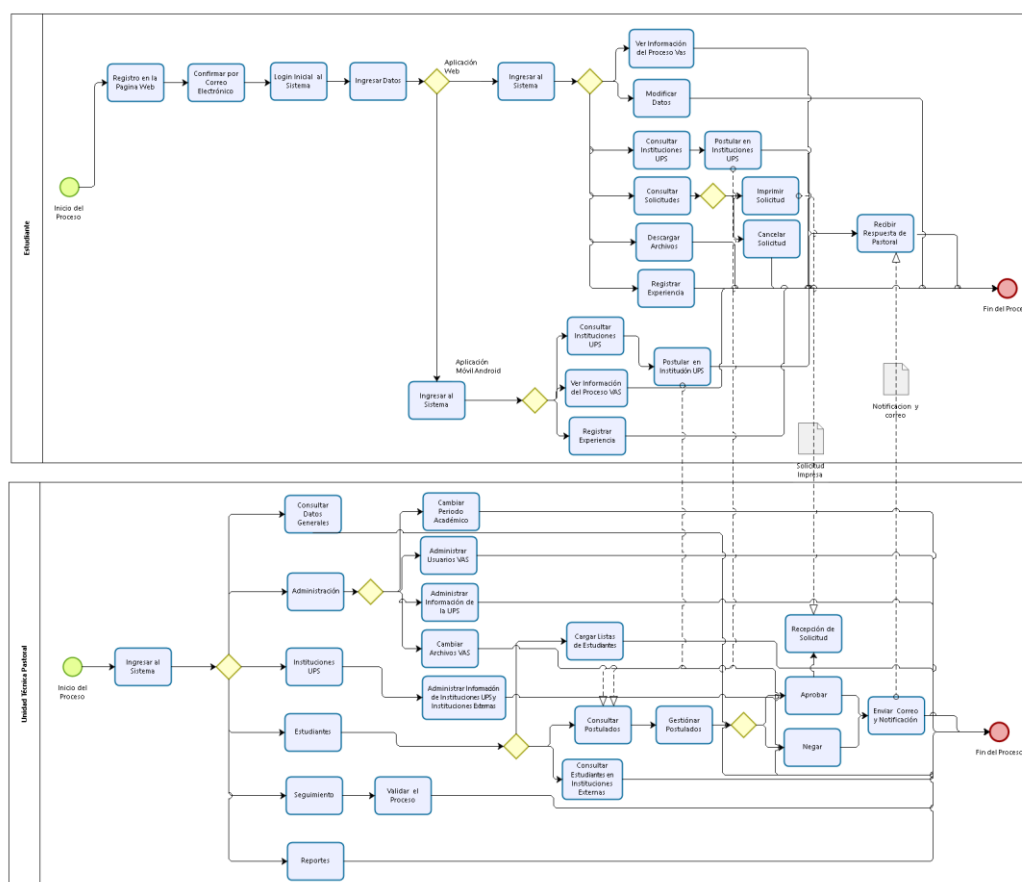


Figura 4. Diagrama proceso mejorado versión final.
Elaborado por: Gahona, R (2017).

El proceso inicia cuando el estudiante accede al portal web y registra sus datos para obtener una cuenta en el proceso VAS, una vez que el estudiante cree su cuenta ingresa al sistema, se completa los datos del estudiante para poder continuar con las tareas que se realizan dentro del proceso VAS, cuando el estudiante haya completado su información, escoge entre realizar la postulación a la institución UPS en el portal web o a su vez realizarlo desde la aplicación móvil Android donde cuentan con la información replicada de las instituciones.

Posteriormente cuando el estudiante se postule a una de las instituciones UPS espera a que el colaborador de la Dirección Técnica de Pastoral valide su información y decida aprobar o negar su solicitud, una vez la postulación sea aprobada el estudiante acude a las oficinas de la Dirección Técnica de Pastoral para hacer sellar sus formularios que se generan de manera automática, el estudiante finaliza el proceso cuando completa el ingreso de la experiencia salesiana, que es el registro de asistencia con el que se finaliza el proceso del estudiante, este ingreso se valida por el colaborador de la Dirección Técnica de Pastoral para evitar que los estudiantes completen este campo sin acudir a la institución.

La figura 4 muestra el diagrama del proceso optimizado con software en su versión final, el mismo que se obtuvo como resultado de cambios en el proceso manual, mejorando el flujo del proceso y eliminando parte de la carga que era realizada por los colaboradores de la Dirección Técnica de Pastoral, cambios tales como, registro de información, postulación y finalización del proceso fueron asignados para que sean realizados por cada estudiante, y que la gestión, seguimiento y administración sean realizados por parte de los colaboradores.

2.2 Especificación de requerimientos

La especificación de requerimientos ayudan al cliente a plasmar en un documento, todas las necesidades que el sistema web y la aplicación móvil debían cumplir.

En el anexo 2 se encuentran todos los requerimientos solicitados por la Dirección Técnica de Pastoral. A continuación, se muestra un resumen con los requerimientos más relevantes que se utilizaron para el desarrollo del sistema:

Inscripción y postulación de estudiantes: Permite a los estudiantes ingresar sus datos personales y seleccionar una institución disponible para que realice su voluntariado.

Registro de instituciones UPS y de instituciones externas: Permite ingresar nuevas instituciones en cualquier momento y así mismo gestionar los cupos para que los estudiantes puedan asistir.

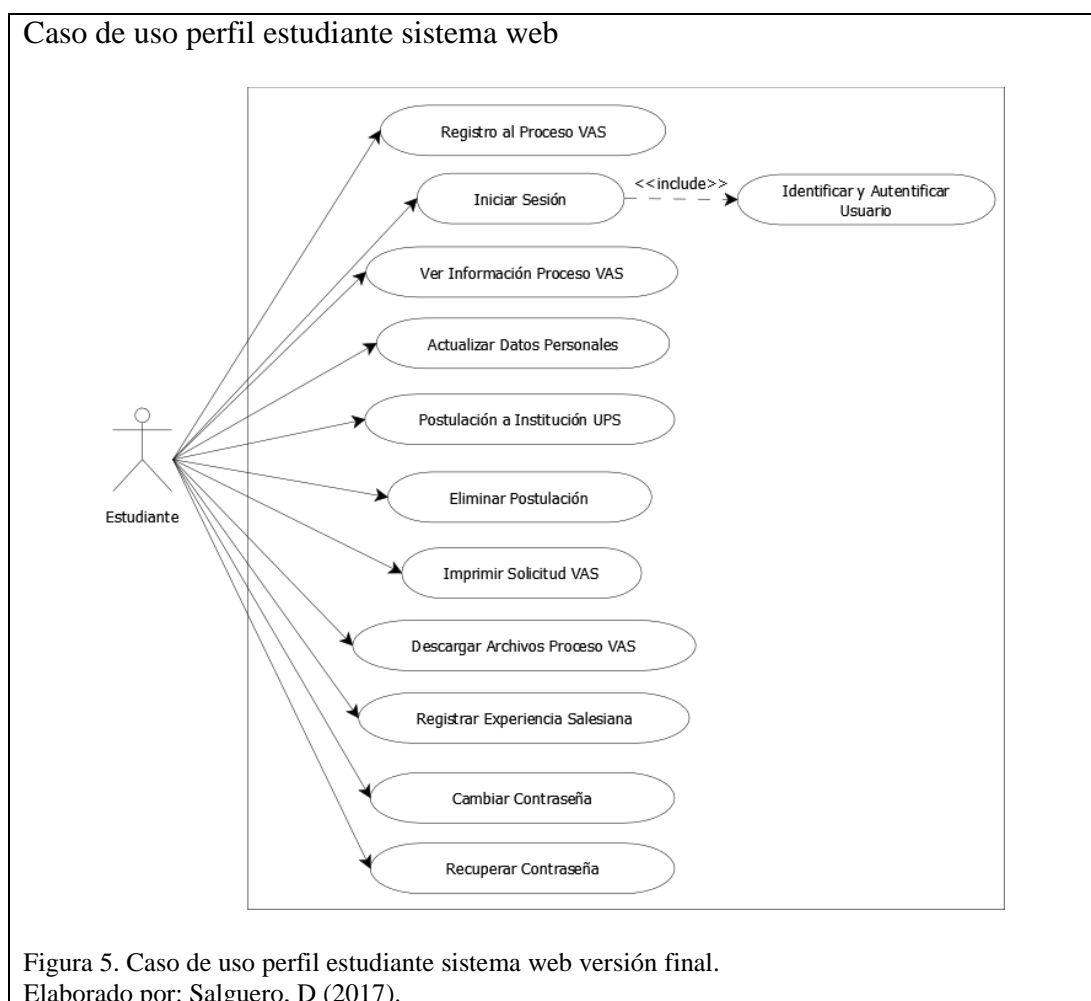
Seguimiento de estudiantes: Permite llevar un control de cada estudiante para verificar si se cumple con el proceso.

Reportes: Permiten obtener información con gráficas del proceso VAS.

2.3 Casos de uso

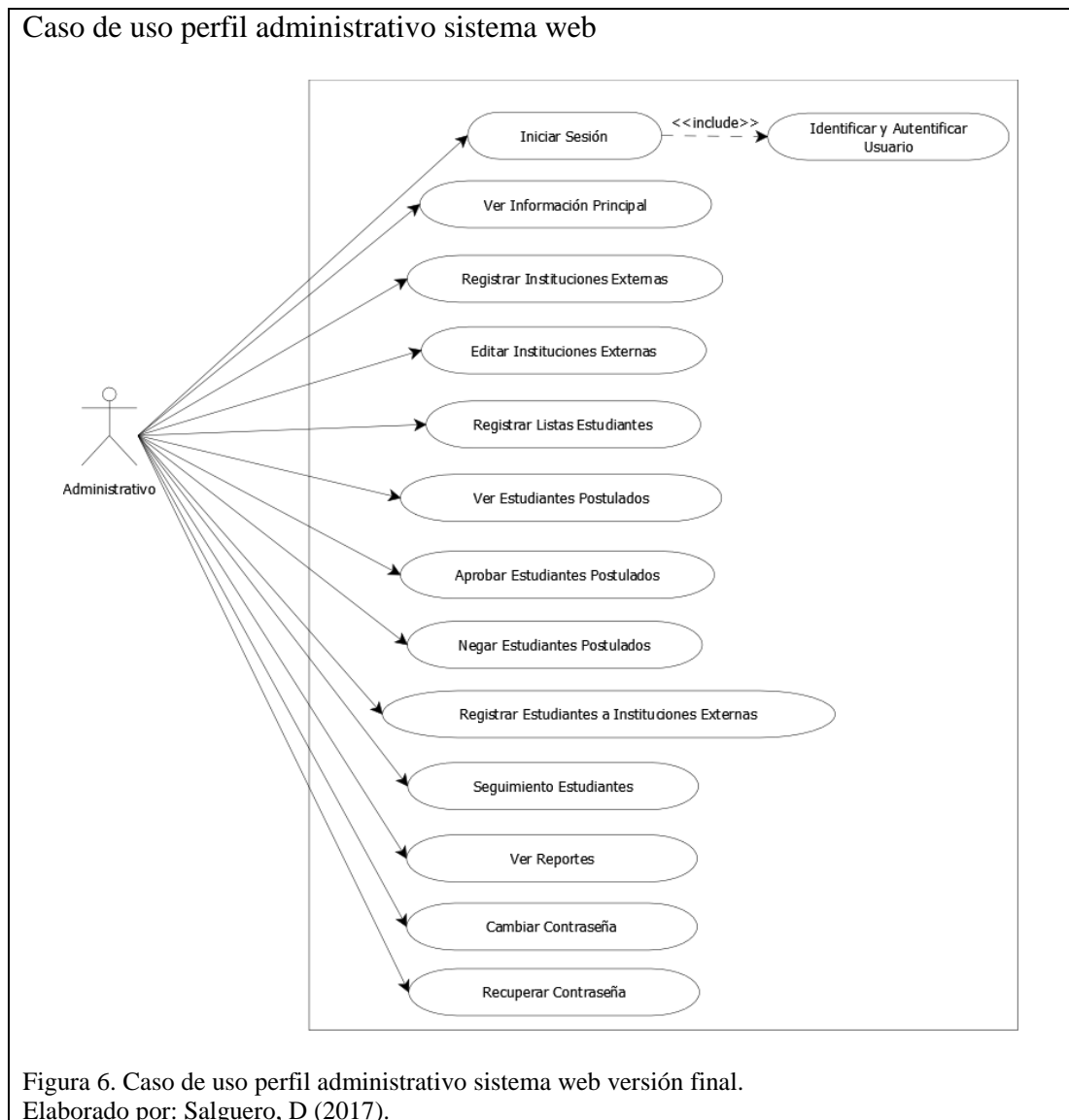
A medida del avance del proyecto y en transcurso de los sprints, los casos de uso de análisis y diseño que permitieron obtener una visión más concreta de las características con las que cuenta cada usuario con su respectivo perfil, tanto en el sistema web como en la aplicación móvil, una vez definidos todos los requerimientos, se obtuvieron las últimas versiones de cada uno de los casos de uso que se muestran a continuación:

2.3.1 Caso de uso perfil estudiante sistema web.



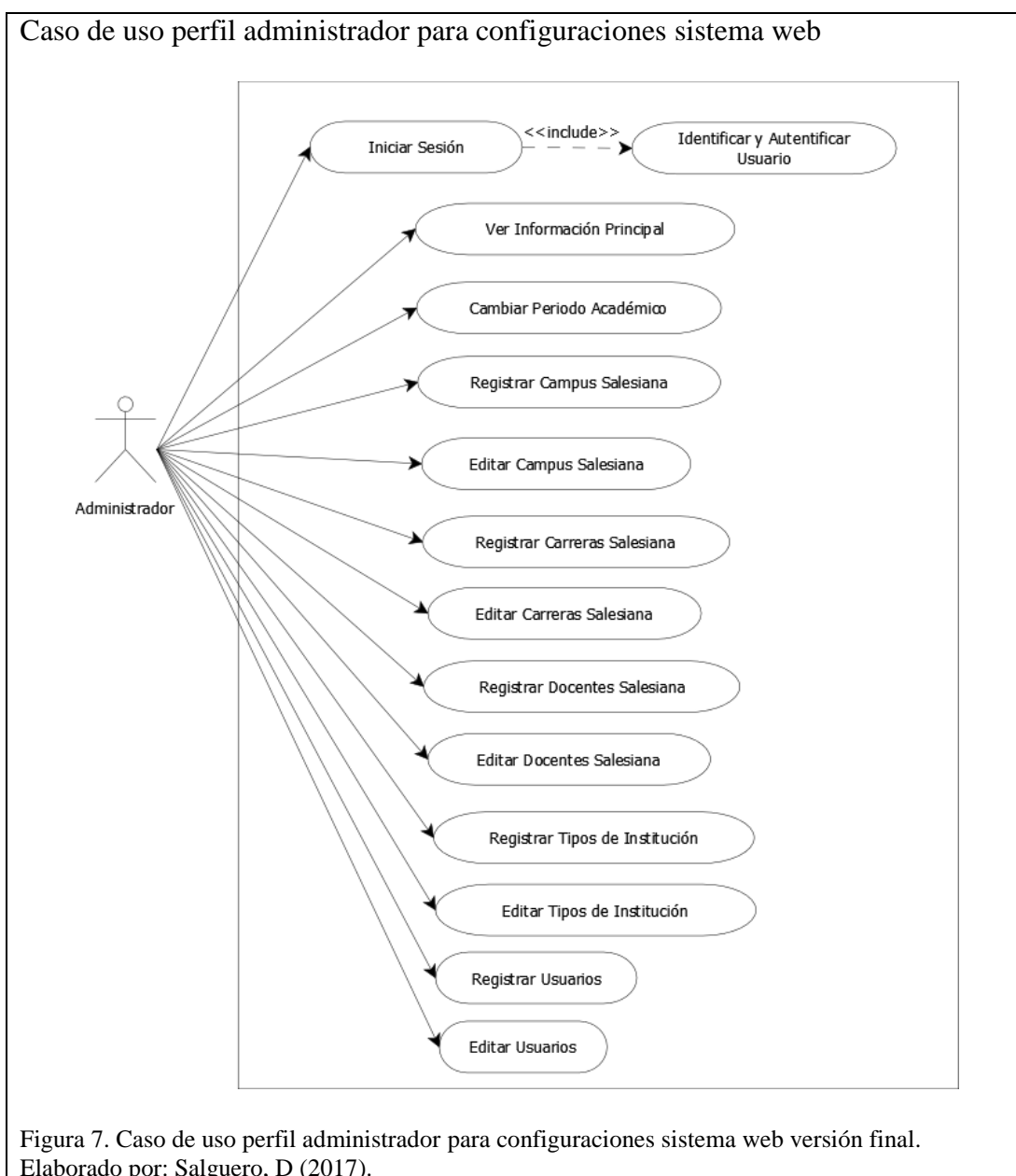
La figura 5 muestra que el perfil de usuario estudiante, cuenta con el registro de la cuenta para el acceso al sistema, de tal forma que una vez creada su cuenta el usuario pueda ingresar a las características específicas tales como: información del proceso, actualizar datos personales, postularse a una institución UPS, eliminar su postulación en cualquier momento, descargar los archivos proporcionados por el usuario administrativo, registrar la su experiencia salesiana y recuperar o cambiar la contraseña de su cuenta.

