

**DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA AUTÓNOMA PARA
TRAMITES Y CONSULTAS DE LA NOTARIA ÚNICA DE SAN JACINTO BOLÍVAR**

JAVIER ANDRÉS CASTAÑEDA PÉREZ
2006119016

HOLMAN ALONSO ORTEGA MERCADO
2006119049

ING. LUIS GARRIDO
DIRECTOR



UNIVERSIDAD DEL **M**MAGDALENA

FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA ELECTRÓNICA
SANTA MARTA COLOMBIA

2017

AGRADECIMIENTOS

Le agradezco a Dios por haberme acompañado y guiado a lo largo de mi carrera, por ser mi fortaleza en los momentos de debilidad y por brindarme una vida llena de aprendizajes, experiencias y sobre todo felicidad.

Le doy gracias a mis padres, Ercília Pérez y Julio Castañeda por los valores que me han inculcado y por haberme dado la oportunidad de tener una excelente educación en el transcurso de mi vida. A mis hermanas, Anney y Jenney, a mi familia por su apoyo y amor incondicional, a los profesores que hicieron parte de mi formación profesional y a todas las personas que me acompañaron estos años de mi vida.

JAVIER ANDRÉS CASTAÑEDA PÉREZ

Mis agradecimientos y dedicates van dirigidas primero a Dios por permitirme alcanzar este objetivo, a mis padres, por su apoyo incondicional y constante motivación, a mis hermanos por el constante cariño, apoyo paciencia y tolerancia, en las buenas y en las malas, brindándome la confianza de seguir adelante y continuar para cumplir mis metas.

También quiero agradecer a todos los amigos y compañeros que siempre estuvieron allí, con quienes he compartido las luchas y victorias y sacrificios en el desarrollo de la carrera.

También agradecerle a universidad del magdalena, a la facultad de Ingeniería Electrónica, a su cuerpo docente, que son la columna vertebral en la formación de profesionales íntegros y con grandes capacidades.

HOLMAN ALONSO ORTEGA MERCADO

TABLA DE CONTENIDO

DESCRIPCIÓN DEL PROYECTO	7
PLANTEAMIENTO DEL PROBLEMA.....	7
<i>Preguntas Problema de Investigación</i>	8
ESTADO DE DESARROLLO O ANTECEDENTES	8
JUSTIFICACIÓN.....	11
IMPACTO ESPERADO DEL PROYECTO	12
OBJETIVOS.....	13
<i>General</i>	13
<i>Específicos</i>	13
DISEÑO METODOLÓGICO	14
<i>Tipo y diseño de investigación</i>	14
<i>Técnicas de Recolección de Datos</i>	14
<i>Técnicas de Investigación</i>	14
<i>Descripción del instrumento utilizado</i>	15
ALCANCES Y LIMITACIONES	16
<i>Limitaciones</i>	16
MARCO TEÓRICO	17
APLICACIONES INFORMÁTICAS	17
<i>Arquitectura de aplicaciones</i>	17
<i>Modelo Cliente / Servidor</i>	18
<i>Aplicaciones web</i>	22
<i>Sistema Operativo Móvil</i>	25
<i>Lenguaje Unificado de Modelado (UML)</i>	26
<i>Bases de Datos</i>	29
<i>PHP</i>	31
<i>JavaScript</i>	40
<i>Hojas de estilo en cascada (CSS - Cascading Style Sheets)</i>	51
<i>Ionic Framework</i>	55
<i>JSON</i>	56
<i>Metodología RUP</i>	59

DESARROLLO DEL PROYECTO.....	64
RESULTADOS	84
CONCLUSIONES.....	88
RECOMENDACIONES.....	90
PRESUPUESTO	91
BIBLIOGRAFÍA.....	92
ANEXOS.....	94

ÍNDICE DE FIGURAS

FIGURA 2.3.1 MODELO CLIENTE/SERVIDOR FUENTE: (PLATTINI, CALVO-MANZANO, CERVERA, & FERNANDÉZ, 2004)	18
FIGURA 2.3.2. EL MODELO CLIENTE-SERVIDOR SOBRE UNA RED. FUENTE: (TANENBAUM, 2009)	19
FIGURA 2.3.3 SEPARACIÓN DE FUNCIONES. FUENTE: (LUJÁN MORA, 2002).	19
FIGURA 2.3.4 PRESENTACIÓN DISTRIBUIDA. FUENTE: (LUJÁN MORA, 2002).....	20
FIGURA 2.3.5 APLICACIÓN DISTRIBUIDA. FUENTE: (LUJÁN MORA, 2002).	21
FIGURA 2.3.6 DATOS DISTRIBUIDOS. FUENTE: (LUJÁN MORA, 2002).	21
FIGURA 2.3.7 ARQUITECTURA DE TRES NIVELES. FUENTE: (LUJÁN MORA, 2002).....	22
FIGURA 2.3.8 ESQUEMA BÁSICO DE UNA APLICACIÓN WEB. FUENTE: (LUJÁN MORA, 2002).....	23
FIGURA 2.3.9 TECNOLOGÍAS EMPLEADAS EN EL CLIENTE Y SERVIDOR WEB. FUENTE: (LUJÁN MORA, 2002).....	24
FIGURA 2.3.10 ENTORNO WEB. FUENTE: (KIOSKEA, 2014)	25
FIGURA 2.3.11 LOGOS APPLE Y ANDROID. FUENTE:(HTTP://PACMAC.ES).....	26
FIGURA 2.3.12 SÍMBOLO UML DE UNA CLASE. FUENTE: (SCHMULLER, 2001).	27
FIGURA 2.3.13 EL SÍMBOLO UML DEL OBJETO. FUENTE: (SCHMULLER, 2001).....	28
FIGURA 2.3.14 DIAGRAMA DE CASOS DE USO UML. FUENTE: (SCHMULLER, 2001).....	29
FIGURA 2.3.15 ARCHIVO PHP. (CAPTURA DE PANTALLA)	32
FIGURA 2.3.16 EJEMPLO PHP. (CAPTURA DE PANTALLA).....	33
FIGURA 2.3.17 EJEMPLO FORMULARIO PHP. FUENTE: (VELÁZQUEZ MEDINA & HUEYO CHINO, 2015).	36
FIGURA 2.3.18 APARIENCIA DEL FORMULARIO. FUENTE: (VELÁZQUEZ MEDINA & HUEYO CHINO, 2015).	36
FIGURA 2.3.19 IMPRESIÓN EN PANTALLA. FUENTE: (VELÁZQUEZ MEDINA & HUEYO CHINO, 2015).....	37
FIGURA 2.3.20 DIRECTORIO DE PROYECTO EN LARAVEL. FUENTE: (HOSTALIAWHITEPAPERS, 2016).	39
FIGURA 2.3.21 VISUAL STUDIO CODE. FUENTE (RIVERO DORTA, 2016)	42
FIGURA 2.3.22 SUBLIME TEXT. FUENTE: (RIVERO DORTA, 2016)	43
FIGURA 2.3.23 WEBSTORM. FUENTE:(RIVERO DORTA, 2016).....	43
FIGURA 2.3.24 LOGO DE JQUERY.	49
FIGURA 2.3.25 LOGO DE JQUERY MOBILE.	50
FIGURA 2.3.26 CONSTRUCCIÓN DE UN JSON OBJECT. FUENTE: (INTRODUCING JSON, 2008).	56
FIGURA 2.3.27 CONSTRUCCIÓN DE UN JSON ARRAY. FUENTE: (INTRODUCING JSON, 2008).	57
FIGURA 2.3.28 FORMACIÓN DE UN STRING. FUENTE: (INTRODUCING JSON, 2008).....	57
FIGURA 2.3.29 OPCIONES VÁLIDAS PARA LA FORMACIÓN DE UN VALUE EN JSON. FUENTE: (INTRODUCING JSON, 2008).	58
FIGURA 2.3.30 FORMATO DE NUMBER EN JSON. FUENTE: (INTRODUCING JSON, 2008).	58
FIGURA 2.3.31 CICLO DE VIDA DE LA MITOLOGÍA RUP. FUENTE: (HTTPS://JUMMP.FILES.WORDPRESS.COM/2011/04/RUP_ESPANOL.GIF)	59

FIGURA 3.0.1 DISEÑO LÓGICO DEL SISTEMA. (DISEÑO PROPIO)	64
FIGURA 3.0.2 ARDUINO MEGA2560. FUENTE: (HTTPS://WWW.ARDUINO.CC).....	67
FIGURA 3.0.3 ARDUINO ETHERNET SHIELD. FUENTE: (ARDUINO ETHERNET SHIELD).....	67
FIGURA 3.0.4 IMPRESORA DE TICKET. FUENTE(HTTPS://WWW.ADAFRUIT.COM/)	68
FIGURA 3.0.5 DIAGRAMA DE CONEXIÓN ARDUINO-IMPRESORA DE TICKET. FUENTE(HTTPS://WWW.ADAFRUIT.COM),	69
FIGURA 3.0.6 METODOLOGÍA RUP. FUENTE: (HTTP://WWW.JLAYA.COM/METODOLOGIA-RUP-C).....	72
FIGURA 3.0.7 MODELO INICIAL DE CASOS DE USO. (DISEÑO PROPIO).....	76

ÍNDICE DE TABLAS

TABLA 1. POCO TIEMPO PARA EL DESARROLLO DEL PROYECTO.	77
TABLA 2. RECURSOS PARA EL DESARROLLO DEL PROYECTO. (DISEÑO PROPIO)	78
TABLA 3. DISTANCIA ENTRE DONDE SE ESTRUCTURÓ EL PROYECTO Y DONDE SE IMPLEMENTÓ. (DISEÑO PROPIO)	78
TABLA 4. RESULTADO DE LAS PRUEBAS. (ELABORACIÓN PROPIA)	81

MANUALES

1. MANUAL DEL USUARIO
2. MANUAL TECNICO

Capítulo 1

DESCRIPCIÓN DEL PROYECTO

PLANTEAMIENTO DEL PROBLEMA

El protocolo constituido en las Notarías de Colombia establece el monitoreo de indicadores que determinan la calidad del servicio de cada una de estas. Constantemente la Superintendencia de Notariado y Registro realiza la respectiva vigilancia y control de seguimiento para verificar dichos indicadores, dejando a cada Notaría cierta autonomía para la manera de prestar el servicio.