2.3.2 Caso de uso perfil administrativo sistema web.



La figura 6 muestra que el perfil de usuario administrativo, cuenta con un inicio de sesión para acceder al sistema, adicionalmente cuenta con: información principal del VAS, registro y edición de instituciones externas, registro de las listas de estudiantes, verificar los estudiantes postulados al VAS, aprobación y negación de estudiantes, registrar estudiantes a instituciones externas, dar seguimiento a los estudiantes, ver reportes generales del sistema, cambiar y recuperar la contraseña de la cuenta.

2.3.3 Caso de uso perfil administrador sistema web.



La figura 7 muestra que el perfil de usuario administrativo para las configuraciones, cuenta con un inicio de sesión para acceder al sistema, adicionalmente cuenta con: información principal, cambio de periodo académico, registro y edición de campus, registro y edición de carreras, registro y edición de docentes, registro y edición de tipos de institución, registro y edición de usuarios del sistema.

Caso de uso perfil administrador para registro y seguimiento sistema web

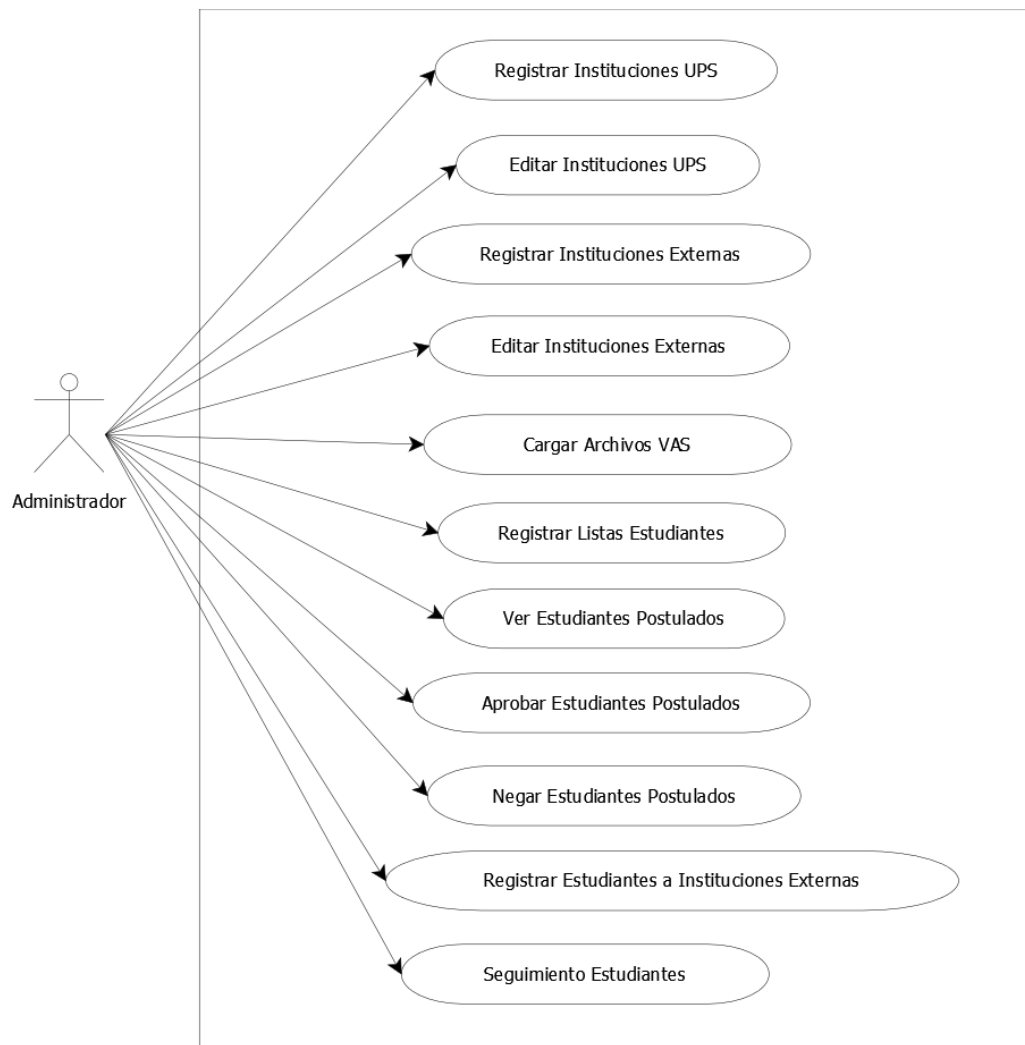


Figura 8. Caso de uso perfil administrador para registro y seguimiento sistema web versión final. Elaborado por: Salguero, D (2017).

La figura 8 muestra que el perfil de usuario administrativo para el registro y seguimiento, cuenta con el registro y edición de instituciones UPS, registro y edición de instituciones externas, carga de archivos de formatos para los estudiantes, registro de listas de estudiantes, ver estudiantes postulados, aprobación y negación de estudiantes postulados, registro de estudiantes a instituciones externas y dar seguimiento a los estudiantes.

Caso de uso perfil administrador para reportes y perfil sistema web

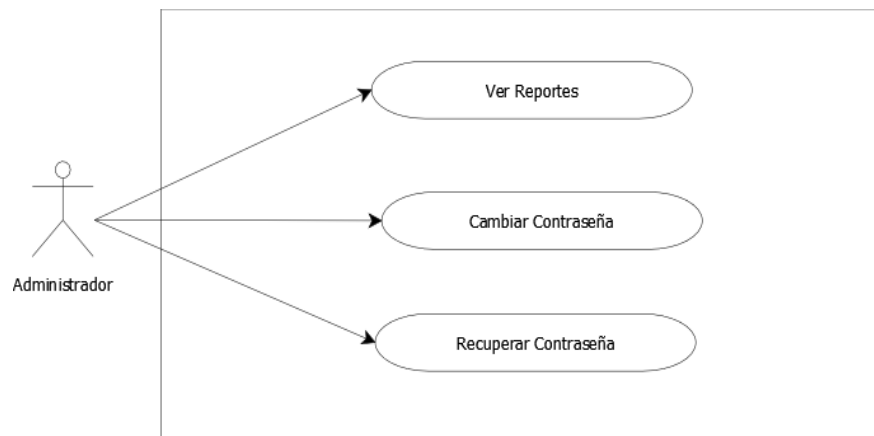


Figura 9. Caso de uso perfil administrador para reportes y perfil sistema web versión final.
Elaborado por: Salguero, D (2017).

La figura 9 muestra que el perfil de usuario administrador para reportes y perfil, cuenta con todos los reportes los sistemas tanto específicos como generales y un cambio y recuperación de la contraseña.

2.3.4 Caso de uso perfil estudiante aplicación móvil.

Caso de uso perfil estudiante aplicación móvil

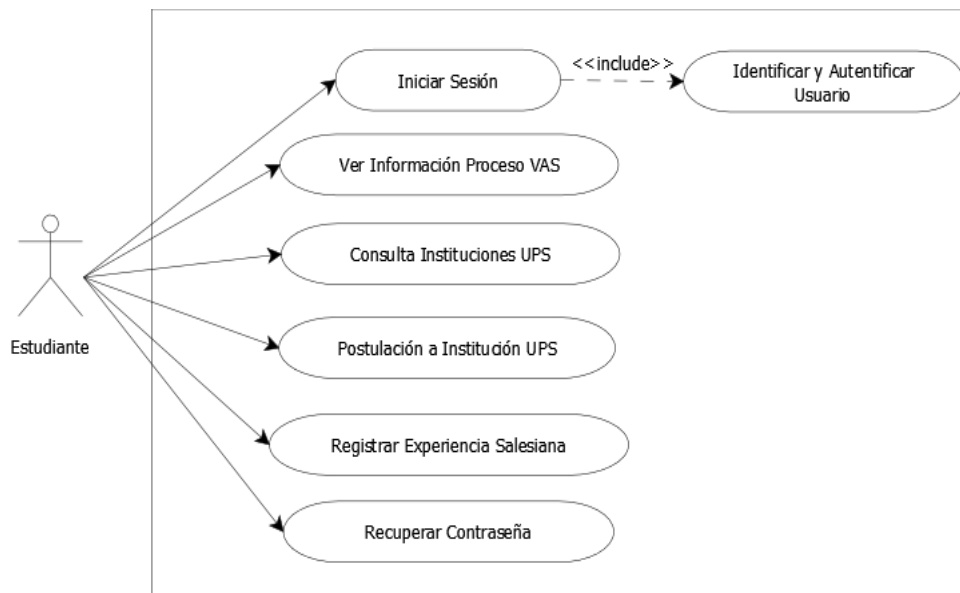


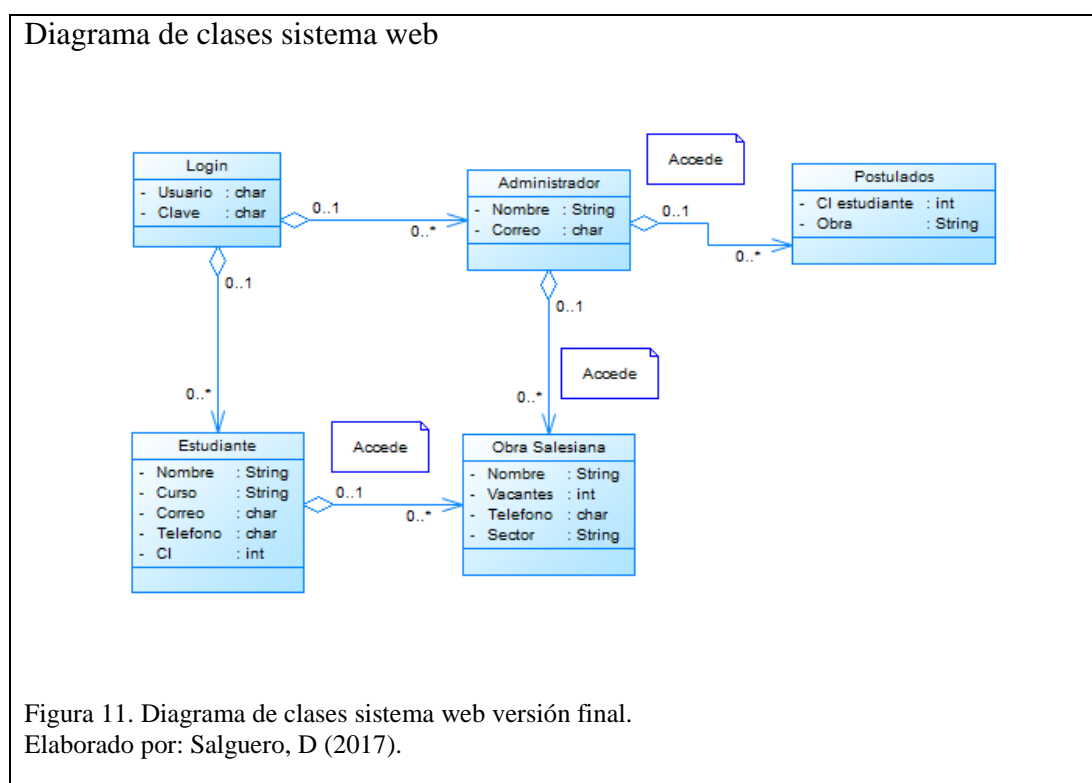
Figura 10. Caso de uso perfil estudiante aplicación móvil versión final.
Elaborado por: Gahona, R (2017).

La figura 10 muestra que el perfil de usuario estudiante, cuenta con un inicio de sesión, puede ver la información del proceso VAS, consultar todas las instituciones UPS disponibles, postulación a una institución, registro de la experiencia salesiana y la recuperación de la contraseña.

2.4 Diagrama de clases

2.4.1 Diagrama de clases sistema web.

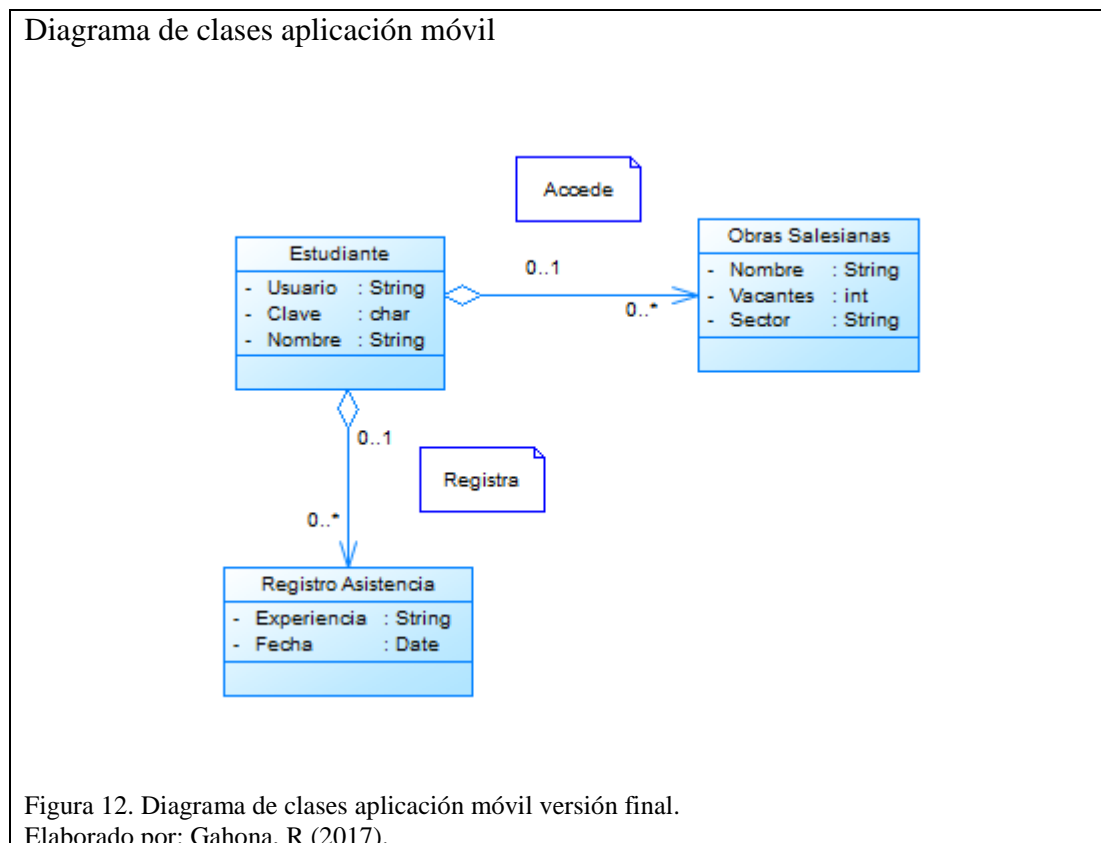
El principal desafío fue el manejar la comunicación entre el cliente y el servidor, una solución para que se comuniquen entre ellos fue la de utilizar una clase genérica que gestione todas las peticiones al servidor y otras clases adicionales generales que faciliten la reutilización de código al momento de la programación.



La figura 11 muestra el diagrama de clases del sistema web el cual se encuentra en su versión final.

2.4.2 Diagrama de clases aplicación móvil.

El principal desafío es la utilización de una sola base de datos para 2 aplicaciones en diferentes lenguajes y tecnologías, que deben funcionar de forma transparente, es por este motivo que no se dispone de almacenamiento interno por parte de la aplicación móvil en los dispositivos y todo se realiza con peticiones directas al servidor.



La figura 12 muestra el diagrama de clases de la aplicación móvil el cual se encuentra en su versión final.

2.5 Diagrama de base de datos

Una vez identificados los casos de uso y de las funcionalidades que los usuarios poseerían, se define el siguiente diagrama de la base de datos con el fin de obtener un esquema que sea flexible para trabajar con el sistema web y la aplicación móvil en conjunto, obteniendo así el siguiente modelo que se encuentra en su versión final:

Tabla 1. Tablas de la base de datos versión final

Tabla	Descripción
configuraciones	Almacena los datos de configuraciones necesarias para el sistema.
docente	Almacena los datos docentes que son ingresados a través el sistema.
estudiante	Almacena los datos de los estudiantes que son ingresados a través el sistema.
experiencia	Almacena los datos de la experiencia salesiana que es ingresada a través el sistema.
listas	Almacena los datos de las listas de los estudiantes que son ingresadas a través el sistema.
listaEstudiantes	Almacena los datos de los estudiantes en cada lista que es ingresada a través el sistema.
obras	Almacena los datos de las Instituciones UPS que son ingresadas a través el sistema.
postulados	Almacena los datos de los estudiantes postulados que son ingresados a través el sistema.
tipo	Almacena los datos de los tipos de instituciones que son ingresados a través el sistema.
tokenUsuarios	Almacena los datos de los tokens de los dispositivos móviles cada vez que un usuario ingresa a través de la aplicación móvil.
usuario	Almacena los datos de los usuarios que son ingresados a través el sistema.

usuario campus	Almacena los datos de la relación entre usuarios y los campus UPS.
----------------	--

Nota: Tabla de tablas de la base de datos en su versión final.

Diagrama entidad relación base de datos

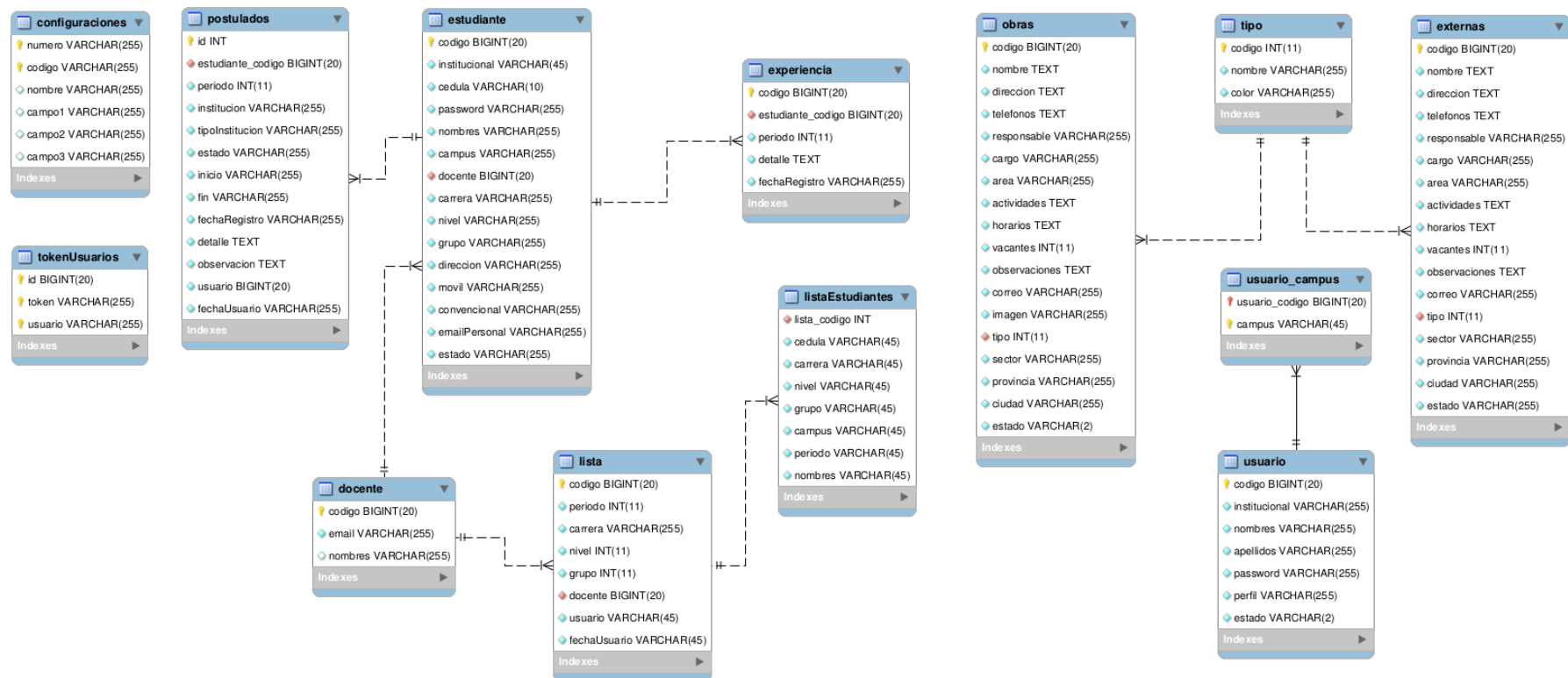
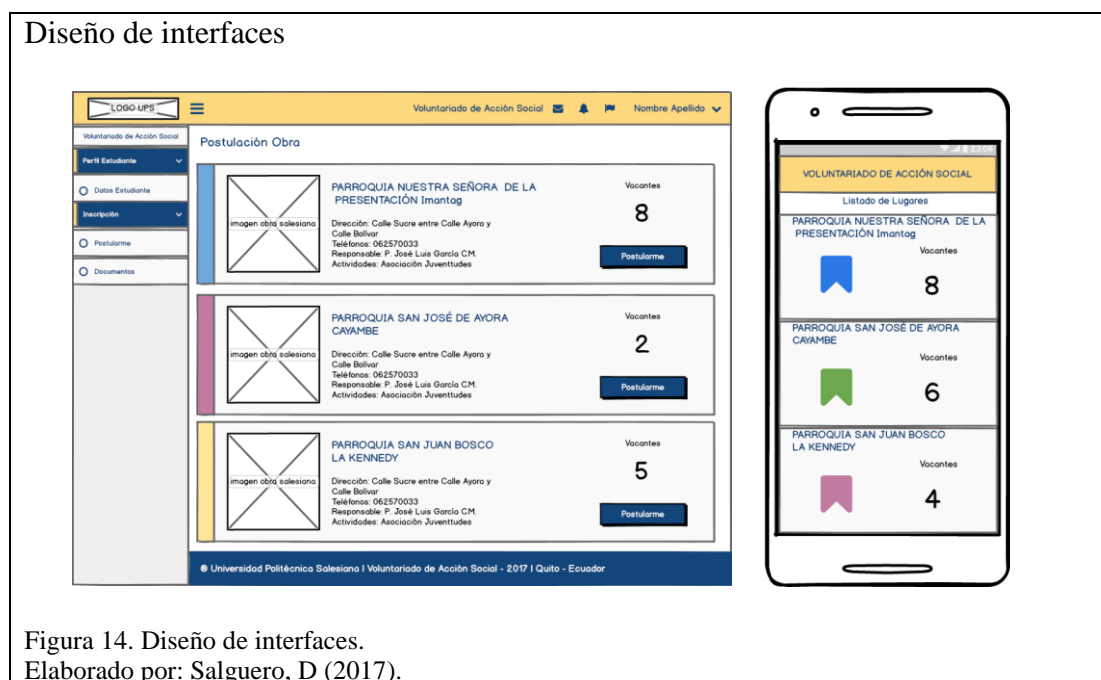


Figura 13. Diagrama de la base de datos versión final.
Elaborado por: Salguero, D (2017).

La figura 13 muestra el diagrama de la base de datos en su versión final, luego de establecer y cumplir con todos los requerimientos solicitados.

2.6 Diseño de interfaces

La estrategia que se emplea para el diseño de las interfaces tanto del sistema web como de la aplicación móvil, fue la de utilizar los colores de la universidad dentro del sistema y utilizar los mismos formatos que la Dirección Técnica de Pastoral utiliza para el manejo de las listas de estudiantes, formatos de inscripción, etc. y de esta manera ayudar a mitigar el rechazo de los usuarios y para que no exista mucho impacto al momento de la utilización del software. Interfaces con pocos pasos ayudaría a que los usuarios se adapten mucho mejor al sistema web y a la aplicación móvil permitiendo así que la interacción con los usuarios sea lo más intuitiva posible y fácil de aprender. El desarrollo basado en componentes permitiría reutilizar piezas de código previamente elaborado y permitiría obtener diversos beneficios como mejora de la calidad y la reducción del ciclo de desarrollo.



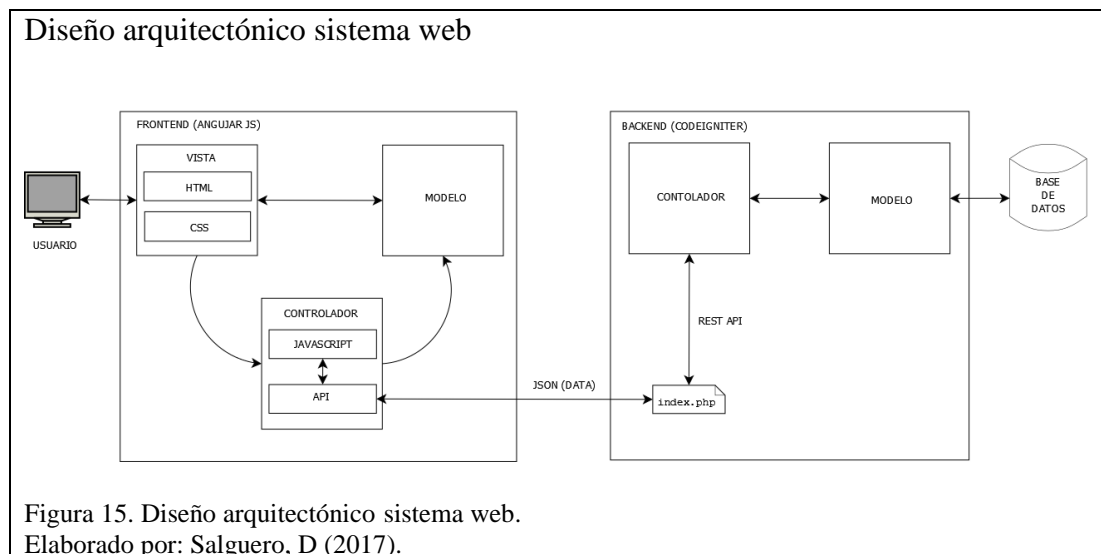
La figura 14 muestra el diseño de interfaces una vez definidos posiciones de componentes, tamaños y colores.

Capítulo 3

3.1 Arquitectura sistema web

Se aplica el patrón arquitectónico de software MVC (Modelo, Vista, Controlador) tanto para la arquitectura de cliente como para la arquitectura del servidor del sistema web, debido a que la aplicación se puede desarrollar rápidamente, de forma modular y mantenible. Al separar las funciones en modelos, vistas y controladores hace que la aplicación sea muy ligera, adicionalmente el diseño modular permite realizar rápidamente el prototipado, y también permite hacer cambios en una parte de la aplicación sin que las demás se vean afectadas.

Se utiliza en el Front-End el framework web AngularJS y en el Back-End el framework web CodeIgniter, teniendo como resultado el siguiente diseño arquitectónico:



La figura 15 muestra la comunicación entre el Front-End y el Back-End en el sistema web, y trabajan de la siguiente forma:

El usuario realiza una petición (get, post, put, delete) la cual es recibida por el controlador y este a su vez se comunica con el modelo genérico para identificar que el tipo de la petición y la dirección del directorio en el Back-End donde se encuentra dicha función.

La petición viaja a través del api de AngularJS en un JSON mediante una llamada al index.php de CodeIgniter.

CodeIgniter a través su api rest hace la llamada respectiva a su controlador el cual llama al modelo que gestiona los datos.

El modelo de CodeIgniter mediante su modelo, gestiona todos los datos de la base de MySQL y responde un estado 200 que indica que la página ha cargado correctamente, y un estado 500 que indica un error interno, lo que se puede traducir que la página web tiene algún error en el código.

La respuesta retorna a través del controlador del Back-End y llega al controlador de AngularJS por medio del index.php para así gestionarlo en el Front-End.

3.2 Estándares de programación sistema web

Las variables y nombres de funciones tanto en el Front-End como en el Back-End se encuentran es español latino a excepción del modelo genérico y de las palabras reservadas de cada lenguaje de programación.

3.2.1 Estándares Front-End.

Los nombres de los controladores cuentan con el nombre seguido del prefijo “Controller” por ejemplo: UsuariosController.

Los nombres de los archivos del controlador inician con el nombre seguido del prefijo “controller.js” por ejemplo: usuarios.controller.js.

Los nombres de los archivos de los servicios inician con el nombre seguido del prefijo “service.js” por ejemplo: storage.controller.js.

Los nombres de los archivos de los servicios inician con el nombre seguido del prefijo “service.js” por ejemplo: storage.controller.js.

Las vistas se encuentran dentro de la carpeta views dentro de su respectiva carpeta en cada módulo.

Los controladores se encuentran dentro de la carpeta controllers dentro de su respectiva carpeta en cada módulo.

3.2.2 Estándares Back-End

Todas las clases se extienden del CI_Model.

Todos los controladores se encuentran en la carpeta controllers.

Todos los modelos se encuentran en la carpeta models.

Los nombres de las funciones en los controladores.

Los nombres de los archivos de los controladores inician con la primera letra en mayúsculas, por ejemplo: Administracion.php.

Los nombres de los archivos de los controladores inician con el nombre seguido del prefijo “_model.php” por ejemplo: Administracion_model.php

Las clases tanto en los controladores como en los modelos Inician con la primera letra en mayúscula.

3.3 Listado de componentes sistema web

El sistema web se encuentra desarrollado en 2 lenguajes de programación, el Front-End se encuentra desarrollado en JavaScript y el Back-End se encuentra desarrollado en PHP, adicionalmente se utilizaron los siguientes frameworks y formatos para el desarrollo del sistema:

Bootstrap v4.0.0 Alpha

AngularJS v1.5

CodeIgniter v3.5

JSON

jquery v3.2.1

3.4 Código relevante sistema web

De todo el código generado en el sistema web se destacan partes importantes y fundamentales que son la base del código y que permitieron agilizar el proceso de desarrollo a lo largo del proyecto.

3.4.1 Modelo genérico Front-End.

Modelo genérico parte 1

```
(function() {  
  'use strict';  
  
  angular.module('Model')  
    .factory('model', function($http, $q, _, $rootScope, server) {  
  
    var method_map = {  
      get: 'get',  
      post: 'create',  
      put: 'update',  
      delete: 'remove'  
    };  
  
    var handle_errors = function(method, element) {  
      return function(response) {  
        if (response.status === 200) {  
          return response.data;  
        }  
        $rootScope.$broadcast('error', {  
          action: method_map[method],  
          element: element,  
          problems: response.data.message,  
          custom_message: response.data.message  
        });  
      };  
    };  
  
    function request_load_data(method, resource, action, params) {  
      var path = full_path(resource, action, params);  
      var options = {  
        transformRequest: angular.identity,  
        headers: {  
          'Content-Type': undefined  
        }  
      };  
      var request_params = new FormData();  
      request_params.append('file', params.data.file);  
      return $http[method](path, request_params, options)  
        .then(handle_errors(method, resource));  
    }  
  }  
})
```

Figura 16. Código modelo genérico parte 1 sistema web.

Elaborado por: Salguero, D (2017).

La figura 16 muestra cómo se encuentra estructurada la primera parte del modelo genérico del Front-End, cuenta con la función `handle_errors` que permite mostrar errores de las peticiones y la función `request_load_data` la misma contiene el servicio `$http` básico de AngularJS que facilita la comunicación con los servidores http remotos del Back-End mediante JSONP, mediante estas funciones se logra la comunicación

con el Back-End para el envío de datos ya que no se contaba con funciones generales que permitan la comunicación y gestión de errores.

Modelo genérico parte 2

```
function request(method, resource, action, params) {
  var compare_undefined = undefined;
  var path = full_path(resource, action, params);
  var request_params = {};
  if (params !== compare_undefined && params.data) {
    request_params = {
      data: params.data
    };
  }
  return $http[method](path, request_params).then(handle_errors(method, resource));
}

function full_path(resource, action, params) {
  var compare_undefined = undefined;
  var path = server.url + resource + '/' + action;
  if (params !== compare_undefined && params.url_params) {
    _(params.url_params).each(function(param, key) {
      path += '/' + key + '/' + param;
    })
  }
  return path;
}

return function(resource) {
  return {
    get: function(action, params) {
      return request('get', resource, action, params);
    },
    create: function(action, params) {
      return request('post', resource, action, params);
    },
    update: function(action, params) {
      return request('put', resource, action, params);
    },
    remove: function(action, params) {
      return request('delete', resource, action, params);
    },
    load_data: function(action, params) {
      return request_load_data('post', resource, action, params);
    }
  };
};

})();
```

Figura 17. Código modelo genérico parte 2 sistema web.
Elaborado por: Salguero, D (2017).

La figura 17 muestra cómo se encuentra estructurada la segunda parte del modelo genérico del Front-End, cuenta con la función request la misma que es llamada a través de los controladores para hacer las peticiones al servidor y la función full_path la cual facilita el manejo de rutas en el cliente, mediante estas funciones se logra obtener configuraciones generales para la reutilización de código.

3.4.2 Módulo general Font-End.

Módulo general

```
(function() {  
  'use strict';  
  
  angular.module("vas", [  
    'ngAnimate',  
    'ngCookies',  
    'ngTouch',  
    'ngSanitize',  
    'ngMessages',  
    'ngAria',  
    'ngResource',  
    'ui.router',  
    'ui.bootstrap',  
    'toastr',  
    'ui-notification',  
    'ngFileUpload',  
    'ngFileSaver',  
    '$q-spread',  
    'Core',  
    'Modulos',  
    'Directivas',  
    'Servicios'  
  ])  
  .config(function(NotificationProvider) {  
    NotificationProvider.setOptions({  
      delay: 3000,  
      startTop: 47,  
      startRight: 10,  
      verticalSpacing: 20,  
      horizontalSpacing: 20,  
      positionX: 'right',  
      positionY: 'top'  
    });  
  })  
  .run(run);  
  
  run.$inject = ['$rootScope', '$location', '$cookies', '$http', '$anchorScroll'];  
  function run($rootScope, $location, $cookies, $http, $anchorScroll) {  
    var anchorScroll = $rootScope.$on("$locationChangeSuccess", function(){  
      $anchorScroll();  
    });  
    $rootScope.$on('$destroy', anchorScroll);  
  }  
  
})();
```

Figura 18. Código módulo general sistema web.

Elaborado por: Salguero, D (2017).