Es por esto que el número de trabajadores con que funciona la oficina, es determinado por el Notario de acuerdo a sus necesidades, al igual que la asignación de los salarios, funciones e ideas que ayuden a mejorar la calidad del servicio.

El personal reporta semanalmente la información y la documentación de los procesos que tuvieron en este tiempo. Mensualmente el notario y su equipo evalúan los resultados con el objetivo de mejorar la calidad del servicio y llevar la contabilidad de los ingresos obtenidos por la prestación de estos servicios, los cuales son fundamentales, debido a que las Notarías son autosostenibles.

Actualmente, uno de los principales problemas que enfrenta la Notaria Única de San Jacinto es el tiempo de atención al usuario, ya que no cuenta con una estructura de manejo de la información adecuada y actualizada, al estar la documentación en libros físicos, la manipulación constante la deterioran, lo que requiere de precaución y por ende demoras.

Si bien esta problemática está asociada a la mayoría de las Notarías del país, en San Jacinto tiene un componente adicional y es que afecta directamente a la comunidad campesina, que por el desconocimiento de procedimientos y por las demoras entre trayectorias, prefieren acudir a personas que ilegalmente se aprovechan del desconocimiento de estas personas y le quitan sus bienes.

PREGUNTAS PROBLEMA DE INVESTIGACIÓN

Con el desarrollo de este proyecto, se pretende dar respuesta total o parcial de las siguientes preguntas:

- ¿Puede automatizarse el proceso de diligenciamiento de documentos y el servicio de consultas (FAQ) de la Notaria y que beneficios se obtendrán al aplicar este sistema?
- ¿Qué ventajas y desventajas presenta el desarrollo de la plataforma mediante aplicaciones web con respecto a otro tipo de soluciones?

ESTADO DE DESARROLLO O ANTECEDENTES

Las aplicaciones web han sido la revolución de los últimos años. El enorme potencial de estas herramientas las convierte en una necesidad para cualquier empresa privada o pública.

Un sistema de datos centralizado en internet nos permite realizar gestiones administrativas, comunicación entre usuarios, venta de servicios, difusión de la información y un sinnúmero de procedimientos que nos facilitarán el trabajo en la empresa y mantendrán unificado nuestro sistema de información. (Indep Consultores, S.L., s.f.)

Las aplicaciones web no se limitan para su utilidad en internet. Estas aplicaciones son muy comunes para publicar en servidores de intranet que simplemente disponen de un servidor web instalado, ya sea que se necesite gestionar datos en una pequeña red para gestionar clientes, facturas, contabilidad, archivo documental o cualquier tipo de datos.

Los siguientes son temas de investigación referentes a aplicaciones web de apoyo para distintos servicios de utilidad de entidades:

- ❖ Un primer trabajo corresponde a David Fernández Moreno(2006) presentado en la Facultad de Informática de Universidad Politécnica de Madrid; realizó: ***“Diseño e implementación de una aplicación web para la gestión y control de visitas a centros educativos”***, en el cual se desarrolló una aplicación web utilizando diferentes tecnologías como *PHP(Hipertexto Preprocessor)*, *AJAX* y *(HTML)Hyper Text Markup Language*,

para gestionar las visitas realizadas por personal de la universidad a colegios e institutos educativos de la comunidad de Madrid – España.

Dentro del proceso de promoción institucional de la UPM(Universidad Politécnica de Madrid), se realizan visitas informativas a distintos centros educativos. El proyecto surge a partir de la necesidad de gestionar estas visitas realizadas por el personal de la UPM. Esta campaña inicio en el año 2003 dirigido a informar a los potenciales alumnos, acerca de la formación y futuro profesional que ofrece la Universidad. La recolección de datos se obtuvo principalmente de la observación, por medio de conductas, aspectos y eventos a observar de los usuarios.

Este trabajo conduce y orienta en el desarrollo de aplicaciones web para facilitar la prestación de servicios sociales de manera *on-line*.

- ❖ Un segundo trabajo corresponde a Nikolaos Polatidis (2012), presentado a la Universidad de Birmingham, realizó: “*Chatbot for admissions*”, este proyecto fue desarrollado con el objetivo de contribuir en la solución a la problemática de comunicación de estudiantes interesados en realizar estudios de postgrado en dicha Universidad, teniendo en cuenta la cantidad de solicitudes que se realizan, el tiempo y los costos que esto genera.

El reclutamiento de estudiantes de postgrado para la universidad de Birmingham es internacional por tanto la comunicación con los solicitantes internacionales trae consigo un fuerte compromiso con el acceso a la información necesaria para poder tomar decisiones de qué y donde estudiar.

Este proyecto se basó en las tecnologías de desarrollo de software, *Java, JavaScript, HTML, CSS, PostgreSQL*, entre otros. La preparación del proyecto se llevó a cabo mediante la observación, la revisión de literatura, investigación de herramientas existentes y entrevista al jefe de admisiones de la universidad.

- ❖ El tercer trabajo por Edgar Esteban Maidana y Roberto Demestri Rigoni(2014), presentado a la Universidad Nacional del Este(UNE) - Paraguay, realizaron: “*Diseño, desarrollo e implementación de pureAJAX, un framework AJAX compacto y eficiente*”, este trabajo se halla enmarcado dentro del área de tecnologías Web, específicamente en el área de aplicaciones asíncronas, pureAJAX es un framework desarrollado a fin de

responder a las necesidades comunes en el desarrollo de aplicaciones AJAX, compatible con todos los navegadores Web, siendo a la vez compacto y funcional.

La comunicación asíncrona permite al navegador Web realizar llamadas HTTP al servidor en segundo plano, sin interrumpir la interacción del usuario con el sitio Web, lo que lo hace interesante para nuestro proyecto.

El framework pureAJAX posee una serie de características convenientes como lo son el tamaño reducido (tan sólo 2,3420 KB), especialización de tareas, eficiencia en la ejecución de llamadas AJAX y facilidad de uso. Los logros obtenidos, pueden deducirse en 3 aspectos: la extensibilidad de la clase principal, tiempo de carga reducido, y la compatibilidad con los distintos navegadores, en las correspondientes plataformas.

- ❖ Un cuarto trabajo pertenece a Diana Cristina Vera Santana, Rubén Alejandro Lara Vásquez y Pedro Fabricio Echeverría Briones (2009), presentado a la Escuela Superior Politécnica del Litoral de Guayaquil – Ecuador. Realizaron la: ***“Implementación de un portal web para la automatización del proceso de consultorías de mentores GOLD de la Región Latinoamericana del IEEE (R9), utilizando arquitectura Java 2 Enterprise Edition - J2EE y tecnología Ajax”***. Dentro del IEEE existe un Grupo de Afinidad de Graduados de la última década llamado GOLD. El objetivo principal de este Grupo de Afinidad GOLD es ayudar a los jóvenes profesionales desde que son estudiantes hasta que ingresan al mundo laboral, para lo cual se ha planteado el programa *Peer tú Peer*, que servirá como canal de comunicación e interacción entre varios usuarios. El programa ha sido diseñado para cubrir las interrogantes que existen en los miembros IEEE de las Secciones, Ramas Estudiantiles y Capítulos Estudiantiles de Latinoamérica, a través de la mentaría de Profesionales IEEE. La recolección de datos se realizó mediante entrevistas y encuestas realizadas al comité GOLD de la Región Latinoamericana de IEEE y del comité de estudiantes de la R9 2008-2009.

En este proyecto se manejaron los conceptos relacionados a la arquitectura J2EE, la tecnología Ajax y los patrones de diseños usados en la implementación de un portal web para la automatización del proceso de consultorías de mentores GOLD de la Región Latinoamérica de la IEEE. Se realiza una descripción del análisis, diseño y plan de pruebas utilizado para los módulos de Administración, Consultorías y Evaluaciones.

JUSTIFICACIÓN

El desarrollo tecnológico en el municipio de San Jacinto repercute en gran medida en la educación de la población en nuevas tecnologías, utilizar una plataforma tecnológica permite fluidez de la información y presentación de la misma, mejora el servicio notarial y la percepción de calidad y seguridad en los trámites notariales del municipio.

Con el mejoramiento de la seguridad en el municipio y el programa de restitución de tierras por parte del gobierno en los Montes de María, se incrementó el número de solicitudes de trámites notariales, con lo cual las entidades públicas y privadas que son el soporte de estos programas les resulta indispensable contar con sistemas tecnológicos actualizados que mejoran la calidad de los mismos.

La importancia de este proyecto se ve en el progreso en la atención al usuario, donde disminuyen los costos y el tiempo en la consulta de trámites, adicionalmente contar con la plataforma de apoyo orientada a la web 2.0 permite a la notaria posicionarse dentro de las notarías de primer nivel del país.

Buena parte de los trámites con el Estado que los colombianos tienen que realizar a diario están migrando al mundo virtual. Como parte de este proceso, el Ministerio de las TIC anunció el lanzamiento del portal www.sivirtual.gov.co, una plataforma que reúne la oferta de trámites electrónicos disponibles en la actualidad. (REDACCIÓN TECNÓSFERA, EL TIEMPO , 2015)

Otro punto importante es el aporte de la Universidad del Magdalena en el apoyo y desarrollo de las empresas de la región caribe. Siendo la entidad de educación superior con un gran impacto en proyectos en el departamento de Bolívar.

IMPACTO ESPERADO DEL PROYECTO

El presente proyecto busca la organización y el fortalecimiento de la seguridad en el manejo de la información de la oficina Notarial del municipio en un plazo corto.

Por otro lado, se busca encaminar a la comunidad para el aprovechamiento del nuevo sistema que brinda una mejor gestión en cuanto a tiempos de respuesta para sus diligencias Notariales.

Y finalmente, en un largo plazo se espera que este proyecto sirva de ejemplo para el desarrollo de futuras herramientas tecnológicas que permitan mejorar la calidad de vida de esta población.

OBJETIVOS

GENERAL

Diseñar e implementar una plataforma autónoma que permita realizar consultas sobre procedimientos notariales y tramitar documentos de manera previa con el fin de mejorar la calidad de servicio de la Notaria Única de San Jacinto Bolívar.

ESPECÍFICOS

- ❖ Revisar conceptos de desarrollo de aplicaciones web, base de datos, el proceso notarial (consultas, tramitología y tipos de documentos).
- ❖ Construir una base de datos que suministre la información solicitada por los usuarios (FAQ y documentos).
- ❖ Diseñar una aplicación web para realizar las consultas notariales y la gestión de documentos.
- ❖ Realizar la intercomunicación base de datos y aplicación web.
- ❖ Implementar el modulo en físico en la oficina de la Notaria Única del municipio de San Jacinto – Bolívar.

DISEÑO METODOLÓGICO

Para el desarrollo de este proyecto se realizó la respectiva documentación acerca de las técnicas y metodologías de desarrollo web, además de sistema de gestión notarial.

TIPO Y DISEÑO DE INVESTIGACIÓN

El enfoque de este trabajo se caracteriza por ser de tipo documental, de campo y exploratoria, ya que se estudiaron proyectos similares previos y la realización de entrevistas no organizadas a las personas directamente relacionadas con el propósito de destacar los aspectos fundamentales de la problemática.

Para la construcción del “*Diseño e implementación de una plataforma autónoma para tramites y consultas de la Notaria única de San Jacinto Bolívar*” se tuvo en cuenta los métodos y tecnologías para el desarrollo del aplicativo web y la implementación de dispositivos hardware libre.

TÉCNICAS DE RECOLECCIÓN DE DATOS

Se reunió la información referente a técnicas y metodologías para el desarrollo de aplicaciones web y del sistema de manejo de información de las Notarías del país.

La recolección de datos se realizó a través de las encuestas, por lo que se hizo uso de herramientas tradicionales haciendo uso de elementos electrónicos.

TÉCNICAS DE INVESTIGACIÓN

Las técnicas empleadas en la Investigación han sido las siguientes:

- Recolección de la Información. Se realizaron los siguientes:
 - La Observación
 - Entrevistas
 - Encuestas

- Planteamiento de Soluciones:
 - La metodología empleada es: Metodología RUP.

- Análisis de Resultado

DESCRIPCIÓN DEL INSTRUMENTO UTILIZADO

Recolección de Información

Se hizo la recolección de la información a través de los siguientes instrumentos:

Observación: Este método consiste en el registro visual de lo que ocurre en una situación real, clasificando, consignando los acontecimientos pertinentes de acuerdo con algún esquema previsto y según el problema que se estudia. Es preciso estar alerta para discriminar adecuadamente todo este conjunto posible de informaciones.

Entrevista: Este método se utilizó para recabar información en forma verbal, a través de preguntas que propone el analista. Sirve para analizar la realidad y estado de la situación problemática, y solamente se trabajará con la jefa del área que involucrará el sistema.

Encuestas: Es el análisis que se aplicó a la muestra y así mismo se determinó el tipo de delito de mayor incidencia expresado en cantidades y porcentajes.

ALCANCES Y LIMITACIONES

El proyecto tiene como alcance el desarrollo de una plataforma que permita a los usuarios de la Notaria del municipio de San Jacinto del departamento de Bolívar, agilizar consultas y la tramitología de documentos digitales que se encuentran almacenados en un servidor.

El sistema contiene elementos de bajo costo que permiten versatilidad al momento de querer añadir otros dispositivos electrónicos, así como una aplicación web y una aplicación móvil en Ionic, que permite ser instalada en cualquier dispositivo.

El Proyecto es exclusivo de la Notaria Única del municipio de San Jacinto Bolívar con la intención de servir de apoyo al personal administrativo de la entidad.

LIMITACIONES

- Por políticas de seguridad de la entidad, el acceso a la información está bajo la red interna de la Notaria(Intranet), es decir que la aplicación no estará disponible para ser utilizada por medio de internet.
- La plataforma tiene acceso a documentos digitales, podrá visualizarlos, pero no imprimirlos en físico.
- Inicialmente la aplicación tendrá algunas preguntas base con sus respectivas respuestas, el encargado de administrar la aplicación actualizará el contenido mediante el panel de control, según sea necesario.

MARCO TEÓRICO

APLICACIONES INFORMÁTICAS

En informática, una aplicación es un tipo de programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de trabajos. Esto lo diferencia principalmente de otros tipos de programas, como los sistemas operativos (que hacen funcionar la computadora), las utilidades (que realizan tareas de mantenimiento o de uso general), y las herramientas de desarrollo de software (para crear programas informáticos).

ARQUITECTURA DE APLICACIONES

El término arquitectura de aplicación es usado en el diseño de aplicaciones, habitualmente del tipo cliente-servidor. En el diseño físico se especifica exactamente donde se encontrarán las piezas de la aplicación. En el diseño lógico o conceptual se especifica la estructura de la aplicación y sus componentes sin tomar en cuenta donde se localizará el software, hardware e infraestructura. Tales conceptos incluyen el orden de procesamiento, mantenimiento y seguimiento comunes en sistemas organizacionales.

Muchas veces se toma demasiado en cuenta el diseño físico de una aplicación. Por añadidura los desarrolladores generalmente asumen, indebidamente, que el diseño lógico se corresponde punto a punto con el diseño físico. Contrario a esto, un diseño adecuado debería permitir su implantación en varias plataformas y configuraciones. Como se puede ver, esta característica de portabilidad es un punto deseable para permitir que su aplicación sea flexible y escalable.

MODELO CLIENTE / SERVIDOR

El modelo arquitectónico cliente/servidor es un modelo de sistema en el que dicho sistema se organiza como un conjunto de servicios y servidores asociados, más unos clientes que acceden y usan los servicios (Ian Sommerville, 2005).

Los principales componentes de este modelo son:

- Servidores que ofrecen servicios a otros subsistemas. Un ejemplo son los servidores de impresoras que ofrecen servicios de impresión.
- Clientes que llaman a los servicios ofrecidos por los servidores. Puede haber varias instancias de un programa cliente ejecutándose concurrentemente.
- Una red que permite a los clientes acceder a estos servicios. Los clientes y los servidores pueden ejecutarse sobre una única máquina.

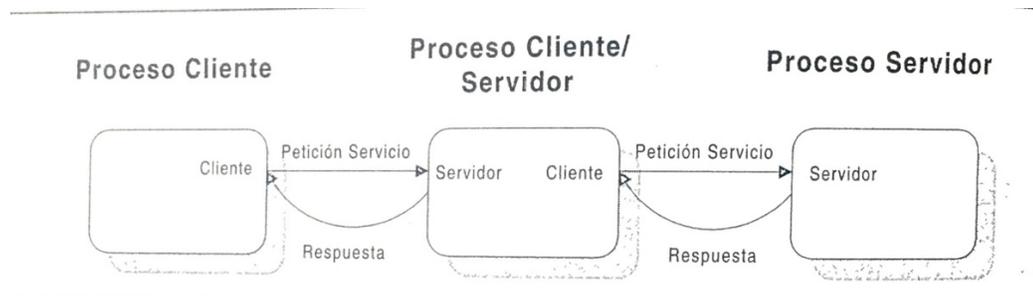


Figura 2.3.1 Modelo Cliente/Servidor Fuente: (Plattini, Calvo-Manzano, Cervera, & Fernández, 2004)

En la *Figura 2.1* el cliente realiza peticiones al servidor, mientras que el servidor se dedica simplemente a responderle. De por sí, el servidor tiene un papel pasivo por lo que necesita que un cliente le demande algo. Los principales servicios de Internet (WWW, FTP, SMTP, etc.) tienen clientes y servidores específicos, aunque en tiempos recientes se intenta integrar todo bajo una interfaz web que es más amigable para el usuario.