La figura 18 muestra cómo se encuentra estructurado el módulo general del Front-End, el mismo que cuenta con todas las inyecciones de dependencias y de librerías (importaciones de archivos) necesarias para que la aplicación funcione correctamente con AngularJS, adicionalmente cuenta con NotificationProvider el cual permite las notificaciones en cliente y el \$locationChangeSuccess que verifica el cambio de rutas, mediante este módulo general se obtuvo la inyección de una manera general y sencilla para no hacerlo en cada módulo del sistema.

3.4.3 Notificaciones dispositivo móvil Back-End.

Notificaciones dispositivo móvil

```
<?php
class Individual_model extends CI_Model {

    function __construct() {
        parent::__construct();
    }

    function send_notification($tokens, $message) {
        $url = 'https://fcm.googleapis.com/fcm/send';
        $fields = array(
            'to' => $tokens,
            'data' => $message,
        );
        $headers = array(
            'Authorization: key = AAAAC8GdwKU:APA91bF273TxyiF3TpafCuurOGILU9swn0PxLNdlnrO2pVYgqGmftKcpFZu',
            'Content-Type: application/json',
        );
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $url);
        curl_setopt($ch, CURLOPT_POST, true);
        curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 0);
        curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
        curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($fields));
        $result = curl_exec($ch);
        if ($result === FALSE) {
            die('Curl failed: ' . curl_error($ch));
        }
        curl_close($ch);
        return $result;
    }
}
```

Figura 19. Código individual_model sistema web.
Elaborado por: Salguero, D (2017).

La figura 19 muestra cómo se encuentra estructurado la clase Individual_model con el api de firebase para el envío de las notificaciones al momento de realizar la acción respectiva, el mismo que cuenta con el Auhorization key que permite la comunicación con firebase, mediante esta clase se logra contar con una clase general de las notificaciones, y así obtener un servicio que facilite esta comunicación.

3.5 Arquitectura aplicación móvil

Para el desarrollo de la aplicación móvil se utiliza el siguiente modelo arquitectónico de desarrollo mvp, programación nativa de Android, por lo tanto, se estructura las vistas de la aplicación con código XML.

Un ejemplo de implementación en el que el usuario dispone de un formulario donde puede introducir contactos.

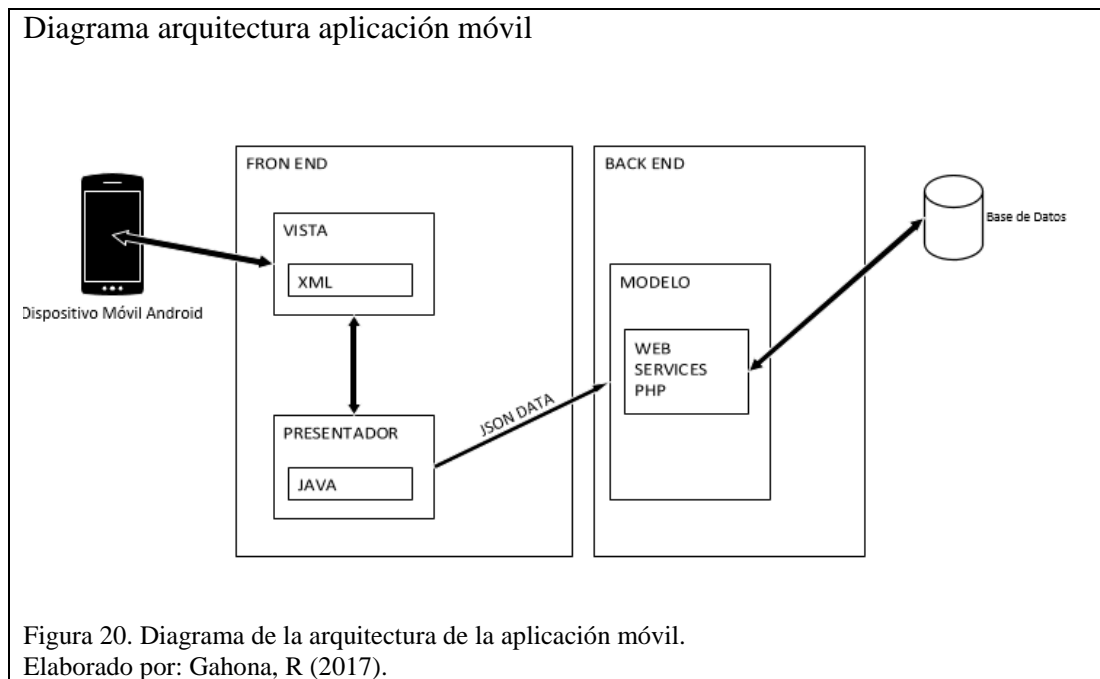
El usuario introduce un contacto y pulsa el botón “añadir contacto”.

El Presentador crea el objeto Contacto con los datos introducidos por el usuario y se lo pasa al Modelo para que lo introduzca en la base de datos.

El Modelo inserta el contacto en la base de datos.

Si todo ha ido bien, el presentador limpia el formulario y refresca la lista de contactos para que aparezca el que acaba de añadir. En caso de que se haya producido algún error, muestra una alerta con un mensaje de error.

La capa de negocio o en este caso el presentador se lo diseña con el lenguaje de programación Java y se utiliza servicios web desarrollados en PHP para la conexión con la capa de datos o modelo.



La figura 20 muestra el diagrama de la arquitectura de la aplicación móvil la cual funciona de la siguiente manera:

El Front-End está constituido por las 2 grandes capas de código, la vista está formada por código XML que estructura cada uno de los elementos que son presentados para la interacción con el usuario y por otro lado existe la capa de conexión con los datos llamada presentador que consta de código Java estructurado en función a los elementos que se presenta en la vista.

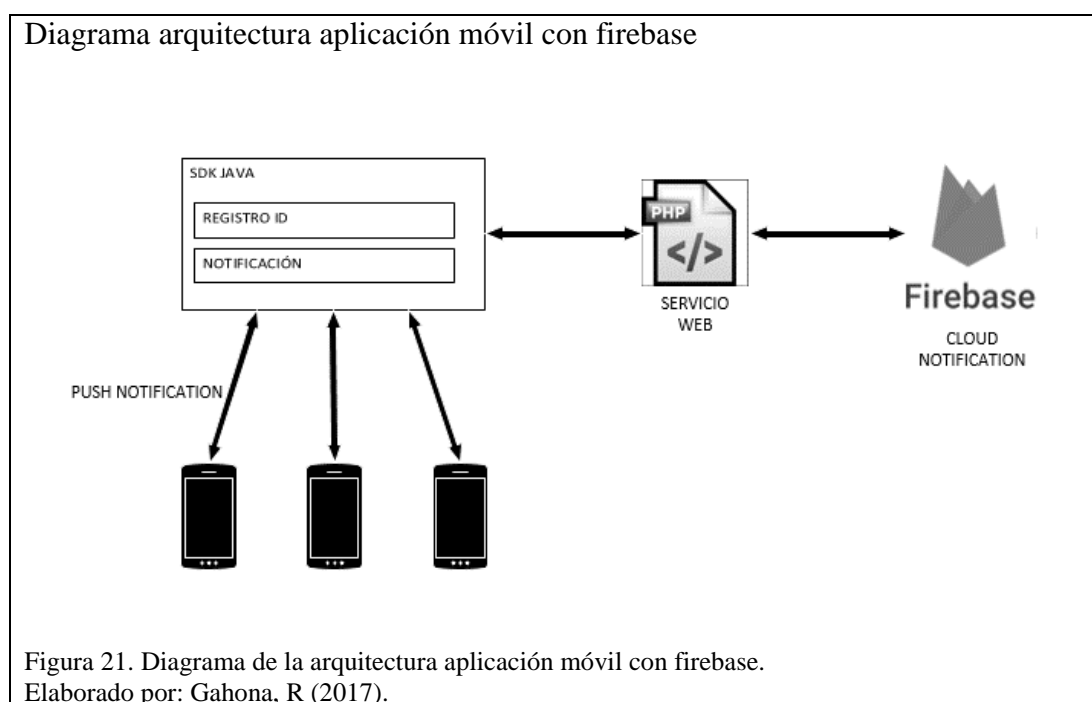
La conexión entre los elementos de la vista y los datos es gestionada por programación en Java que relaciona elementos identificados por un (id) y de esta manera al ser enlazados se los puede gestionar para realizar los ingresos y consultas de información.

Desde la aplicación se gestionan los datos con llamadas asincrónicas a servicios web elaborados en PHP (llamadas puntuales al servidor que no consumen conexiones en segundo plano). Las llamadas asincrónicas reciben los datos enviados de los servicios web con conexión directa a la base de datos.

Los servicios web envían y reciben datos en formato JSON, estos datos son decodificados en el presentador y posterior administrados para mostrarlo en una forma visualmente atractiva para los usuarios.

Los llamados a los servicios web se los parametrizo en una clase Java donde se puede gestionar su direccionamiento individualmente e independiente de la vista mostrada al usuario.

Para la presentación en la vista se utiliza adaptadores de objetos que permiten gestionar los elementos de la vista y poder mostrar datos dinámicos.



La figura 21 muestra el diagrama de la aplicación móvil que utiliza el api de Google para el envío de notificaciones push, la misma que funciona de la siguiente manera:

Es necesario el instanciar una cuenta de Google y acceder a los servicios de firebase, una vez que se ha procedido con la configuración de la cuenta a utilizar, dirigirse al panel de configuración de firebase, una vez en se esté en este apartado se debe crear un nuevo proyecto para poderlo asignar al nombre del proyecto en desarrollo.

Una vez se haya configurado el servidor, es necesario estructurar los servicios web que permitan la comunicación directa con la aplicación, esto se lo realizara mediante una base de datos.

En el lado de la aplicación es necesario instanciar 2 servicios con los que debe contar la aplicación conjuntamente con los permisos de red que deben ser otorgados para que la aplicación pueda funcionar sin problemas. El primer servicio es el de asignación de token para ser registrados como dispositivos disponibles para el envío de las notificaciones, el otro servicio es el de notificaciones que permite que la aplicación use los servicios propios del sistema operativo y pueda mostrar información en forma de notificaciones push.

Luego que se registren los 2 servicios anteriores, se debe estructurar el código Java para poder gestionar el ingreso de los token asignados en la base de datos del sistema mediante el uso de llamada asincrónica y a posterior ingresos desde la aplicación móvil.

Ya registrado el dispositivo conjuntamente con un usuario, se utiliza el servicio web para que con el uso de la key (código alfanumérico único que sirve de identificador) entregada por firebase se pueda enviar contenido personalizado a cada uno de los usuarios que registren dispositivos y sesione activas en ellos, las llamadas son gestionadas desde el sistema web del sistema.

Las notificaciones enviadas contienen información sobre las respuestas obtenidas a las solicitudes generadas por los estudiantes, al momento que el estudiante vea la notificación y al pulsar sobre ella lo redirecciona a la aplicación para que pueda revisar y validar la información.

3.6 Estándares de programación aplicación móvil

3.6.1 Programación Android.

Se estructura una sola clase para el llamado a las ip de los servicios web.

Se mantuvo el nombre de las clases Java de forma similar a los layout (archivo con extensión XML que corresponde a los componentes visuales de los que dispone el formulario) de las vistas para poder identificarlos.

Se brinda permiso de Internet, red local y de redes WIFI para el uso de la aplicación.

Los colores de la aplicación se los establece dentro del apartado de values (archivo con extensión XML con el contenido de parámetros de la aplicación) y en el estilo propio de toda la aplicación para la armonía con el aplicativo web.

3.6.2 Programación de servicios web.

Se estructuro los archivos de login en una sola carpeta, al igual que los archivos de la postulación que realizan los estudiantes.

Los archivos de los maneja por separado para evitar que un daño o error en alguno de ellos ocasione daños en toda la aplicación.

Todos los archivos mantienen conexiones MySQLi para evitar incompatibilidad con versiones de servidores superiores a 5.0.

Todos los archivos mantienen configuración de caracteres utf-8 (es un set de caracteres universal, un estándar en el que se definen todos los caracteres necesarios para la escritura de la mayoría de los idiomas hablados en la actualidad).

3.6.3 Base de datos.

La tabla de registro de token dispone de 3 columnas las cuales son únicas para poder realizar reemplazo en el caso de que se encuentre en el mismo teléfono más de un

usuario al momento de ingresar y consultar la información, y el ingreso de nuevos datos en el caso de tratarse de dispositivos diferentes.

3.7 Listado de componentes aplicación móvil

La aplicación Android para dispositivos móviles se desarrolla en función a dos elementos puntuales, la presentación de la aplicación construida con componentes nativos de Android hechos en lenguaje XML y enlazados con Java para la manipulación de los datos de forma dinámica.

Y por otro lado se encuentra el motor de gestión de datos que está constituido por servicios web desarrollados en PHP, a continuación, se listara las tecnologías utilizadas para su construcción.

XML

Java

JSON

PHP

3.8 Código relevante aplicación móvil

A pesar de que todo el código de la aplicación es importante destacan ciertos fragmentos de código que son el motor principal del funcionamiento, y que dan una apertura al conocimiento de la estructuración total de la codificación.

3.8.1 AndroidManifests.xml.

“Todas las aplicaciones deben tener un archivo AndroidManifest.xml (con ese nombre exacto) en el directorio raíz. El archivo de manifiesto proporciona información esencial sobre tu aplicación al sistema Android, información que el sistema debe tener para poder ejecutar el código de la app.” (Developers, s.f.)

Este archivo es vital para el funcionamiento de la aplicación, es donde se denota las funciones y el curso de ejecución de cada una de ellas para el buen funcionamiento del sistema. También es el apartado donde se cargan los permisos que la aplicación dispone para el uso correcto en los dispositivos de los estudiantes.

1. Es importante recalcar que también dispone del llamado a los 2 servicios necesarios para la integración con firebase.

Manifiesto de la aplicación AndroidManifests.xml parte 1

```
<service
  android:name=".FirebaseMessagingService">
  <intent-filter>
    <action android:name="com.google.firebase.MESSAGING_EVENT"/>
  </intent-filter>
</service>
<service
  android:name=".FirebaseInstanceIdService">
  <intent-filter>
    <action android:name="com.google.firebase.INSTANCE_ID_EVENT"/>
  </intent-filter>
</service>
```

Figura 22. Código AndroidManifest.xml con la configuración de los servicios parte 1.
Elaborado por: Gahona, R (2017).

La figura 22 muestra la primera parte de la estructura necesaria para declarar los servicios utilizados para la integración con firebase y la recepción de notificaciones.

La estructura de los servicios denotados posee estrictamente su nombre y por supuesto la funcionalidad otorgada por el tipo de servicio utilizado, tanto el de instance id que otorga un token al dispositivo como también el de messaging (nombre del servicio utilizado para recibir notificaciones desde firebase) que permite la recepción de notificaciones push (mensajes que se envían de forma directa a los dispositivos de los usuarios).

Manifiesto de la aplicación AndroidManifests.xml parte 2

```
<activity android:name=".Login">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity
    android:name=".Main2Activity"
    android:label="Voluntariado Acción Social"
    android:theme="@style/AppTheme.NoActionBar" />

<activity android:name=".RegistroAcEstudiantes"
    android:parentActivityName=".Main2Activity">
    <meta-data android:name="android.support.PARENT_ACTIVITY"
        android:value=".Main2Activity"/>
</activity>
<activity android:name=".AcercaDe"
    android:parentActivityName=".Main2Activity">
    <meta-data android:name="android.support.PARENT_ACTIVITY"
        android:value=".Main2Activity"/>
</activity>
```

Figura 23. Código AndroidManifest.xml con su configuración parte 2.
Elaborado por: Gahona, R (2017).

La figura 23 muestra la segunda parte de la estructura necesaria para declarar los servicios utilizados para la integración con firebase y la recepción de notificaciones.

La aplicación móvil dispone de 4 interfaces Android llamadas (activity) con una amplia variedad de componentes visuales que ayudan al usuario a el ingreso y consulta de información que debe realizar dentro del proceso VAS. Las activity tienen características puntuales muy marcadas como su nombre, su rol dentro de la aplicación y de ser necesario a que activity corresponde como padre o hijo (una activity puede depender de otra para ser ejecutada, en este caso se denomina que es una activity hija). Solo debe existir un activity como lanzador, puesto que se debe estructurar la aplicación con una secuencia de paso a seguir por el usuario.

3.8.2 Google-services.json.

Este archivo es importante para la integración del servicio de notificaciones push enviadas gracias a firebase, cabe recalcar que el archivo es único para cada proyecto, y una vez que se lo genere, no puede ser utilizado para versionamiento de proyectos que no tengan como base el anterior por completo, incluido el nombre único de proyecto.

Archivo de configuración otorgado por firebase

```
"project_info": {
  "project_number": "497169580197",
  "firebase_url": "https://vas-version-2.firebaseio.com",
  "project_id": "vas-version-2",
  "storage_bucket": "vas-version-2.appspot.com"
},
"client": [
  {
    "client_info": {
      "mobilesdk_app_id": "1:497169580197:android:c2d6edcc67e2c07c",
      "android_client_info": {
        "package_name": "com.example.rodody.respaldo1.respaldo1"
      }
    },
    "oauth_client": [
      {
        "client_id": "497169580197-k6r4sc7fui97rv9b7cbugqotj1ponf.apps.googleusercontent.com",
        "client_type": 3
      }
    ],
    "api_key": [
      {
        "current_key": "AIzaSyDXMG7ut3B2_1lNRTEKqbFMb-ZPdyamlKs"
      }
    ],
    "services": {
      "analytics_service": {
        "status": 1
      },
      "appinvite_service": {
```

Figura 24. Código Google-services.json
Elaborado por: Gahona, R (2017).