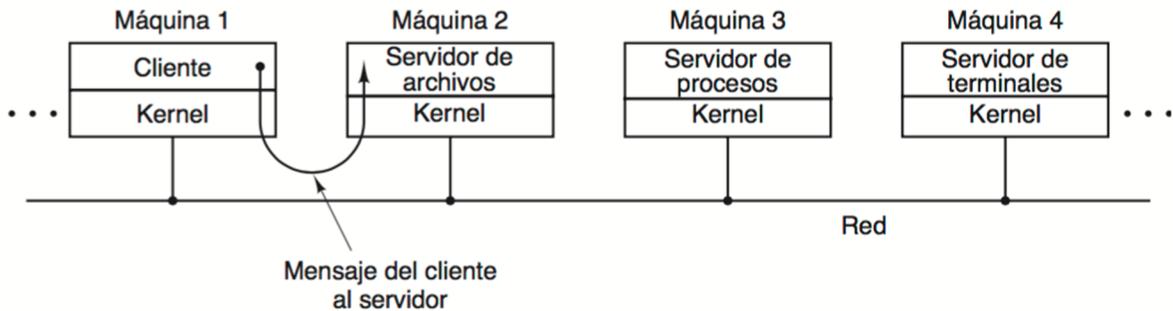


Figura 2.3.2. El modelo cliente-servidor sobre una red. Fuente: (Tanenbaum, 2009)

Los procesos pueden o no estar en una sola máquina física. Tales procesos en una aplicación cliente/servidor pueden localizarse en la misma máquina o separados físicamente. El diseño lógico, y no el físico, es el que determina en qué grado una aplicación es cliente/servidor (Ian Sommerville, 2005).

Separación de funciones

La arquitectura cliente/servidor nos permite la separación de funciones en tres niveles.

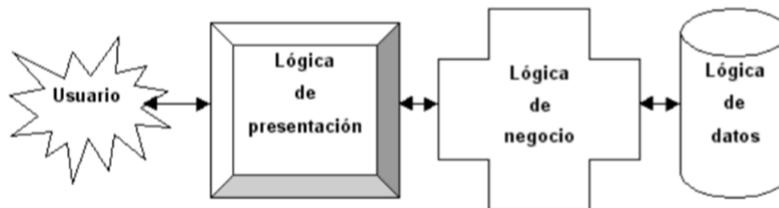


Figura 2.3.3 Separación de Funciones. Fuente: (Luján Mora, 2002).

- **Lógica de presentación.** Se encarga de la entrada y salida de la aplicación con el usuario. Sus principales tareas son: obtener información del usuario, enviar la información del usuario a la lógica de negocio para su procesamiento, recibir los resultados del procesamiento de la lógica de negocio y presentar estos resultados al usuario.
- **Lógica de negocio (o aplicación).** Se encarga de gestionar los datos a nivel de procesamiento. Actúa de puente entre el usuario y los datos. Sus principales tareas son: recibir la entrada del nivel de presentación, interactuar con la lógica de datos para ejecutar las reglas de negocio (business rules) que tiene que cumplir la

aplicación (facturación, cálculo de nóminas, control de inventario, etc.) y enviar el resultado del procesamiento al nivel de presentación.

- **Lógica de datos.** Se encarga de gestionar los datos a nivel de almacenamiento. Sus principales tareas son: almacenar los datos, recuperar los datos, mantener los datos y asegurar la integridad de los datos.

Si un sistema distribuido se diseña correctamente, los tres niveles anteriores pueden distribuirse y redistribuirse independientemente sin afectar al funcionamiento de la aplicación (Luján Mora, 2002).

Modelos de distribución en aplicaciones cliente/servidor.

Según como se distribuyan las tres funciones básicas de una aplicación (presentación, negocio y datos) entre el cliente y el servidor, podemos contemplar tres modelos: presentación distribuida, aplicación distribuida y datos distribuidos (Luján Mora, 2002).

Presentación distribuida.

El cliente sólo mantiene la presentación, el resto de la aplicación se ejecuta remotamente. La presentación distribuida, en su forma más simple, es una interfaz gráfica de usuario a la que se le pueden acoplar controles de validación de datos, para evitar la validación de los mismos en el servidor (Luján Mora, 2002).

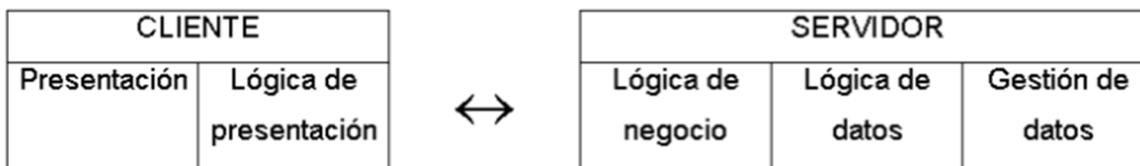


Figura 2.3.4 Presentación Distribuida. Fuente: (Luján Mora, 2002).

Aplicación distribuida.

Es el modelo que proporciona máxima flexibilidad, puesto que permite tanto a servidor como a cliente mantener la lógica de negocio realizando cada uno las funciones que le sean más

propias, bien por organización, o bien por mejora en el rendimiento del sistema (Luján Mora, 2002).

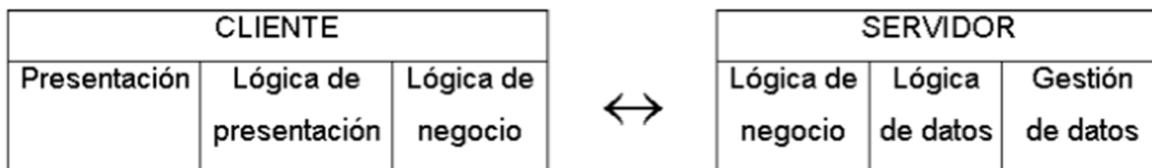


Figura 2.3.5 Aplicación Distribuida. Fuente: (Luján Mora, 2002).

Datos distribuidos.

Los datos son los que se distribuyen, por lo que la lógica de datos es lo que queda separada del resto de la aplicación. Se puede dar de dos formas: cheros distribuidos o bases de datos distribuidas (Luján Mora, 2002).

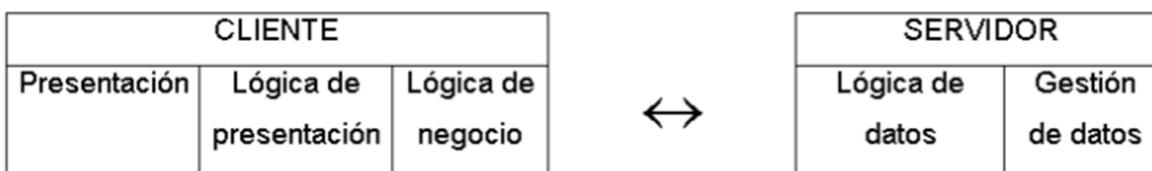


Figura 2.3.6 Datos Distribuidos. Fuente: (Luján Mora, 2002).

Arquitecturas de dos y tres niveles

La diferencia entre las arquitecturas de dos y tres niveles (o capas) estriba en la forma de distribución de la aplicación entre el cliente y el servidor.

Normalmente cuando se habla de aplicaciones de dos niveles se está haciendo referencia a una aplicación donde el cliente mantiene la lógica de presentación, de negocio, y de acceso a los datos, y el servidor únicamente gestiona los datos. Suelen ser aplicaciones cerradas que supeditan la lógica de los procesos cliente al gestor de bases de datos que se está usando.

En las arquitecturas de tres niveles, la lógica de presentación, la lógica de negocio y la lógica de datos están separadas, de tal forma que mientras la lógica de presentación se ejecutará normalmente en la estación cliente, la lógica de negocio y la de datos pueden estar repartidas entre distintos procesadores. En este tipo de aplicaciones suelen existir dos servidores: uno contiene la lógica de negocio y otro la lógica de datos.

El objetivo de aumentar el número de niveles en una aplicación distribuida es lograr una mayor independencia entre un nivel y otro, lo que facilita la portabilidad en entornos heterogéneos y la escalabilidad en caso de incorporación de nuevos clientes (Luján Mora, 2002).

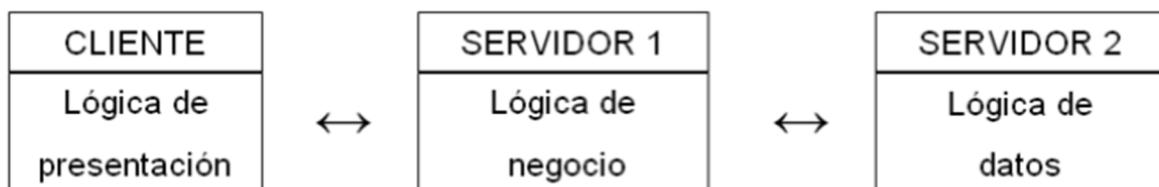


Figura 2.3.7 Arquitectura de tres niveles. Fuente: (Luján Mora, 2002)

APLICACIONES WEB

En las aplicaciones web suelen distinguirse tres niveles (como en las arquitecturas cliente/servidor de tres niveles): el nivel superior que interacciona con el usuario (el cliente web, normalmente un navegador), el nivel inferior que proporciona los datos (la base de datos) y el nivel intermedio que procesa los datos (el servidor web).

Una aplicación web (web-based application) es un tipo especial de aplicación cliente/servidor, donde tanto el cliente (el navegador, explorador o visualizador1) como el servidor (el servidor web) y el protocolo mediante el que se comunican (HTTP) están estandarizados y no han de ser creados por el programador de aplicaciones.

El protocolo HTTP forma parte de la familia de protocolos de comunicaciones TCP/IP, que son los empleados en Internet. Estos protocolos permiten la conexión de sistemas heterogéneos, lo que facilita el intercambio de información entre distintos ordenadores.

HTTP se sitúa en el nivel 7 (aplicación) del modelo OSI (Luján Mora, 2002).

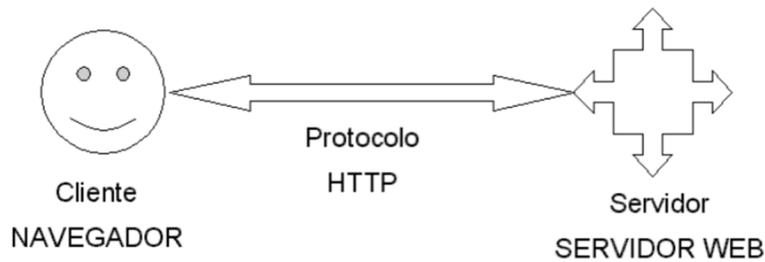


Figura 2.3.8 Esquema básico de una Aplicación Web. Fuente: (Luján Mora, 2002).

El cliente

El cliente web es un programa con el que interacciona el usuario para solicitar a un servidor web el envío de los recursos que desea obtener mediante HTTP2.

La parte cliente de las aplicaciones web suele estar formada por el código HTML que forma la página web más algo de código ejecutable realizado en lenguaje de script del navegador (JavaScript o VBScript) o mediante pequeños programas (applets) realizados en Java. También se suelen emplear plugin¹ que permiten visualizar otros (Luján Mora, 2002).

El servidor

El servidor web es un programa que está esperando permanentemente las solicitudes de conexión mediante el protocolo HTTP por parte de los clientes web. En los sistemas Unix suele ser un demonio y en los sistemas Microsoft Windows un servicio (Luján Mora, 2002).

La parte servidor de las aplicaciones web está formada por:

- Páginas estáticas (documentos HTML) que siempre muestran el mismo contenido.
- Recursos adicionales (multimedia, documentos adicionales, etc.) que se pueden emplear dentro de las páginas o estar disponibles para ser descargados y ejecutados (visualizados) en el cliente.

¹ Un plug-in es un módulo de software que se instala como un añadido a un programa o sistema y que proporciona nuevas características o servicios al programa o sistema

- Programas o scripts que son ejecutados por el servidor web cuando el navegador del cliente solicita algunas páginas. La salida de este script suele ser una página HTML estándar que se envía al navegador del cliente. Tradicionalmente este programa o script que es ejecutado por el servidor web se basa en la tecnología CGI. En algunos casos pueden acceder a bases de datos.

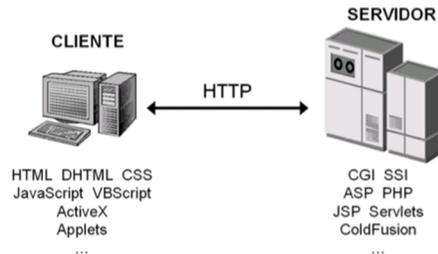


Figura 2.3.9 Tecnologías empleadas en el Cliente y Servidor Web. Fuente: (Luján Mora, 2002).

Entornos Web

Internet

Es un conjunto descentralizado de redes interconectadas que utilizan la familia de protocolos TCP/IP, garantizando las redes físicas heterogéneas que la componen funcionen como una red lógica única, de alcance mundial.

Intranet

Una Intranet es una red de ordenadores privados que utiliza tecnología internet para compartir dentro de una organización parte de sus sistemas de información y sistemas operacionales. El termino intranet se utiliza en oposición a internet, una red entre organizaciones, haciendo referencia por el contrario a una red comprendida en el ámbito de una organización.

Extranet

Una extranet es una red privada que utiliza protocolos de internet, protocolos de comunicación y probablemente infraestructura pública de comunicación para compartir de forma segura parte de la información u operación propia de una organización con proveedores, compradores, socios, clientes o cualquier otro negocio u organización. Se puede decir en otras palabras que una extranet es parte de la intranet de una organización que se extiende a usuarios fuera de ella.

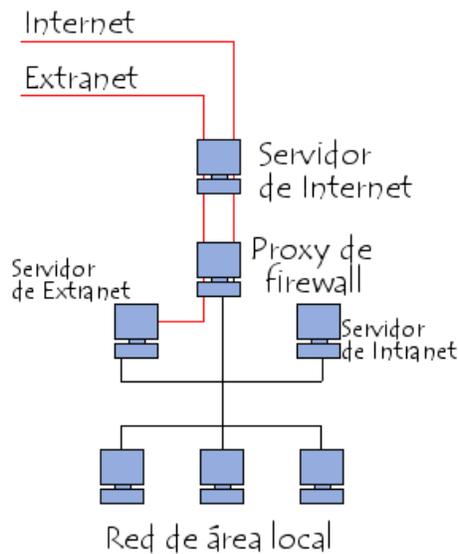


Figura 2.3.10 Entorno Web. Fuente: (Kioskea, 2014)

SISTEMA OPERATIVO MÓVIL

Un sistema operativo móvil o SO móvil es un sistema operativo que controla un dispositivo móvil al igual que las computadoras utilizan Windows o Linux entre otros. Sin embargo, los sistemas operativos móviles son mucho más simples y están más orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos.

Características básicas

Kernel. El núcleo o kernel proporciona el acceso a los distintos elementos del hardware del dispositivo. Ofrece distintos servicios a las superiores como son los controladores o drivers para el hardware, la gestión de procesos, el sistema de archivos y el acceso y gestión de la memoria.

Middleware. El middleware es el conjunto de módulos que hacen posible la propia existencia de aplicaciones para móviles. Es totalmente transparente para el usuario y ofrece

servicios claves como el motor de mensajería y comunicaciones, códecs multimedia, intérpretes de páginas web, gestión del dispositivo y seguridad.

Entorno de ejecución de aplicaciones. El entorno de ejecución de aplicaciones consiste en un gestor de aplicaciones y un conjunto de interfaces programables abiertas y programables por parte de los desarrolladores para facilitar la creación de software.

Interfaz de usuario. Las interfaces de usuario facilitan la interacción con el usuario y el diseño de la presentación visual de la aplicación. Los servicios que incluye son el de componentes gráficos (botones, pantallas, listas, etc.) y el del marco de interacción.

Principales Sistemas Operativos Móviles

ANDROID: actualmente Android pertenece a Google, pero es un sistema abierto cualquier fabricante puede desarrollar en él sus productos.

IOS: (Anteriormente denominado iPhone OS) es un sistema operativo móvil de Apple desarrollado originalmente para el iPhone siendo después usado en el iPod Touch y en el iPad.



Figura 2.3.11 Logos Apple y Android. Fuente: (<http://pacmac.es>)

LENGUAJE UNIFICADO DE MODELADO (UML)

El UML (Lenguaje Unificado de Modelado) es una de las herramientas más emocionantes en el mundo actual del desarrollo de sistemas. Esto se debe a que permite a los creadores de

sistemas generar diseños que capturen sus ideas en una forma convencional y fácil de comprender para comunicarlas a otras personas (Schmuller, 2001).

Diagramas del UML

El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos. En lugar de indicarle a usted cuales son los elementos y las reglas, veamos directamente los diagramas ya que los utilizará para hacer el análisis del sistema (Schmuller, 2001).

Diagrama de clases

Piense en las cosas que le rodean (una idea demasiado amplia, pero ¡inténtelo de cualquier forma!). Es probable que muchas de esas cosas tengan atributos (propiedades) y que realicen determinadas acciones. Podríamos imaginar cada una de esas acciones como un conjunto de tareas.

Una clase es una categoría o grupo de cosas que tienen atributos y acciones similares. He aquí un ejemplo: cualquier cosa dentro de la clase Lavadoras tiene atributos como son la marca, el modelo, el número de serie y la capacidad. Entre las acciones de las cosas de esta clase se encuentran: "agregar ropa", "agregar detergente", "activarse" y "sacar ropa".