Uno de los requerimientos principales es dar a conocer a los estudiantes la información pertinente de forma remota y eficiente, por ello se opta por la utilización de notificaciones push que llegan directamente al dispositivo móvil Android de los estudiantes.

La figura 24 muestra ciertos elementos clave para entender el funcionamiento del servicio de notificaciones push que se implementa en la aplicación móvil.

Como primer punto se tiene la sección de información del proyecto, es aquí donde se puede visualizar la configuración creada para el proyecto ejecutado y se puede evidenciar claramente el código único de proyecto conjuntamente con datos y direcciones del servidor utilizado.

Como siguiente apartado, se tiene la información única del cliente, este nombre es único para las aplicaciones desarrolladas en el gestor utilizado y en general es un estándar colocar un nombre único para poder utilizar el proyecto y colgarlo en la nube en los servidores de Google Play. En este punto se puede entender mejor que los archivos utilizados y el registro de la aplicación debe ser único para que se pueda garantizar que el servicio funcione de manera idónea.

Como otro de los puntos importantes dentro de este archivo existe la presencia de la clave (key) del proyecto, en este caso es una clave única que se asigna para el servidor, puede ser gestionada desde la plataforma de firebase y permite que se acceda a sus servicios.

3.8.3 Estructura de envío de datos con post (asincrónico).

La aplicación móvil dispone de conexiones a la base de datos del sistema mediante servicios web que envían y reciben información de forma independiente y con relación a una supuesta sesión simulada gracias a la ayuda de codificación en Java que permite relacionar la información que el usuario envía o recibe en su dispositivo móvil.

Estructura para el envío de parámetros

```
@Override
protected Map<String, String> getParams() throws AuthFailureError {
    HashMap<String, String> hashMap = new HashMap<String, String>();
    hashMap.put("cod", codigoEst);
    hashMap.put("peri", periodo);
    hashMap.put("codI", idobra.get(position).toString());
    hashMap.put("finicio", textofechainicio.getText().toString());
    hashMap.put("ffin", textofechafin.getText().toString());
    hashMap.put("fecha", fechayhoradeingreso);
    hashMap.put("detalle", detalleHo.getText().toString());
    return hashMap;
}
```

Figura 25. Código con datos enviados desde la aplicación móvil.
Elaborado por: Gahona, R (2017).

El requerimiento expresa que es necesario enviar información desde la aplicación móvil, la cual debe tener un grado de seguridad moderado, debido a que la información siempre es importante, al no contar con una base de datos de forma local en la aplicación, es necesario que el envío de información sea lo más eficiente posible.

La figura 25 muestra la estructura básica para el envío de parámetros hacia el servicio web que ingresa la información de la postulación realizada por el estudiante en cualquiera de las instituciones UPS.

Su funcionamiento está estructurado de la siguiente manera, es necesario definir el tipo de dato que se va a enviar al servicio web para que se proceda con el ingreso de la información suministrada, en este caso se lo estructura con el envío de un par de parámetros de tipo cadena, y su estructura está compuesta por: nombre del parámetro y valor del parámetro. Cabe recalcar que este envío no afecta al tipo de dato que acepte la base de datos, puesto que los campos donde estos datos son insertados son de tipo varchar con una longitud de 255, lo que permite una completa compatibilidad con los tipos de datos enviados.

Envío de información mediante método post

```
request = new StringRequest(Request.Method.POST, URL, new Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        try {
            JSONObject jsonObject = new JSONObject(response);
            if (jsonObject.names().get(0).equals("success")) {
                Toast.makeText(getApplicationContext(), "Error: " + jsonObject.getString("success"), Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(getApplicationContext(), "Grandioso: " + jsonObject.getString("insert"), Toast.LENGTH_SHORT).show();
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
});
```

Figura 26. Código con datos enviados desde la aplicación móvil.
Elaborado por: Gahona, R (2017).

La figura 26 muestra como una vez que el estudiante llene los campos correspondientes y pulse sobre la opción de envío, el servicio web responde con un mensaje en formato JSON que corresponde a la acción que se realiza con la consulta de existencia que se ejecuta cada vez que el estudiante desea postularse en una de las instituciones UPS. Los mensajes significan:

Success: Se encuentran resultados y no puede postularse.

Insert: No se encuentran registros y se procese al ingreso.

3.8.4 Estructura básica para la vista de instituciones UPS.

Se presenta la información de forma visualmente atractiva para que los estudiantes puedan realizar su postulación con la seguridad de saber las actividades, autoridades y además datos relevantes de la institución seleccionada, para ello se opta por utilizar un elemento fácil de manipular y con muchas ventajas visuales para el usuario, este elemento se llama ListView, que permite el uso de adaptadores desarrollados a medida, lo que es una gran ventaja en la carga de grandes cantidades de información con formato específico.

Adaptador de lista de instituciones.

```
public View getView(int position, View convertView, ViewGroup parent) {  
    View itemView = inflater.inflate(R.layout.list_row, parent, false); // conection con el  
    // variables usadas  
    txtTitle = (TextView) itemView.findViewById(R.id.list_row_title);  
    smartImageView = (ImageView) itemView.findViewById(R.id.list_row_image);  
    txtvacantes = (TextView) itemView.findViewById(R.id.numerovacantes);  
    txtsector = (TextView) itemView.findViewById(R.id.textsector);  
    textovacantes = (TextView) itemView.findViewById(R.id.textovacantes);  
    textosector = (TextView) itemView.findViewById(R.id.textosector);  
    // fin variables  
    String urlfinal = variables.IPImagenesObras() + imagen.get(position).toString(); // URL  
    Rect rect = new Rect(smartImageView.getLeft(), smartImageView.getTop(), smartImageView.getR  
    // smartImageView.setImageUrl(urlfinal, rect); // setea las imagenes de la base desde la  
    txtTitle.setText(titulo.get(position).toString()); // setea datos  
    txtvacantes.setText(vacantes.get(position).toString()); // setea datos  
    // txtsector.setText(sector.get(position).toString()); // setea datos
```

Figura 27. Fragmento de código del adaptador.

Elaborado por: Gahona, R (2017).

El requerimiento expresa la necesidad de mostrar la información existente de las instituciones UPS, con las que cuentan actualmente en el listado habilitado para postulaciones. Para ello se establece seguir el patrón de diseño (ubicación y tipo de componentes que se colocan en los formularios), colores, y asemejarse lo más posible a la experiencia que se puede tener con la plataforma web.

La figura 27 muestra los componentes visuales que se desarrollan en XML con código Java que permite una manipulación de datos dinámica y mucho más robusta.

3.9 Pruebas de la solución

Con el fin de garantizar el correcto funcionamiento del sistema web y la aplicación móvil tanto independientes como en conjunto y de que los requerimientos solicitados por el cliente se cumplieron correctamente se realizaron diferentes tipos de pruebas que ayudaron a encontrar defectos y a aumentar el nivel de calidad.

3.9.1 Plan de pruebas.

Con el fin de verificar el correcto funcionamiento y de que el software cumplía con los requerimientos, se decide realizar 3 tipos de pruebas tanto para el sistema web como para la aplicación móvil:

Pruebas funcionales: Mediante la ejecución, revisión y retroalimentación a través de la técnica de caja negra de las funcionalidades previamente diseñadas, se busca evaluar, probar y validar cada una de las opciones con las que cuenta el sistema web y la aplicación móvil y de que el software hace lo que debe y sobre todo, lo que fue especificado.

Pruebas de rendimiento: Mediante estas pruebas se logra determinar lo rápido que realiza una tarea el sistema web y la aplicación móvil en condiciones particulares de trabajo, y ayuda a validar y verificar que se cumpla con los atributos de calidad, tales como la escalabilidad y uso de los recursos.

Pruebas de entendibilidad: Mediante estas pruebas se logra medir la aceptación y facilidad de uso del sistema web y la aplicación móvil.

3.9.1.1 Pruebas funcionales.

El objetivo de realizar estas pruebas, fue garantizar y comprobar que todos los requerimientos y funcionalidades solicitadas fueron atendidos en su totalidad y así constatar que todo esté completamente funcional.

En el anexo 5 se puede encontrar el detalle completo de las pruebas realizadas, las mismas que son las pruebas finales y que se realizaron el día de la aceptación del software mas no las pruebas que se desarrollaron al finalizar cada uno de los incrementos e iteraciones en cada sprint, obteniendo así un resumen con los siguientes resultados:

Tabla 2. Resumen de pruebas funcionales

Código Prueba	Aplicativo	Prueba satisfactoria		Fecha	Observaciones
		SI	NO		
001 - Comprobar registro de datos de estudiante	web	*		2017/06/28	
002 - Comprobar validación en login de usuario y contraseña	web	*		2017/06/28	
003 - Comprobar el acceso al sistema y a las interfaces asignadas a cada perfil de usuario	web	*		2017/06/28	
004 - Comprobar acceso a información,	web	*		2017/06/28	

dependiendo de los campus asignados por el administrador a cada usuario administrativo					
005 - Comprobar visualización de información únicamente de los campos asignados	web	*		2017/06/28	
006 - Comprobar incremento de periodo y cambiar estado de instituciones UPS	web	*		2017/06/28	
007 - Comprobar registro de usuarios, comprobar edición de usuarios, comprobar visualización de	web	*		2017/06/28	

usuarios del sistema					
008 - Comprobar registro de campus, comprobar edición de campus, comprobar visualización de campus	web	*		2017/06/28	
009 - Comprobar registro de campus, comprobar edición de campus, comprobar visualización de campus	web	*		2017/06/28	
010 - Comprobar registro de carreras, comprobar edición de	web	*		2017/06/28	

carreras, comprobar visualización de carreras					
011 - Comprobar registro de docentes, comprobar edición de docentes, comprobar visualización de docentes	web	*		2017/06/28	
012 - Comprobar registro la carga o subida de archivos en pdf	web	*		2017/06/28	
013 - Comprobar registro de tipos de institución, comprobar edición de tipos de institución, comprobar visualización de	web	*		2017/06/28	

tipos de institución					
014 - Comprobar registro de tipos de institución, comprobar edición de tipos de institución, comprobar visualización de tipos de institución	web	*		2017/06/28	
015 - Comprobar registro de instituciones UPS, comprobar edición de instituciones UPS , comprobar visualización de instituciones UPS	web	*		2017/06/28	
016 - Comprobar registro de instituciones externas,	web	*		2017/06/28	

comprobar edición de instituciones externas , comprobar visualización de instituciones externas					
017 - Comprobar registro de listas de estudiantes, comprobar carga de archivos, comprobar edición de listas de estudiantes, comprobar visualización de listas de estudiantes	web		*	2017/06/28	Adicionar docente al registro de la lista de estudiantes
018 - Comprobar si se aprueba o se niega la postulación a un estudiante,	web	*		2017/06/28	

comprobar el envío de notificación					
019 - Comprobar si se registra un estudiante no postulado a una institución externa	web	*		2017/06/28	
020 - Comprobar si actualiza el estado del proceso a bloqueado o completado en institución UPS	web	*		2017/06/28	
021 - Comprobar si actualiza el estado del proceso a bloqueado o completado en institución externa, comprobar el	web	*		2017/06/28	

envío de notificación					
022 - Comprobar si actualiza el estado del proceso a bloqueado o completado en institución externa, comprobar el envío de notificación	web	*		2017/06/28	
023 - Comprobar los reporte coinciden con la información generada en el sistema	web	*		2017/06/28	
024 - Visualización del proceso VAS, comprobar si se actualiza los datos del estudiante,	web	*		2017/06/28	

comprobar la búsqueda instituciones y registro de postulación, visualización de solicitudes registradas, comprobar la eliminación de una postulación, comprobar descarga de ficha de solicitud aprobada, comprobar descarga de formatos proporcionados por el usuario administrador, comprobar registro, edición de experiencia salesiana					
---	--	--	--	--	--

025 - Comprobar si se actualiza la nueva contraseña del usuario	web	*			
026 - Comprobar si envía un correo con su contraseña del sistema	web	*		2017/06/28	
027 - Comprobar validación en login de usuario y contraseña	móvil	*		2017/06/28	
028 - Comprobar la búsqueda instituciones y registro de postulación	móvil	*		2017/06/28	
029 - Comprobar registro, edición de experiencia salesiana	móvil	*		2017/06/28	
030 - Comprobar la recepción de notificaciones	móvil	*		2017/06/28	

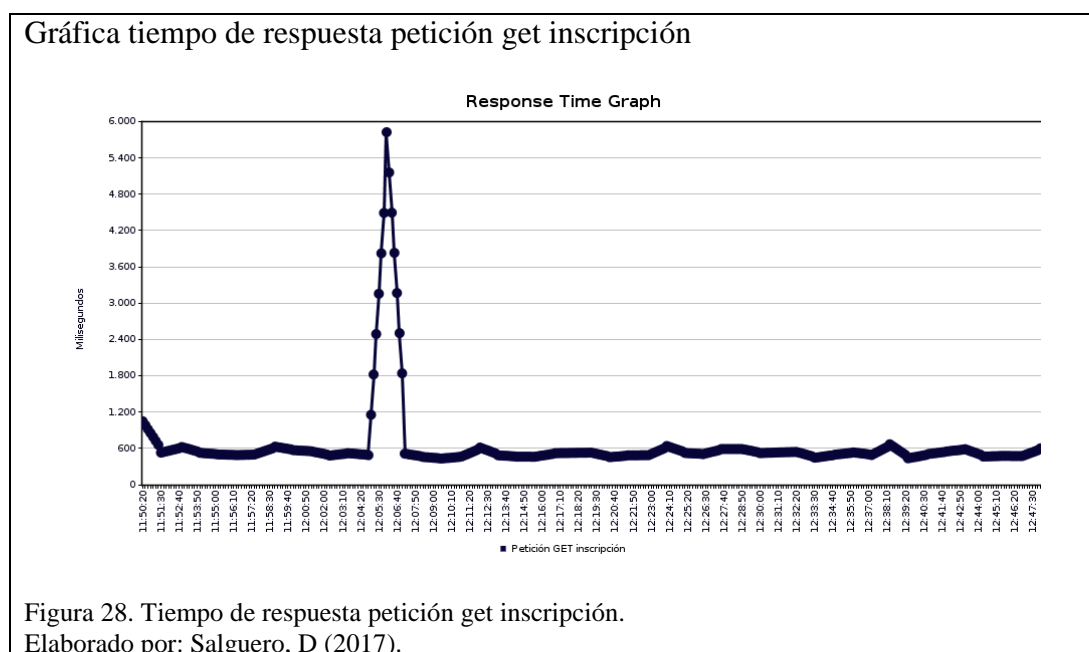
Nota: Tabla resumen de pruebas funcionales

Obteniendo así un 96.67% de satisfacción mediante un cálculo simple de porcentaje ya que de 30 pruebas 29 fueron satisfactorias en cuando a la funcionalidad tanto de la aplicación móvil como del sistema web.

3.9.1.2 Pruebas de rendimiento sistema web.

El objetivo de realizar estas pruebas, fue garantizar y verificar que el sistema web tenga un tiempo de respuesta considerable al momento de realizar las peticiones get y post, se realizaron las pruebas de rendimiento mediante JMeter que es una herramienta Java, que permite realizar pruebas de rendimiento y pruebas funcionales en sistemas web. Las pruebas se realizaron a funciones críticas enfocadas en las peticiones get y post que pueden consumir muchos recursos del servidor en las peticiones, al contar con poca concurrencia tanto de estudiantes como de usuarios administrativos se realizaron los siguientes casos prueba:

Caso 1. Con 50 usuarios en un periodo de 3600 segundos de subida en 1 repetición. Las gráficas muestran el tiempo de respuesta tanto de las peticiones get como post.