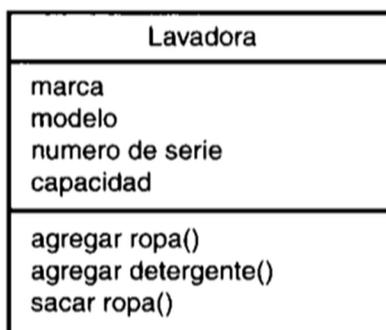


Figura 2.3.12 Símbolo UML de una Clase. Fuente: (Schmuller, 2001).

La figura anterior le muestra un ejemplo de la notación del UML que captura los atributos y acciones de una lavadora. Un rectángulo es el símbolo que representa a la clase, y se divide en tres áreas. El área superior contiene el nombre, el área central contiene los atributos, y el

área inferior las acciones. Un diagrama de clases está formado por varios rectángulos de este tipo conectados por líneas que muestran la manera en que las clases se relacionan entre sí (Schmuller, 2001).

Diagrama de objetos

Un objeto es una instancia de clase (una entidad que tiene valores específicos de los atributos y acciones). Su lavadora, por ejemplo, podría tener la marca Laundatorium, el modelo Washmeister, el número de serie GL57774 y una capacidad de 7 Kg.

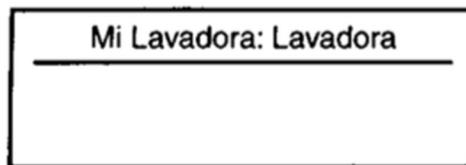


Figura 2.3.13 El Símbolo UML del objeto. Fuente: (Schmuller, 2001).

La figura anterior le muestra la forma en que el UML representa a un objeto. Vea que el símbolo es un rectángulo, como en una clase, pero el nombre está subrayado. El nombre de la instancia específica se encuentra a la izquierda de los dos puntos (:), y el nombre de la clase a la derecha (Schmuller, 2001).

Diagrama de casos de uso

Un caso de uso es una descripción de las acciones de un sistema desde el punto de vista del usuario. Para los desabolladores del sistema, ésta es una herramienta valiosa, ya que es una técnica de aciertos y errores para obtener los requerimientos del sistema desde el punto de vista del usuario. Esto es importante si la finalidad es crear un sistema que pueda ser utilizado por la gente en general (no sólo por expertos en computación).

Posteriormente trataremos este tema con mayor detalle; por ahora, le mostraré un ejemplo sencillo. Usted utiliza una lavadora, obviamente, para lavar su ropa. La figura siguiente le muestra cómo representaría esto en un diagrama de casos de uso UML (Schmuller, 2001).

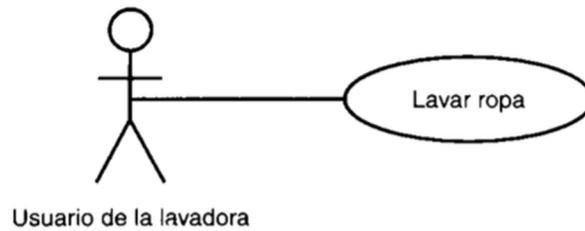


Figura 2.3.14 Diagrama de Casos de Uso UML. Fuente: (Schmuller, 2001).

A la figura correspondiente al Usuario de la lavadora se le conoce como actor. La elipse representa el caso de uso. Vea que el actor (la entidad que inicia el caso de uso) puede ser una persona u otro sistema.

BASES DE DATOS

Una base de datos es una colección de datos relacionados, es decir, un conjunto de hechos que pueden registrarse y que tienen un significado implícito. Por lo general, las bases de datos representan aspectos del mundo real y son diseñadas, construidas y pobladas con datos que tienen un propósito específico, se caracterizan por la coherencia de los datos que la integran (Elmasri & Navathe, 2002).

Bases de datos relacionales

Constituye el modelo más utilizado en la actualidad para el modelado de problemas reales y la administración de datos de manera dinámica. Almacena la información en varias tablas (filas y columnas de datos) o ficheros independientes y realiza búsquedas que permiten relacionar datos que han sido almacenados en más de una tabla. Se basa en el uso de relaciones, donde cada relación es una tabla compuesta por registros (las filas de una tabla) y campos (las columnas de una tabla). En el modelo relacional cada fila representa un hecho que normalmente se corresponde con un vínculo o una entidad del mundo real. El nombre de la tabla y de las columnas ayuda a interpretar el significado de los valores que están en cada fila. En el modelo relacional una fila se denomina “tupla”, una cabecera de columnas se denomina “atributo” y la tabla se denomina “relación”. En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia. Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos. La

información puede ser recuperada o almacenada mediante “consultas” que ofrecen una amplia flexibilidad y poder para administrar la información (Elmasri & Navathe, 2002).

Lenguaje estructurado de consulta (SQL)

Es un lenguaje de base de datos normalizado, utilizado por los diferentes manejadores de bases de datos para realizar determinadas operaciones sobre los datos o sobre la estructura de los mismos. Está diseñado como un lenguaje amplio que incluye instrucciones para definir, consultar y actualizar las bases de datos. Las funcionalidades que proporciona el SQL van más allá de la simple consulta o recuperación de datos. Esta a su vez permite definir los tipos de datos y manipular los datos. Además, el SQL permite la concesión y denegación de permisos, la implementación de restricciones de integridad, controles de transacción y la alteración de esquemas. El lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones agregadas. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos (Elmasri & Navathe, 2002).

El sistema de gestión de bases de datos MySQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. Desde enero de 2008 una subsidiaria de SUN MICROSYSTEMS desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado, se ofrece para cualquier uso compatible con la licencia GNU GPL, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C (Piñeiro Gómez, 2004).

MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones Web hay baja concurrencia en la modificación de datos, y en cambio el entorno es intensivo en lectura de datos lo que hace a MySQL ideal para este tipo de aplicaciones.

MySQL fue desarrollado bajo los lenguajes C y C++ y se destaca por su gran adaptación a diferentes entornos de desarrollo, permitiendo su interacción con los lenguajes de

programación más utilizados como: PHP, Perl y Java y su integración en distintos sistemas operativos.

MySQL es muy utilizado en aplicaciones Web, como Drupal o phpBB, en diversas plataformas como Unix y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP que a menudo aparece en combinación con MySQL lo cual no es una excepción en este proyecto.

Inicialmente MySQL carecía de elementos considerados esenciales en las bases de datos relacionales tales como: integridad referencial y transacciones. Estos elementos atrajeron a los desarrolladores de páginas Web con contenido dinámico, justamente por su simplicidad; aquellos elementos faltantes fueron llenados a medida que las aplicaciones lo utilizaban.

Poco a poco los elementos faltantes en MySQL están siendo incorporados tanto por desarrollos internos como por desarrolladores de software libre.

Entre las características disponibles en las últimas versiones de MySQL se pueden destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente
- Disponibilidad en gran cantidad de plataformas y sistemas
- Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles
- Transacciones y claves foráneas
- Conectividad segura
- Replicación, búsqueda e indexación de campos de texto

PHP

PHP tal y como se conoce hoy en día es en realidad el sucesor de un producto llamado PHP/FI. Creado en 1994 por Rasmus Lerdorf, la primera encarnación de PHP era un conjunto simple de ficheros binarios Common Gateway Interface (CGI) escritos en el lenguaje de programación C (“Historia de PHP”, 2016). Su nombre se debe a que originalmente Lerdorf lo utilizo para contar las veces que era visitado su curriculum.

PHP también conocido con el término Hypertext Preprocessor, ya que su motor, analiza las páginas buscando si hay alguna instrucción o comando que ejecutar antes de mostrarlas al cliente.

Los documentos HTML pueden incluir códigos PHP, incluso generarlos. Además, su sintaxis tiene características similares de Java, C y Perl, lo que permite asimilarlo fácilmente.

Su importancia se refleja en el desarrollo de páginas dinámicas, por su utilidad y relación con la gestión de bases de datos y manejo de formularios (cuentas de usuarios, contraseñas, etc.). Mientras que lenguajes como Javascript se ejecutan del lado del Cliente (Navegadores o Browser), PHP se encarga de enviar la información al Servidor al darle clic a cualquier botón de envío.

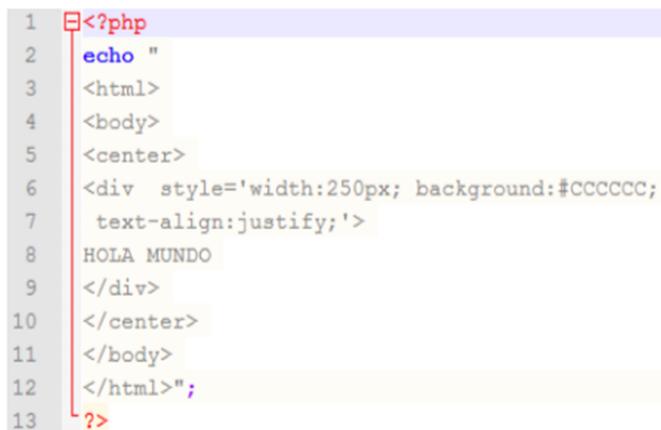
Otros aspectos que hace importante a PHP es la gratuidad y el soporte para múltiples sistemas operativos, bases de datos y servidores web.

Comandos Básicos

Como ejemplo se tomará la salida en pantalla del “hola mundo”. El resultado sería tan sencillo como:

```
<html>
<body>
<?php echo "HOLA MUNDO"; ?> </body>
</html>
```

Sin embargo, puede escribirse un poco más adornado:

A screenshot of a code editor showing PHP code. The code is as follows:

```
1 <?php
2 echo "
3 <html>
4 <body>
5 <center>
6 <div style='width:250px; background:#CCCCCC;
7 text-align:justify;'>
8 HOLA MUNDO
9 </div>
10 </center>
11 </body>
12 </html>";
13 ?>
```

The code is color-coded: PHP tags are red, HTML tags are blue, and the output text is black. The code is enclosed in a light blue box.

Figura 2.3.15 Archivo PHP. (Captura de pantalla)

Al correr el programa de la Figura 2.15, se obtiene el mismo resultado que con el código anterior. En el primer caso, el código php, se encuentra entre las líneas 6 y 8, y se encuentra, aparentemente, incrustado entre comandos de HTML. En la Figura 2.15, `<? php ?>` abarca todo el programa, y los comandos HTML, son producidos por él.

Se logra ver entonces que el lenguaje PHP que imprime en pantalla es:

```
<?php echo "..."; ?>
```

Variables PHP

Para la definición de variables, en PHP, los nombres de las variables van precedidos del signo de pesos: \$.

```
1 <html>
2 <body >
3 <?php
4 $nom='Francisco';
5 echo $nom;
6 echo "<br>$nom";
7 echo '<br>'. $nom. ' ';
8 ?>
9 </body>
10 </html>
```

Figura 2.3.16 Ejemplo PHP. (Captura de pantalla)

Al correr el programa saldrá el nombre Francisco tres veces seguidas, en la línea 4, se encuentra en comillas asociado a la variable `$nom`.

Estructuras de control

Las estructuras de control permiten modificar la ejecución de un programa. Generalmente se agrupan en condicionales, como el `if ()` y bucles, como el `while`. El `if` es una de las de mayor uso, su sintaxis es:

```
if(condición) { ... sentencias ...
}
```

Al evaluarse la condición y en caso de ser verdadera se ejecutarán las sentencias.

De la misma naturaleza, se encuentra el if...else

```
if(condición) {...sentencias...} else {...sentencias...}
```

Ahora bien, dentro de los bucles, se enlistarán dos estructuras: while y for.

```
while(condición) { ... sentencias ...  
}
```

Esta instrucción nos permite realizar un conjunto de instrucciones, un número determinado de veces. Por su parte, el ciclo for es muy semejante a while, pero ligeramente más complejo.

```
for (exp_ini; exp_condición; bucle) { ... sentencias ...  
}
```

En la expresión de inicialización (exp_ini), se declara e inicia la variable o variables. La condición, por su parte, debe contener los requerimientos para que las sentencias en { } se ejecuten. Finalmente, el bucle modifica el valor de las variables utilizadas como contadores del mismo (Velázquez Medina & Hueyo Chino, 2015).

Datos de entrada, cadenas y arrays en PHP

Se reconocen en los formularios de HTML, la forma estándar de entrada de datos que tomará PHP. Veamos como procede la toma y procesamiento de datos.

La sintaxis de un formulario HTML se aprecia en la siguiente línea:

```
<form action='archivo.php' method='post' >  
<input type='text' name='a' size='10' >  
<input type='submit' name='enviar' value='CALCULAR'> </form>
```

Dichas instrucciones generan las clásicas cajas de texto que abundan en internet. Expliquemos brevemente sus partes.

El atributo *action* indica la dirección del archivo o url donde se procesará la información. Por su parte, *method* indica la forma de transferencia de datos, en este caso, post. El segundo

renglón, genera propiamente dicho, la caja de texto, donde entren 10 caracteres y cuyo dato de entrada, al ser enviado será identificado por PHP como:

\$ POST['a']

El penúltimo renglón, produce un botón de envío con la palabra: CALCULAR. El *name* de esta línea, es el nombre con el cual sabremos si el botón fue oprimido.

Se debe decir que el formulario, puede ser colocado en una página HTML, pero la página de procesamiento, indicada en *action*, en nuestro caso, debe ser PHP (Velázquez Medina & Hueyo Chino, 2015).

En código de la figura 2.17 se agregó un **if** y un formulario, tomando de base el código de la figura 2.15, con el fin de que este pida nombre por teclado y al enviar aparezca un letrero de bienvenida con el nombre introducido.

El formulario corresponde a las líneas 6 a 9. Se ha colocado como el archivo que procesará la información, el mismo de salida. El nombre introducido en la caja de texto (línea 7), se identificará al momento del envío como **\$ POST['tunombre']**.

```

1 <html>
2 <body >
3 <center>
4 <div style='width:250px; background:#CCCCCC;
5 text-align:justify;'>
6 <form action='file1.php' method='post' >
7 <input type='text' name='tunombre' size='10' >
8 <input type='submit' name='enviar' value='ENVIAR'>
9 </form>
10 <?php
11 if(isset($_POST['enviar'])) {
12     $tunombre=$_POST['tunombre'];
13     echo "HOLA $tunombre ";
14 }
15 ?>
16 </div>
17 </center>
18 </body>
19 </html>

```

Figura 2.3.17 Ejemplo Formulario PHP. Fuente: (Velázquez Medina & Hueyo Chino, 2015).



Figura 2.3.18 Apariencia del formulario. Fuente: (Velázquez Medina & Hueyo Chino, 2015).

Al momento de cargar la página, el comando **if**, verifica, antes que nada, si el botón ENVIAR ha sido pulsado. Ésta tarea la realiza con la función

isset (\$ POST['enviar'])

¿Qué ocurre al momento de colocar tu nombre y pulsar ENVIAR? La función **isset** indica tal hecho y entonces, se ejecutan las sentencias escritas dentro del **if**. Por simplicidad, asignamos (vea línea 12) un nombre más sencillo a la variable enviada

\$tunombre = \$ POST['tunombre']

En realidad, para no excederse en nombres de variables, recuperamos el nombre con el cual fue identificada en el formulario. Posteriormente, viene la impresión en pantalla.



Figura 2.3.19 Impresión en pantalla. Fuente: (Velázquez Medina & Hueyo Chino, 2015).

String y arrays

Por un lado, un string representa una cadena de caracteres. Como el ejemplo anterior:

`$nom='Francisco';`

De hecho, en los formularios, los datos introducidos son tomados como cadenas. Es posible identificar a cada carácter de un string. Si consideramos a la variable **`$nom='Francisco'`**, resulta sencillo, entender que

`$nom[0]=F,$nom[1]=r, $nom[2]=a ...`

El desarrollo del trabajo nos presentará otros comandos referentes a string. Por su parte, un array, hablando en términos computacionales, es un vector o matriz cuya estructura permite el almacenamiento de un conjunto de datos, bajo un mismo nombre.

La estructura de un array tiene, evidentemente, una infinidad de aplicaciones. Agregue el hecho de que los datos que formen parte de array pueden ser de diferente naturaleza: números, string, array, etc. (Velázquez Medina & Hueyo Chino, 2015).

Laravel

Laravel es el nombre de un framework creado para trabajar con PHP creado en el año 2011 por Taylor Otwell y que con el paso del tiempo, ha ido ganando terreno a otros framework para trabajar con PHP como Symfony o Zend Framework.

Se trata de framework de desarrollo con una curva de aprendizaje muy rápida y que maneja una sintaxis expresiva, elegante, con el objetivo de eliminar la molestia del desarrollo web facilitando las tareas comunes, como la autenticación, enrutamiento, sesiones y caché. Proporciona potentes herramientas necesarias para construir aplicaciones robustas y que puede ser utilizado tanto para proyectos a nivel empresarial como para proyectos más sencillos, lo que significa que es perfecto para todos los tipos de proyectos (HostaliaWhitepapers, 2016).

Entre las ventajas de utilizar el framework Laravel encontramos:

- Reducción de costos y tiempos en el desarrollo y posterior mantenimiento de la aplicación creada.
- Curva de aprendizaje relativamente baja si se compara con otros frameworks de PHP.
- Flexible y adaptable no sólo al uso del sistema MVC tradicional, sino que para reducir las líneas de código propone lo que denomina "**Routes with clousures**".
- Modular y con un amplio sistema de paquetes y drivers con el que se puede extender las funcionalidades de forma sencilla, robusta y segura.
- Sencillez a la hora de utilizar los datos mediante Eloquent, que se trata de un ORM cuya interacción con las bases de datos es totalmente orientada a objetos, siendo compatible con la gran mayoría de bases de datos del mercado
- Facilita el manejo de las rutas de nuestra aplicación, así como la generación de URLs amigables que ayudan a mejorar el posicionamiento web
- Uso del sistema de plantillas Blade, que se caracterizan por ser más simples y que además incluyen un sistema de caché que las hace más rápidas
- Una gran comunidad y mucha documentación, sobre todo en su sitio oficial
- Cuenta con una herramienta de líneas de comando llamada Artisan que permite programar tareas programadas como por ejemplo para ejecutar migraciones, pruebas a determinadas horas.