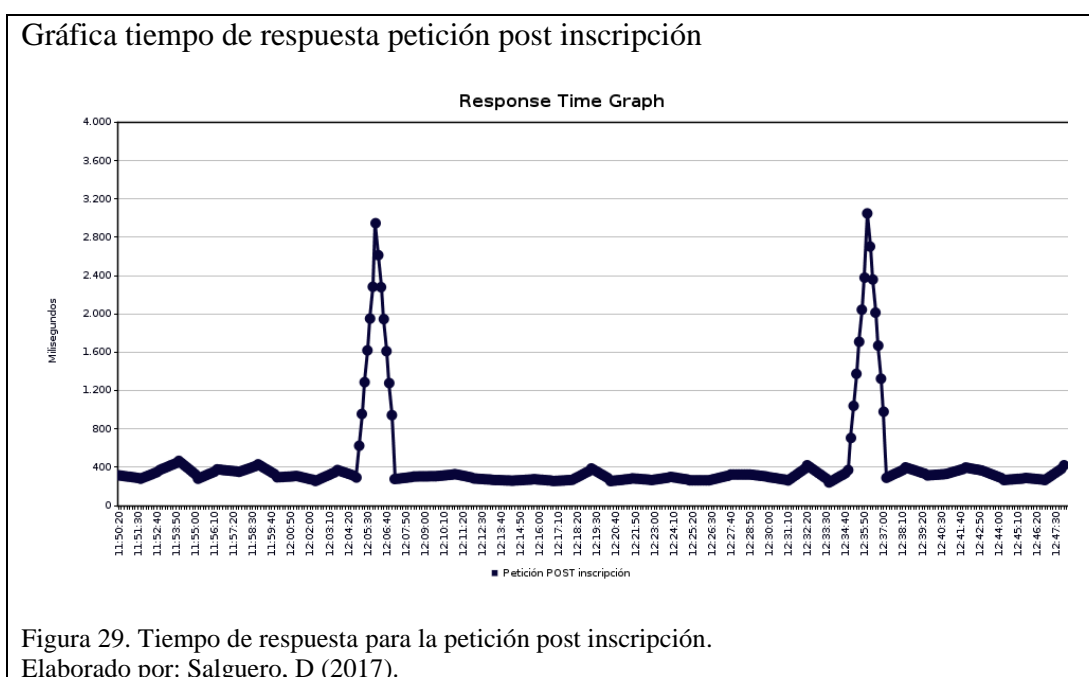
La figura 28 muestra el tiempo de respuesta para la petición get al momento de listar las instituciones UPS registradas en el sistema para que los estudiantes seleccionen y

postulen a cualquier institución de la lista generada, teniendo así un reporte general de la prueba de rendimiento con los siguientes resultados:

Tabla 3. Resultados generales de las peticiones get inscripción

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Petición get inscripción	50	636	431	5824	746,69	0%	1,417	0,26	0,00	18860,6
Total	50	636	431	5824	746,69	0%	1,417	0,26	0,00	18860,6

Nota: Tabla de resultados generales de las peticiones get inscripción



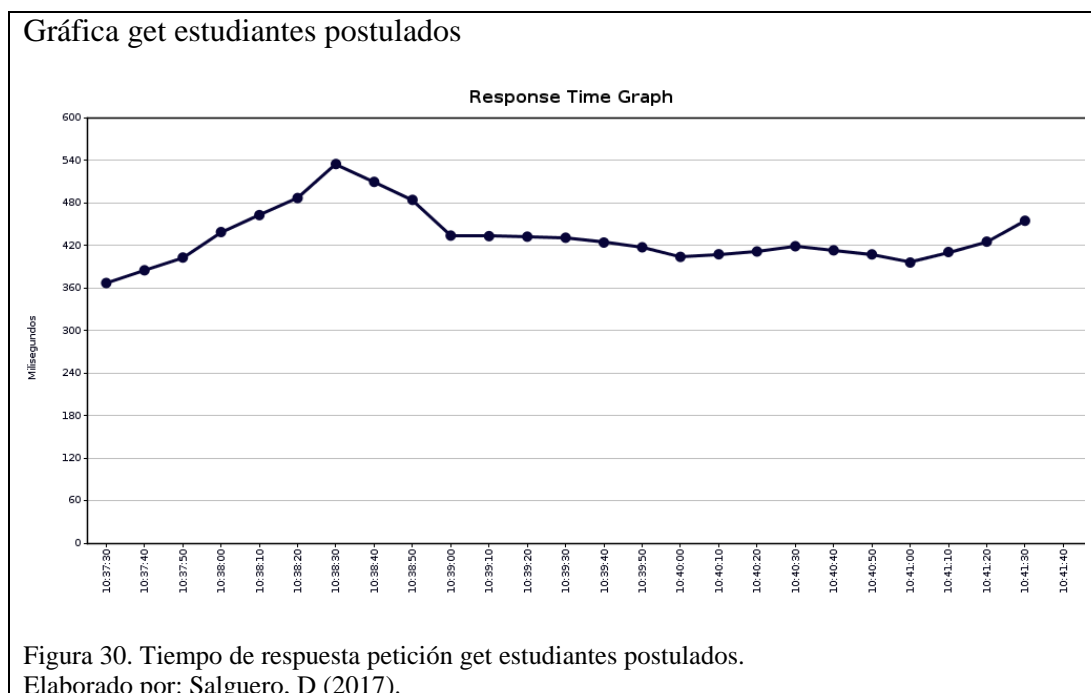
La figura 29 muestra el tiempo de respuesta para la petición post al momento de que el estudiante se postula a una institución UPS, teniendo así un reporte general de la prueba de rendimiento con los siguientes resultados:

Tabla 4. Resultados de las peticiones post inscripción

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Petición post inscripción	50	421	240	3049	528,66	0%	1,417	0,06	0,00	4461,9
Total	50	421	240	3049	528,66	0%	1,417	0,06	0,00	4461,9

Nota: Tabla de resultados de las peticiones post inscripción

Caso 2. Con 10 usuarios en un periodo de 300 segundos de subida en 1 repetición.

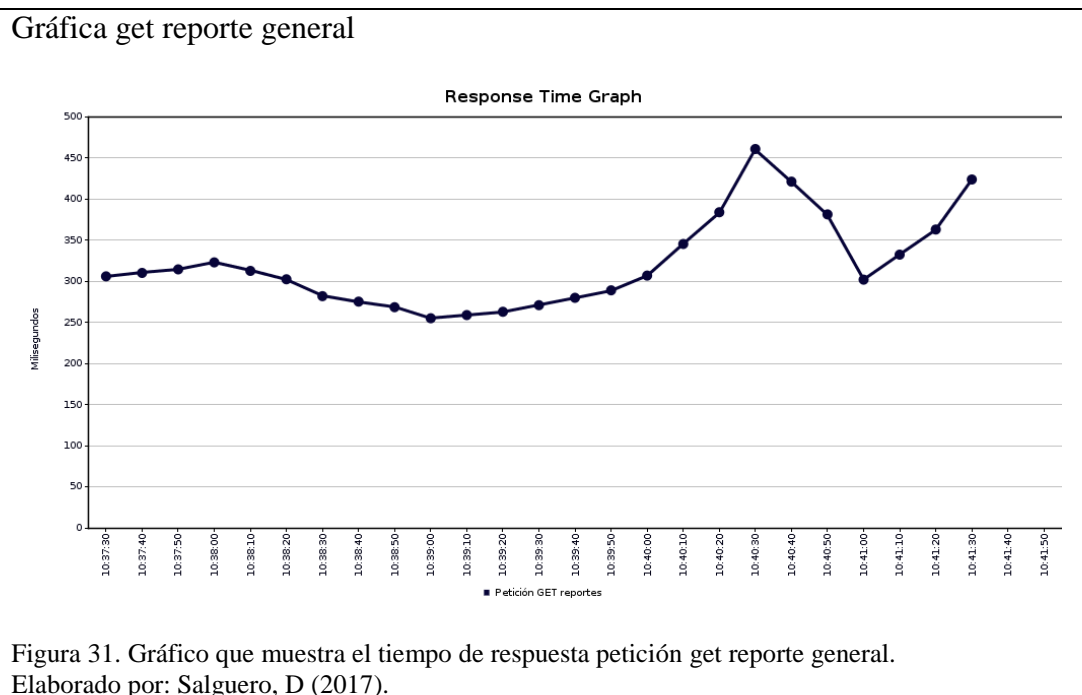


La figura 30 muestra el tiempo de respuesta para la petición get al momento de listar los estudiantes que se encuentran postulados, teniendo así un reporte general de la prueba de rendimiento con los siguientes resultados:

Tabla 5. Resultados de las peticiones get postulados

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Petición get postulados	10	422	344	535	49,49	0%	3,699	0,06	0,01	1706,0
Total	10	422	344	535	49,49	0%	3,699	0,06	0,01	1706,0

Nota: Tabla de resultados de las peticiones get postulados



La figura 31 muestra el tiempo de respuesta para la petición get al momento mostrar un reporte general, teniendo así un reporte general de la prueba de rendimiento con los siguientes resultados:

Tabla 6. Resultados de las peticiones get reportes

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Petición get reportes	10	319	255	461	65,62	0%	3,700	0,04	0,01	1189,5
Total	10	319	255	461	65,62	0%	3,700	0,04	0,01	1189,5

Nota: Tabla de resultados de las peticiones get reportes

3.9.1.3 Pruebas de rendimiento aplicación móvil.

La aplicación móvil no cuenta con base de datos local puesto que debe integrarse directamente al funcionamiento del sistema web que dispone de conexión directa a la base de datos única para el sistema, por este motivo se utiliza la llamada a servicios web que facilitan la conexión para recuperar y enviar datos desde los dispositivos móviles. El objetivo de realizar estas pruebas, fue garantizar y verificar el rendimiento en dispositivos de prueba para las peticiones get y post hacia el servidor ejecutando la aplicación en los 3 procesos críticos dentro de ella, los cuales son: consulta de vacantes con sus filtros, postulación a obra salesiana y el ingreso de la experiencia salesiana. Los datos y gráficas mostradas fueron obtenidos directamente desde el monitor de rendimiento de aplicaciones que ofrece Android Studio.

Para la ejecución de las pruebas se utilizaron 3 dispositivos, 1 de ellos es un smartphone personal físico, y los otros 2 fueron visualizados con la herramienta que ofrece el mismo Android Studio, el detalle técnico de los dispositivos utilizados se muestra a continuación.

Samsung Note 5 (Dispositivo Físico).

Tabla 7. Características Samsung Note 5

Nivel de aplicación	24
Versión de Sistema Operativo	7.0.0
Memoria RAM	4Gb

Nota: Tabla de características físicas Samsung Note 5

Nexus 5X.

Tabla 8. Características Nexus 5X

Nivel de aplicación	25
Versión de Sistema Operativo	6.0.1
Memoria RAM	1.5Gb

Nota: Tabla de características físicas Nexus 5X

Pixel

Tabla 9. Características Pixel

Nivel de aplicación	25
Versión de Sistema Operativo	7.1.1
Memoria RAM	2Gb

Nota: Tabla de características físicas Pixel

Prueba 1: Consulta de vacantes con filtros con dispositivo móvil modelo, Samsung

Note 5

Gráfica de recursos usados por aplicación en consulta de instituciones UPS.



Figura 32. Prueba 1 de rendimiento aplicación móvil.
Elaborado por: Gahona, R (2017).

La figura 32 muestra la cantidad de recursos usados por la aplicación al ejecutar una búsqueda en las Instituciones UPS con el modelo Samsung Note 5.

Prueba 2: Consulta de vacantes con filtros con dispositivo móvil modelo, Nexus 5X

Gráfica de recursos usados por aplicación en consulta de instituciones UPS.

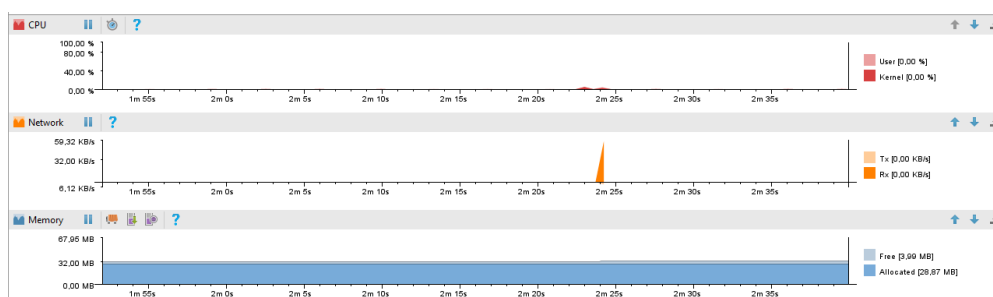


Figura 33. Prueba 2 de rendimiento aplicación móvil.

Elaborado por: Gahona, R (2017).

La figura 33 muestra la cantidad de recursos usados por la aplicación al ejecutar una búsqueda en las Instituciones UPS con el modelo Nexus 5X.

Prueba 3: Consulta de vacantes con filtros con dispositivo móvil modelo, Pixel

Gráfica de recursos usados por aplicación en consulta de instituciones UPS.

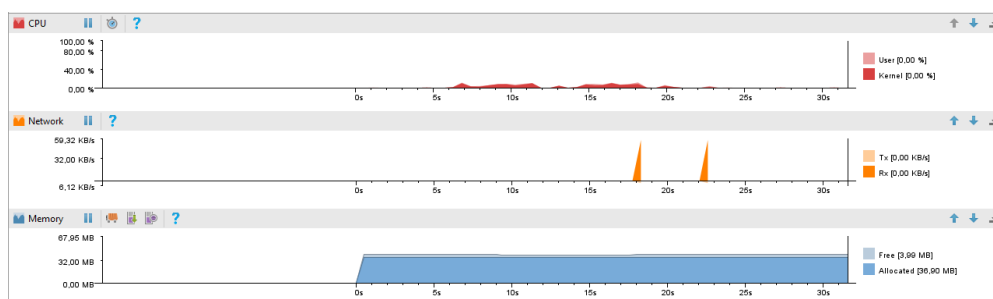


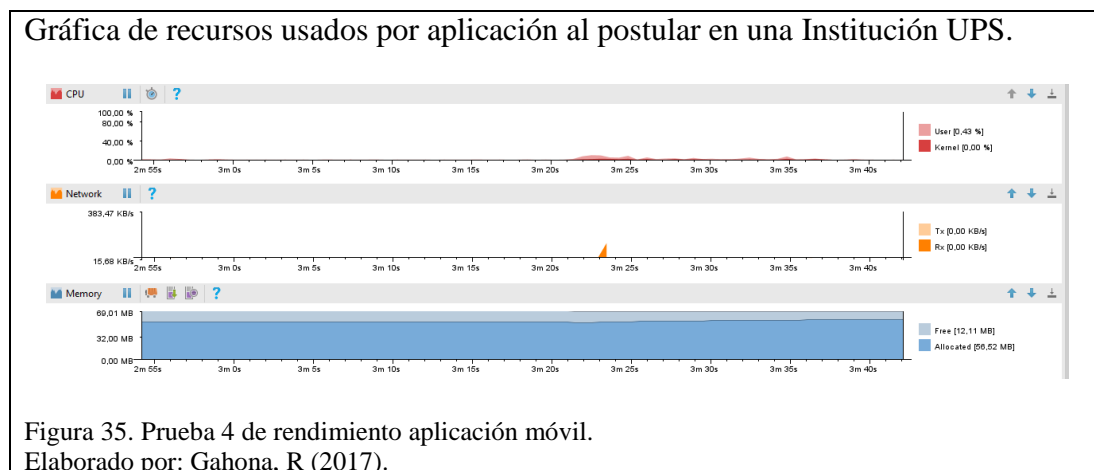
Figura 34. Prueba 3 de rendimiento aplicación móvil.

Elaborado por: Gahona, R (2017).

La figura 34 muestra la cantidad de recursos usados por la aplicación al ejecutar una búsqueda en las Instituciones UPS con el modelo Pixel, los recursos se denominan a la cantidad de Memoria, Procesamiento y Conexiones a la Red de datos que requiere cada actividad.

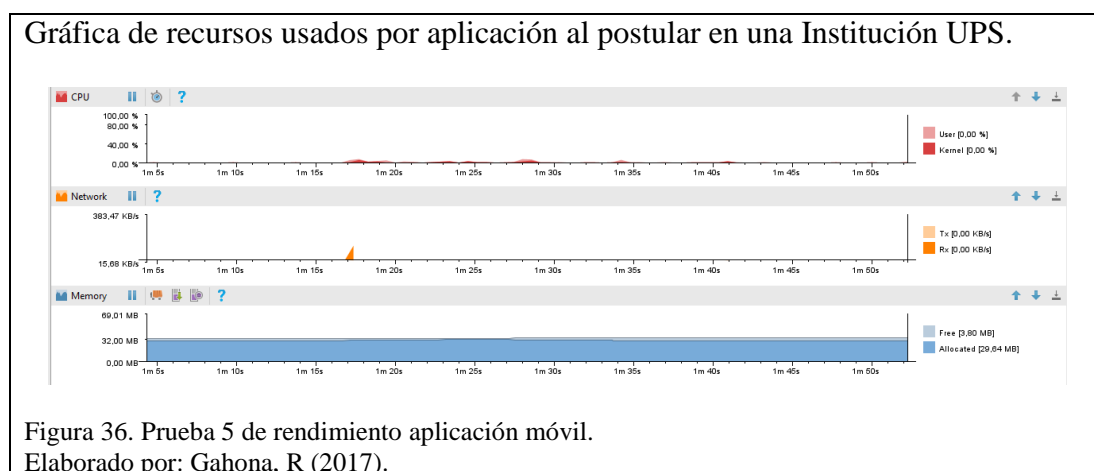
Se puede garantizar que si dispone de un dispositivo compatible con la aplicación puede disfrutar de un rendimiento óptimo al momento de realizar la consulta de vacantes.

Prueba 4: Postulación a una Institución UPS con el dispositivo modelo, Samsung Note 5.



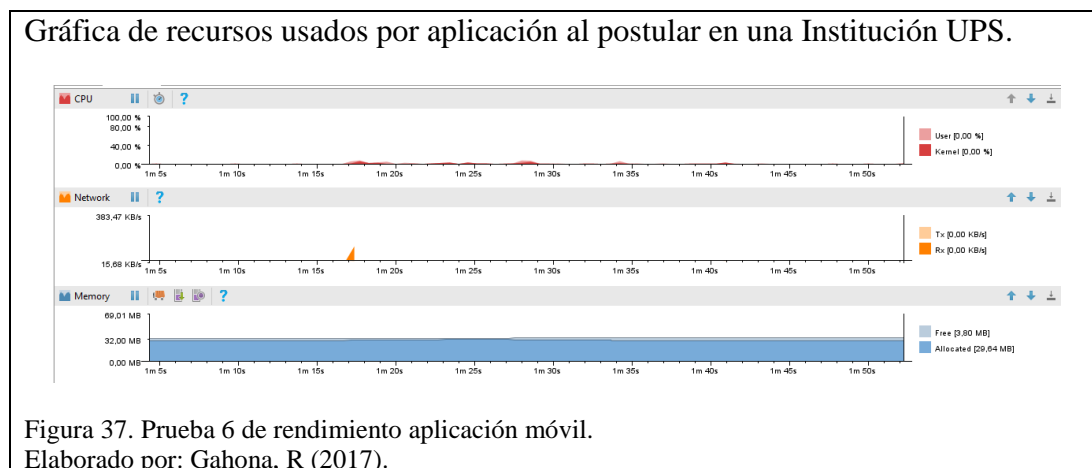
La figura 35 muestra la cantidad de recursos usados por la aplicación al ejecutar una postulación en las Instituciones UPS con el modelo Samsung Note 5.

Prueba 5: Postulación a una Institución UPS con el dispositivo modelo, Nexus 5X.



La figura 36 muestra la cantidad de recursos usados por la aplicación al ejecutar una postulación en las Instituciones UPS con el modelo Nexus 5X.

Prueba 6: Postulación a una Institución UPS con el dispositivo modelo, Pixel.



La figura 37 muestra la cantidad de recursos usados por la aplicación al ejecutar una postulación en las Instituciones UPS con el modelo Pixel.

Se puede evidenciar que al momento de utilizar la aplicación que su consumo de CPU no carga información persistente con almacenamiento local, el consumo de red que la aplicación genera es mínimo puesto que no tiene hilos (procesos en segundo plano que abran conexión no controladas que consuman datos) continuos que estén ejecutándose en segundo plano los mismos que generan grandes cantidades de consumos de red que puede afectar a la velocidad de procesamiento del dispositivo. Tomando en cuenta este dato se pudo asegurar que el consumo de megas cuando se utilice la aplicación móvil sin conexión a WIFI es eficiente y de bajo cargo.

Prueba 7: Envío de experiencia salesiana por parte del estudiante con el dispositivo Samsung Note 5.

Gráfica de recursos usados por aplicación al enviar y consultar la experiencia salesiana.

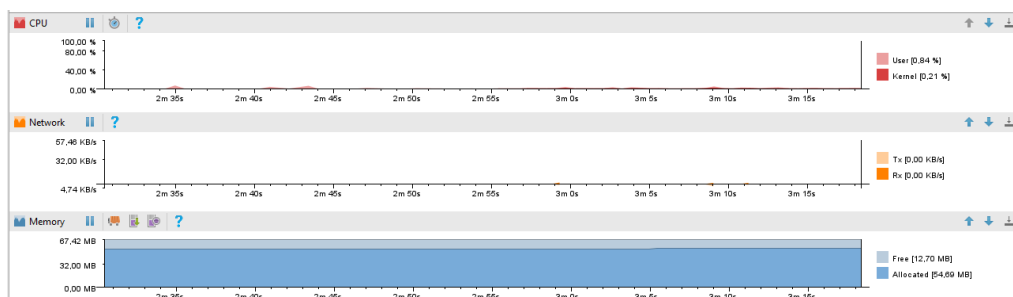


Figura 38. Prueba 7 de rendimiento aplicación móvil.
Elaborado por: Gahona, R (2017).

La figura 38 muestra la cantidad de recursos usados por la aplicación al ejecutar el registro de la experiencia salesiana por parte del estudiante con el modelo Samsung Note 5.

Prueba 8: Envío de experiencia Salesiana por parte del estudiante con el dispositivo Nexus 5X.

Gráfica de recursos usados por aplicación al enviar y consultar la experiencia salesiana.

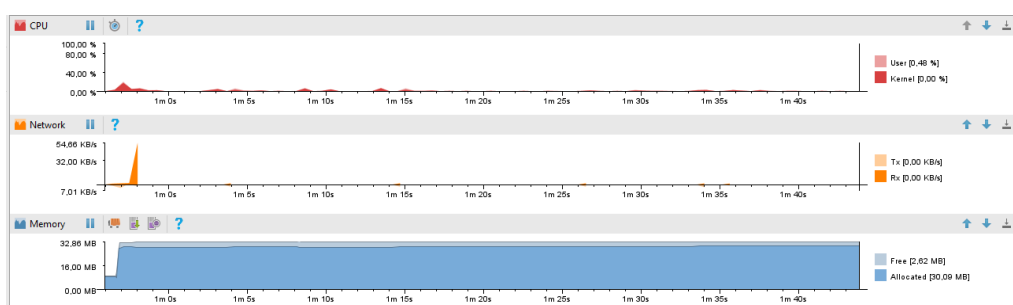


Figura 39. Gráfico prueba 8 de rendimiento aplicación móvil.
Elaborado por: Gahona, R (2017).

La figura 39 muestra la cantidad de recursos usados por la aplicación al ejecutar el registro de la experiencia salesiana por parte del estudiante con el modelo Nexus 5X.

Prueba 9: Envío de experiencia salesiana por parte del estudiante con el dispositivo Pixel.

Gráfica de recursos usados por aplicación al enviar y consultar la experiencia salesiana.

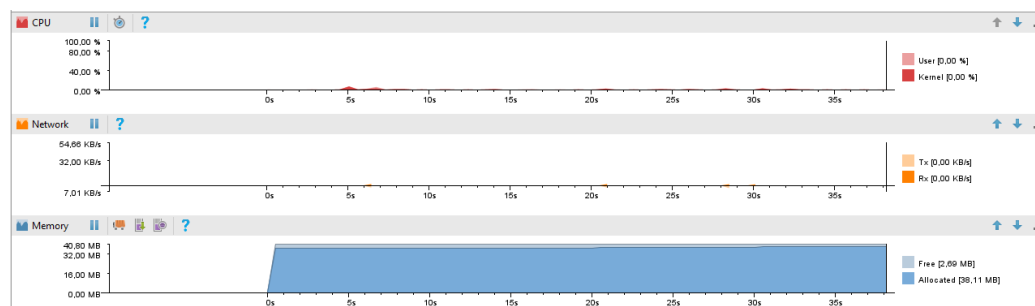


Figura 40. Prueba 9 de rendimiento aplicación móvil.

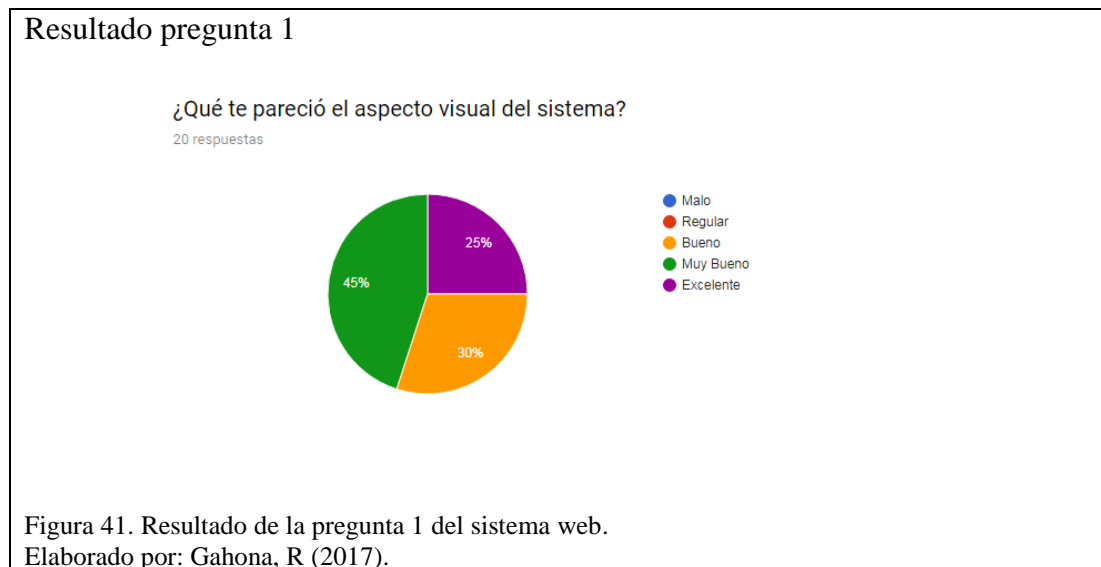
Elaborado por: Gahona, R (2017).

La figura 40 muestra la cantidad de recursos usados por la aplicación al ejecutar el registro de la experiencia salesiana por parte del estudiante con el modelo Pixel.

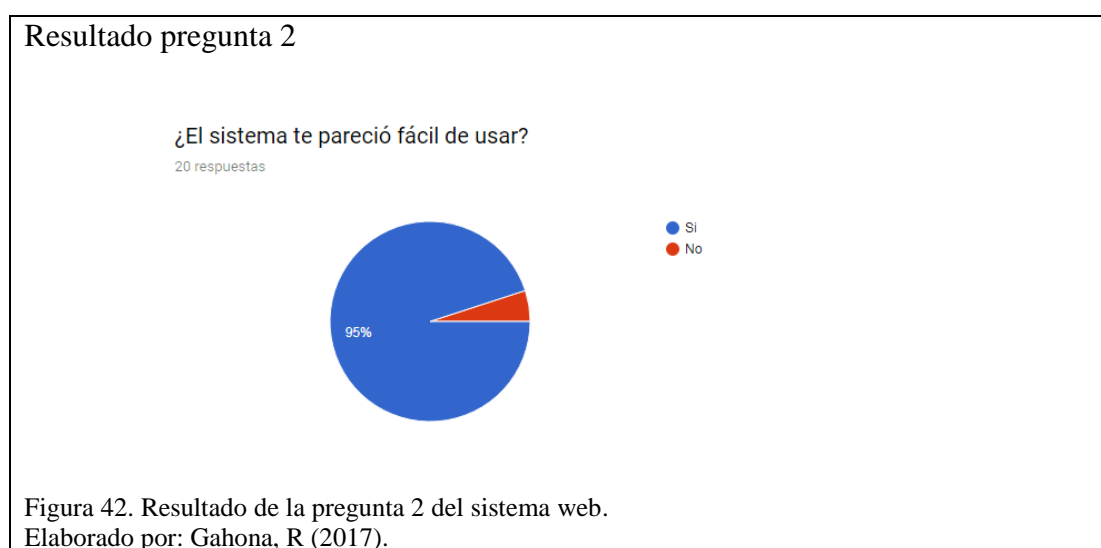
Se vuelve a evidenciar que el consumo de recursos en los dispositivos es muy bajo puesto que no se ejecutan procesos en segundo plano. Se puede aseverar que la aplicación móvil funciona de manera eficiente en cuenta a consumo de recurso del dispositivo móvil se trata, las pruebas fueron efectuadas de forma local, no obstante, cabe recalcar que al depender completamente de una conexión a la red para acceder a los servicios web, se utiliza el mismo gestor Android Studio puesto que cuenta con su propio ambiente de pruebas que permite la valoración en tiempo real al ejecutar la aplicación desarrollada ya sea en un dispositivo físico real, como en un dispositivo virtualizado, y se pudo validar que los tiempos de respuesta pueden verse poco afectados con una deficiente conexión a Internet.

3.9.1.4 Pruebas de entendibilidad sistema web.

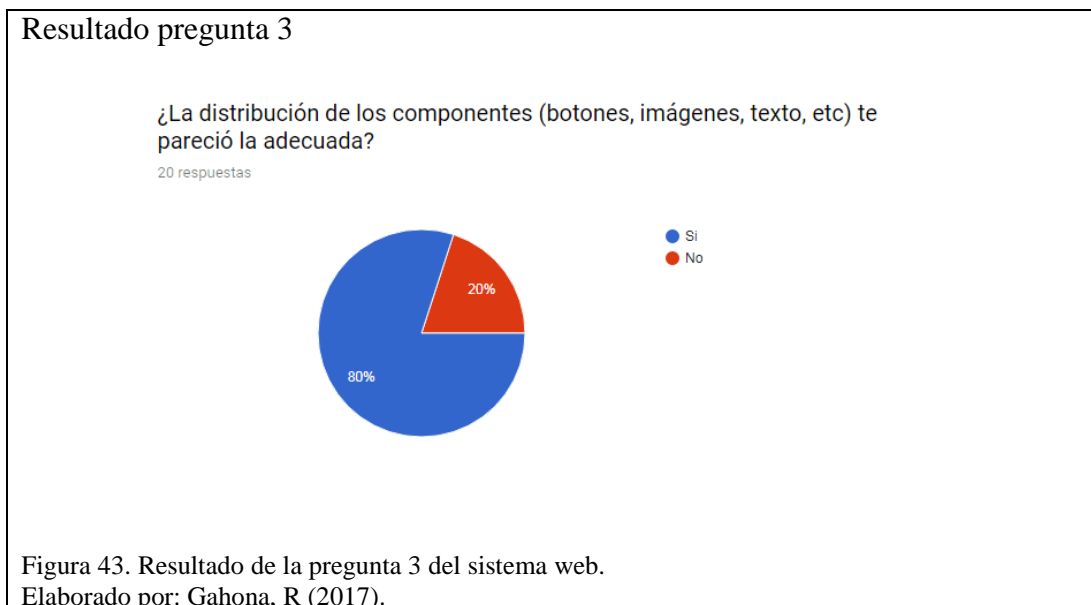
Se realiza una prueba a veinte usuarios con fin de evaluar la aceptabilidad del mismo, para lo cual, una vez que los usuarios utilizaron el sistema web, se procede a realizar una encuesta la misma que arroja los siguientes resultados:



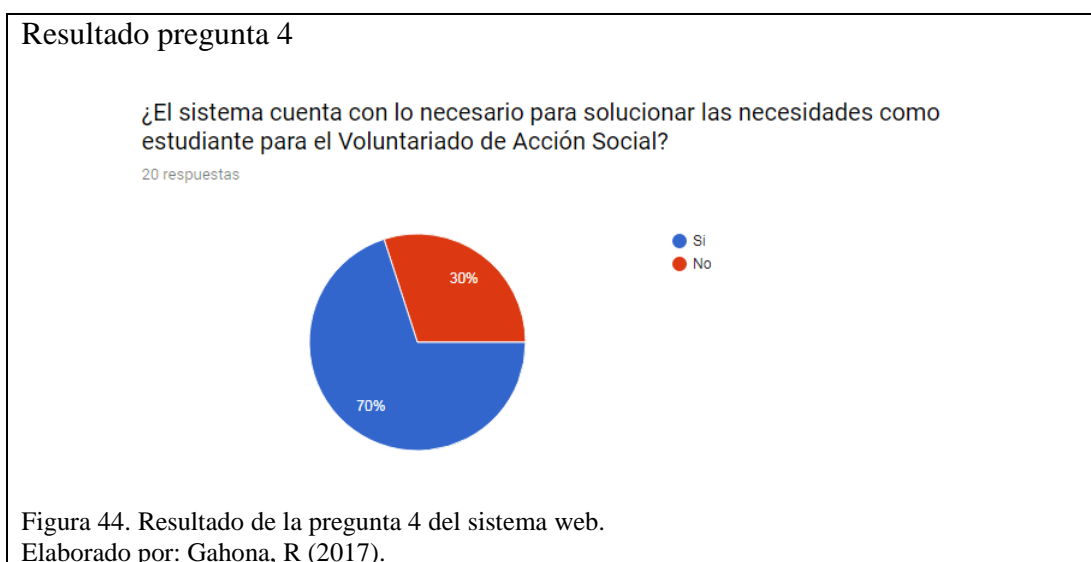
La figura 41 muestra el resultado realizado a la primera pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 45% de muy bueno y un 30% de bueno que los usuarios encontraron en el aspecto visual del sistema web.



La figura 42 muestra el resultado realizado a la segunda pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 95% que los usuarios afirman que el sistema web es fácil de usar.



La figura 43 muestra el resultado realizado a la tercera pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 80% que los usuarios afirman que la distribución de los componentes fue la adecuada en el sistema web.

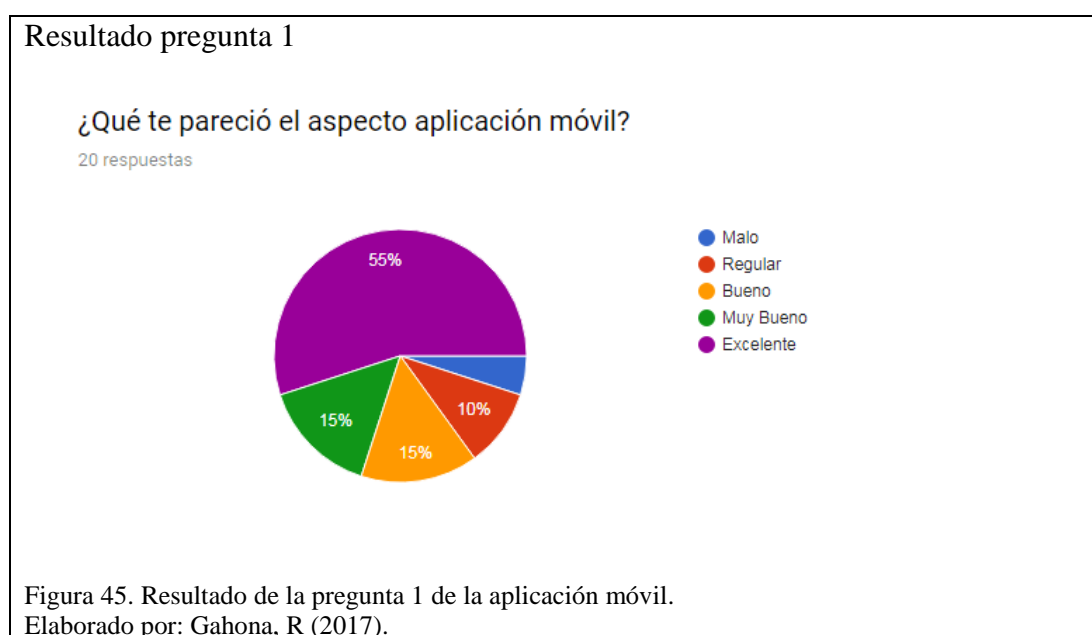


La figura 44 muestra el resultado realizado a la cuarta pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 70% que los usuarios

afirman que el sistema web cuenta con lo necesario para solucionar las necesidades del VAS.

3.9.1.5 Pruebas de entendibilidad aplicación móvil.

Se realiza una prueba a veinte usuarios con fin de evaluar la aceptabilidad del mismo, para lo cual, una vez que los usuarios utilizaron la aplicación móvil, se procede a realizar una encuesta la misma que arroja los siguientes resultados:



La figura 45 muestra el resultado realizado a la primera pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 55% de excelente y un 15% de muy bueno que los usuarios encontraron en el aspecto visual del sistema web.

Resultado pregunta 2

¿La aplicación móvil te pareció fácil de usar?

20 respuestas

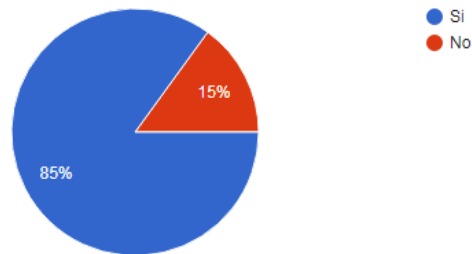


Figura 46. Resultado de la pregunta 2 de la aplicación móvil.
Elaborado por: Gahona, R (2017).

La figura 46 muestra el resultado realizado a la segunda pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 85% que los usuarios afirman que la aplicación móvil es fácil de usar.

Resultado pregunta 3

¿La distribución de los componentes (botones, imágenes, texto, etc) te pareció la adecuada?

20 respuestas

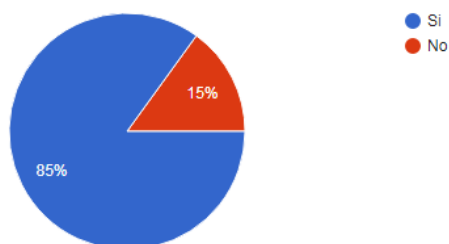


Figura 47. Resultado de la pregunta 3 de la aplicación móvil.
Elaborado por: Gahona, R (2017).

La figura 47 muestra el resultado realizado a la tercera pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 85% que los usuarios afirman que la distribución de los componentes fue la adecuada en la aplicación móvil.

Resultado pregunta 4

¿El aplicación móvil cuenta con lo necesario para solucionar las necesidades como estudiante para el Voluntariado de Acción Social?

20 respuestas

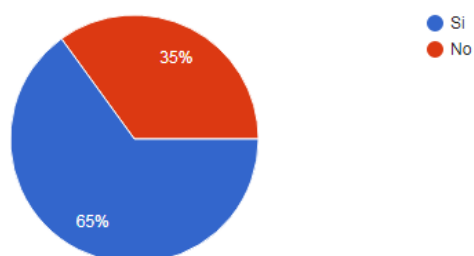


Figura 48. Resultado de la pregunta 4 de la aplicación móvil.
Elaborado por: Gahona, R (2017).

La figura 48 muestra el resultado realizado a la cuarta pregunta de la encuesta, la misma que indica mediante un cálculo simple de porcentaje el 65% que los usuarios afirman que la aplicación móvil cuenta con lo necesario para solucionar las necesidades del VAS.

CONCLUSIONES

- La falta de madurez en la gestión de procesos para el manejo del VAS afectó al desarrollo del software, ya que al no saber o entender cómo se manejan las actividades o tareas en el proceso, ocasionó una mayor dificultad al tratar automatizar los mismos mediante un software.
- El involucramiento y compromiso del cliente en cada una de las fases del desarrollo del proyecto fue vital para cumplir con el objetivo general planteado al inicio del mismo.
- La utilización de servicios web es una solución eficiente, debido a que, con esta tecnología, se consigue evitar la duplicación de datos al contar con una sola conexión, con lo que se logra una optimización de los recursos.
- El uso de software para la ejecución del proceso VAS genera optimización de los recursos ya que a comparación de un proceso manual, toda la información se centraliza y gestiona en un solo sistema al cual todos los interesados tienen acceso.

RECOMENDACIONES

- Ejecutar el sistema web con el navegador Google Chrome para garantizar un correcto funcionamiento de los componentes gráficos desarrollados en con BootStrap 4.0 conjuntamente con Angular JS.
- Utilizar dispositivos móviles Android con sistema operativo 4.2.2 Jelly Bean en adelante para garantizar un correcto funcionamiento de la aplicación móvil.
- Utilizar Apache 2 v2.4, MySQL v5.7 y PHP v7.0 para la implementación (despliegue) del sistema web ya que este cuenta con características, programación y componentes específicos que no funcionan con versiones anteriores de estas herramientas.
- Utilizar Google Play Store para la distribución de la aplicación móvil ya que tiene un costo de \$25.00 y si se lo hace por otro medio cada estudiante en su dispositivo Android deberá activar la opción de orígenes desconocidos para instalar la aplicación y esto puede generar molestias.

GLOSARIO DE TÉRMINOS

UPS: Universidad Politécnica Salesiana.

TI: Tecnologías de la Información.

VAS: Voluntariado de Acción Social.

Scrum: Proceso de desarrollo de software iterativo y creciente.

Product owner: Persona que representa a los interesados internos y externos.

Scrum master: Facilita la aplicación de Scrum y gestiona cambios.

Front-End: Parte de la aplicación que interactúa con el usuario final.

Back-End: Parte de la aplicación que procesa las peticiones del Front-End.

TIC's: Tecnologías de información y comunicación.

API: Funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro software.

Token: Código alfanumérico de longitud variable y de asignación única en los dispositivos, identificador utilizado por el servidor de notificaciones.

Push: Es una forma de comunicación en la que una aplicación servidora envía un mensaje a un cliente-consumidor. Es decir, es un mensaje que un servidor envía a una persona alertándolo de que tiene una información nueva.

Key: Identificador alfanumérico que otorga firebase para el uso de sus servicios a un proyecto.

LISTA DE REFERENCIAS

- Abiztar. (s.f.). *Casos a incluir y casos a extender*. Recuperado el 06 de octubre de 2017, de <http://www.abiztar.com.mx/articulos/casos-a-incluir-casos-a-extender.html>
- Albasanz. (s.f.). *Manual de XML*. Recuperado el 18 de septiembre de 2017, de XML Que es?: <http://www.mundolinux.info/que-es-xml.htm>
- Alberto Basalo, M. A. (28 de Agosto de 2014). *desarrolloweb.com*. Obtenido de Qué es AngularJS: <https://desarrolloweb.com/articulos/que-es-angularjs-descripcion-framework-javascript-conceptos.html>
- Alvarez, M. A. (23 de Noviembre de 2009). *desarrolloweb.com*. Obtenido de CodeIgniter: <https://desarrolloweb.com/articulos/codeigniter.html>
- AngularJS. (s.f.). *AngularJS*. Recuperado el 9 de Febrero de 2017, de <https://angularjs.org/>
- Apache. (s.f.). *Apache JMeter*. Recuperado el 18 de septiembre de 2017, de Apache JMeter TM: <http://jmeter.apache.org/>
- Bahit, E. (diciembre de 2014). *POO y MVC en PHP*. Recuperado el 17 de septiembre de 2017, de El paradigma de la Programación Orientada a Objetos en PHP y el patrón de arquitectura de Software MVC: <http://www1.herrera.unt.edu.ar/biblcet/wp-content/uploads/2014/12/eugeniabahitpooymvcenphp.pdf>
- BBVAOPEN4U. (23 de marzo de 2016). *BBVA API MARKET*. Recuperado el 18 de septiembre de 2017, de API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos: <https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos>
- Cardona, M. P. (14 de octubre de 2016). *SEO Y SEM*. Recuperado el 17 de septiembre de 2017, de Firebase, qué es y para qué sirve la plataforma de Google: <http://www.iebschool.com/blog/firebase-que-es-para-que-sirve-la-plataforma-desarrolladores-google-seo-sem/>
- CEPE/ONU. (2012). *Guia de Implementacion de la Facilitacion de Comercio*. Recuperado el 11 de septiembre de 2017, de Proceso "Tal Como Está" (As is): <http://tfig.unece.org/SP/contents/as-is-process.htm>
- Cordova, J. (12 de Septiembre de 2016). *Solución informática para mejorar la eficacia en el proceso de registro de evaluación y asistencia del instituto de educación superior tecnológico público Chocope, a partir del año 2015*. Obtenido de Universidad Nacional de Trujillo: <http://dspace.unitru.edu.pe/handle/UNITRU/7851>
- ddsutn. (2012). *Introducción a arquitectura Web*. Recuperado el 21 de Febrero de 2017, de Arquitectura Web: <http://ddsutn.com.ar/cursos/cursadas->

anteriores/miercoles-a-la-noche-
2012/seguimientoclasesmienoches2012/introduccion-web

develapps. (s.f.). *Modelo Vista Presentador*. Recuperado el 03 de agosto de 2017, de <http://www.develapps.com/es/noticias/modelo-vista-presentador-mvp-en-android>

Developers. (s.f.). *Manifiesto de al app*. Recuperado el 05 de agosto de 2017, de <https://developer.android.com/guide/topics/manifest/manifest-intro.html?hl=es-419>

Diputacion Foral de Bizkaia. (Junio de 2006). *Voluntariado Hechar una Mano*. Recuperado el 5 de septiembre de 2017, de <http://ibdigital.uib.es/greenstone/collect/cd2/index/assoc/responsa/ble1.dir/responsable1.pdf>

Firebase. (s.f.). *Firebase*. Recuperado el 17 de septiembre de 2017, de Firebase te ayuda a crear mejores apps para dispositivos móviles y hacer crecer tu empresa.: <https://firebase.google.com/?hl=es-419>

Gardey, J. P. (2013). *definicion*. Obtenido de VALIDACIÓN: <http://definicion.de/validacion/>

Globe. (2017). *Pruebas de caja negra*. Recuperado el 23 de octubre de 2017, de Testing funcional a nivel de datos externos: <https://www.globetesting.com/2012/08/pruebas-de-caja-negra/>

Gutierrez, A. (17 de agosto de 2015). *¿Qué es una app y cómo descargarlas?* Obtenido de About en español: <http://windowsespanol.about.com/od/AccesoriosYProgramas/f/Que-Es-Una-App.htm>

Gutierrez, C. (2014). *¿Para qué sirve el Scrum en la Metodología Ágil?* Obtenido de Intelligence Business: <http://www.i2btech.com/blog-i2b/tech-deployment/para-que-sirve-el-scrum-en-la-metogologia-agil/>

IBM. (11 de 11 de 2011). *InfoSphere Master Data Management Collaboration Server 10.0.0*. Recuperado el 10 de septiembre de 2017, de Definición de casos de uso: https://www.ibm.com/support/knowledgecenter/es/SS2U2U_10.0.0/com.ibm.pim.dev.doc/pim_tsk_arc_definingusecases.html

JSON. (s.f.). *Introducción a JSON*. Recuperado el 18 de septiembre de 2017, de <http://www.json.org/json-es.html>

JSON.ORG. (s.f.). *Introducing JSON*. Obtenido de JSON.ORG: <http://www.json.org/>

Lisset, U. (1 de Septiembre de 2017). *Sistema de planificación, ejecución y control del proceso de desarrollo de clases en la carrera profesional de medicina humana en la modalidad no presencial de la universidad privada antenor orrego de trujillo utilizando el LMS Moodle y un marco metodológ.* Obtenido de Repositorio: <http://repositorio.upao.edu.pe/handle/upaorep/2814>

- Merino, J. P. (2014). *definicion*. Obtenido de DEFINICIÓN DE VERIFICACIÓN: <http://definicion.de/verificacion/>
- Microsoft. (2015). *Documentacion*. Recuperado el 11 de septiembre de 2017, de Diagramas de casos de uso de UML: Instrucciones: <https://msdn.microsoft.com/es-es/library/dd409432.aspx>
- Montenegro, J. C. (13 de Agosto de 2016). *jovenes y desarrollo*. Recuperado el 15 de Febrero de 2017, de Voluntariado Salesiano: Una perspectiva desde la juventud: <http://www.jovenesydesarrollo.org/actualidad/voluntariado-salesiano-una-perspectiva-juventud>
- Motos, V. (7 de junio de 2014). *Analiza las bases de datos SQLite de tu smartphone Android con ADEL (Android Data Extractor Lite)* . Obtenido de Hack Players: <http://www.hackplayers.com/2014/06/analiza-sqlite-android-con-ADEL.html>
- Pardo, F. O. (22 de Agosto de 2008). *voltairenet.org*. Recuperado el 9 de Febrero de 2017, de Grupos de atención prioritaria en la nueva Constitución: <http://www.voltairenet.org/article157889.html>
- Pavón, P. P., & Pérez, D. (2003). La evaluación en los cursos online. *Departamento Pedagógico del Instituto de Formación Online*. Recuperado el 1 de Septiembre de 2017, de <http://e-spacio.uned.es/fez/eserv.php?pid=bibliuned:1464&dsID=n03pavonVE00.pdf>
- PEI INSTITUCION EDUCATIVA LA LEONA. (s.f.). *INSTITUCION EDUCATIVA LA LEONA*. Recuperado el 21 de Febrero de 2017, de <https://institucioneducativaleona.wordpress.com/academica/seguimiento-academico/>
- Proyectos ágiles.org. (s.f.). *Ejecución de la iteración (Sprint)*. Obtenido de Proyectos ágiles.org: <https://proyectosagiles.org/ejecucion-iteracion-sprint/>
- Quin, L. (11 de 10 de 2016). *W3*. Recuperado el 18 de septiembre de 2017, de Extensible Markup Language (XML): <https://www.w3.org/XML/>
- Revelo, J. (20 de Octubre de 2014). *Tutorial De Bases De Datos SQLite En Aplicaciones Android*. Obtenido de Hermosa programación: <http://www.hermosaprogramacion.com/2014/10/android-sqlite-bases-de-datos/>
- REYES, J. E. (6 de Febrero de 2013). Obtenido de Las TICs y la Gestión empresarial: <http://www.eoi.es/blogs/mtelcon/2013/02/06/las-tics-y-la-gestion-empresarial/>
- RODRÍGUEZ, A. E. (s.f.). *JSON I - ¿QUÉ ES Y PARA QUÉ SIRVE JSON?* Recuperado el 18 de septiembre de 2017, de <https://geekytheory.com/json-i-que-es-y-para-que-sirve-json/>
- Saffirio, M. (4 de julio de 2009). *Tecnologías de la Información y Procesos de Negocios (BPM)*. Recuperado el 12 de septiembre de 2017, de As-is; To-Be; Gap: <https://msaffirio.wordpress.com/2009/07/04/as-is-to-be-gap/>

- Sánchez, J. (2004). *cartagena99.com*. Recuperado el 21 de Febrero de 2017, de MySQL:
<http://www.cartagena99.com/recursos/programacion/apuntes/mysql.pdf>
- Terrera, G. (26 de febrero de 2016). *Noticias*. Recuperado el 23 de octubre de 2017, de PRUEBAS DE CAJA NEGRA Y UN ENFOQUE PRÁCTICO:
<https://testingbaire.com/pruebas-caja-negra-enfoque-practico/>
- Universidad a distancia de Madrid. (s.f.). *Manual de XML*. Obtenido de Area de programación y desarrollo: <http://www.mundolinux.info/que-es-xml.htm>
- Universidad Politecnica Salesiana. (s.f.). *Universidad Politecnica Salesiana*. Recuperado el 9 de Febrero de 2017, de Razon de Ser: <http://www.ups.edu.ec/web/guest/razon-de-ser>
- xatakandroid. (8 de febrero de 2011). *¿Qué es Android?* Recuperado el 15 de Febrero de 2017, de <https://www.xatakandroid.com/sistema-operativo/que-es-android>