Estructura de un proyecto Laravel

De todos los archivos y carpetas que podemos ver en la figura, los más importantes son "**app**", "**public**" y "**vendor**".

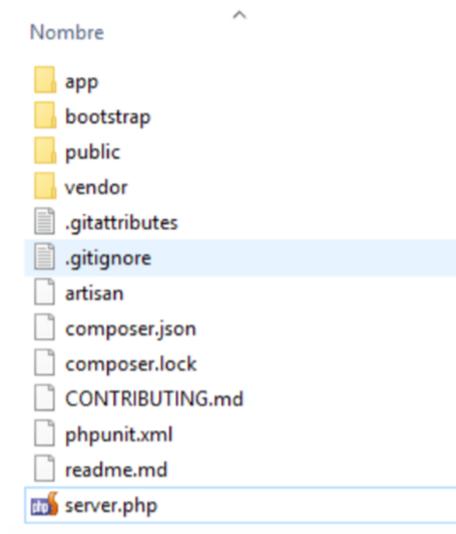


Figura 2.3.20 Directorio de Proyecto en Laravel. Fuente: (HostaliaWhitepapers, 2016).

Directorio app

Se trata de la carpeta más importante y donde se suele trabajar la mayor parte del tiempo. En su interior nos encontraremos nuestros modelos de base de datos, los controladores, las vistas o los archivos de configuración de la aplicación, todo ello ordenado en una serie de carpetas y archivos que explicamos a continuación:

- **app/router.php:** Será el archivo donde se definirán las URLs de nuestra aplicación.
- **app/filter.php:** En este archivo se indicarán los distintos filtros que se aplicarán a las rutas que definamos. Por ejemplo, si se quiere acceder a una zona privada y no se está logueado, que le redirija al formulario de login.
- **app/config:** En el interior de esta carpeta, nos encontraremos los archivos de configuración de nuestra aplicación.
- **app/controller:** Dentro de este directorio, se almacenarán los diferentes controladores que se utilizarán para el funcionamiento de nuestra página web.
- **app/database:** Se guardarán las migraciones que son versiones de la base de datos, escritas con Schema Builder propio de Laravel.
- **app/models:** Esta otra carpeta contendrá los modelos que interaccionan con nuestra base de datos y que pueden ser modificados en cualquier momento.

- **app/views:** Hace referencia a la vista de la aplicación, es decir, todo el código HTML que se utilizaría en la página web que fuéramos a construir
WHITEPAPERS: Laravel, un framework de PHP.

Directorio public

En el interior de este directorio, nos encontramos el archivo "**index.php**" que es el encargado de lanzar la aplicación. Además de esto, también será donde almacenaremos todos nuestros archivos CSS, JS e imágenes que utilizaremos para desarrollar la aplicación.

Directorio Vendor

En la carpeta vendor, lo que nos encontramos es todo el código correspondiente al framework Laravel, así como código de terceras personas o empresas que lo comparten con el resto de la comunidad para que lo utilicen en cualquier proyecto.

JAVASCRIPT

Javascript es lo que se conoce como **lenguaje script**, es decir: se trata de código de programación que se inserta dentro de un documento. Javascript fue desarrollado por la empresa **Netscape** con la idea de potenciar la creación de páginas Web dinámicas para su navegador **Navigator**.

Javascript (en contra de lo que se podría suponer) es totalmente distinto de Java. Java crea programas totalmente independientes y operativos; Javascript es más sencillo porque lo único que permite es insertar código especial dentro del HTML de una página, su función es ampliar las posibilidades de HTML. Javascript no crea programas independientes, dependen por completo del código HTML de la página.

La ventaja fundamental de Javascript es que su aprendizaje y uso son muy sencillos y que permite realizar labores complejas en una página sin necesidad de aprender CGI. (Sánchez, 2003)

Versiones de JavaScript

Puesto que JavaScript fue desarrollado por Netscape, los navegadores de esta empresa lo incluyen desde la versión 2. Microsoft por su parte incluyó en la versión 3 una variante de este código llamado **JScript** que es casi idéntico al original JavaScript. Después se estandarizó el lenguaje, aunque ambas compañías poseen elementos que no son comunes con el estándar (aunque prácticamente todo el estándar es reconocido por ambas).

Así aparición el JavaScript 1.1 que es admitido por Navigator 3 y por Explorer 4. Y las versiones 1.2 y 1.3 que son reconocidas por las versiones 4 y posteriores de ambos navegadores. (Sánchez, 2003)

Inclusión de Javascript en las páginas

Para hacer que un documento HTML incluya instrucciones en Javascript se debe hacer uso de la etiqueta **<SCRIPT>** de esta forma:

```
<script language="JavaScript">  
    código JavaScript  
</script>
```

Si se quiere especificar qué versión de Javascript se utiliza, para evitar que navegadores que no soportan la versión decodifiquen el Javascript, entonces se usa, por ejemplo:

```
<script language="JavaScript1.3">
```

Los navegadores que no soportan Javascript, no interpretarían las instrucciones Javascript sino que mostrarían el texto de las instrucciones en la página. Para evitar que estos navegadores lean el código en Javascript se hace:

```
<script language="Javascript">  
<!--  
    código JavaScript  
//-->  
</script>
```

El signo “<!--” indica principio de comentario en HTML y el signo “-->” indica fin de comentario. A su vez el signo “//” indica comentario en Javascript (el intérprete de Javascript no tendrán en cuenta esa línea). (Sánchez, 2003)

AngularJS

AngularJS es, en síntesis, un framework de código abierto y gratuito desarrollado por Google. Está basado en el popular lenguaje Javascript y su objetivo principal es crear aplicaciones web dinámicas y eficientes.

A diferencia de otros frameworks populares, AngularJS es un framework estructural, no depende ni este compuesto por elementos gráficos, imágenes o CSS, solamente se enfoca en administrar la parte lógica de tu aplicación.

Una aplicación AngularJS en su mínima expresión es, en síntesis, un documento HTML enlazado a un archivo JavaScript en el cual se encuentran un grupo de comandos e instrucciones (Solis, 2015).

Entorno de desarrollo

Es esencial escoger un editor o IDE, entre los mejores encontramos los editores **Visual Studio Coda** o **Sublime Text 2/3** y al IDE **JetBrains WebStorm**.

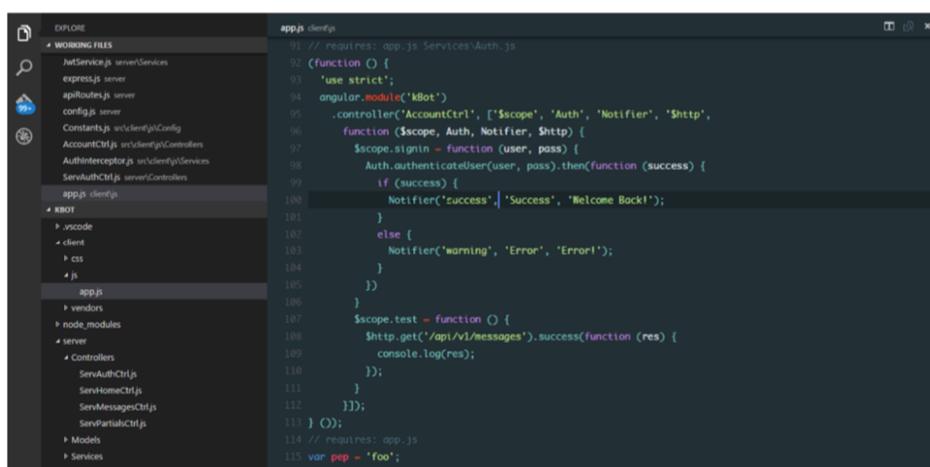


Figura 2.3.21 Visual Studio Code. Fuente (Rivero Dorta, 2016)

```

(function () {
  'use strict';
  angular.module('kBot')
    .config(['$stateProvider', '$stateProvider', '$urlRouterProvider',
    function ($stateProvider, $stateProvider, $urlRouterProvider) {
      $urlRouterProvider.otherwise("/");
      $locationProvider.html5Mode(true);
      $stateProvider
        .state('home', {
          url: '/',
          templateUrl: 'partials/home'
        })
        .state('state2', {
          url: "/state2",
          template: "partials/state2.html"
        });
    }
  ));
}
// requires: app.js ConfigConstants.js
(function () {
  'use strict';
}

```

Figura 2.3.22 Sublime Text. Fuente: (Rivero Dorta, 2016)

```

Auth.$inject = ['$window', '$http', '$q', 'AuthToken', 'ENDPOINT'];
function Auth($window, $http, $q, AuthToken, ENDPOINT) {
  var service = {
    authenticateUser: authenticateUser,
    currentUser: currentUser,
    isAuthenticated: isAuthenticated
  };
  return service;
}
//
var currentUser = undefined;
function authenticateUser(user, pass) {
  var def = $q.defer();
  $http.post(ENDPOINT + 'login', { username: user, password: pass })
    .then(handleResponse);
  function handleResponse(res) {
    if (res.data.success) {
      currentUser = res.data.user;
      AuthToken.setToken(res.data.token);
      def.resolve(true);
    } else {
      def.resolve(false);
    }
  }
}
return def.promise;

```

Figura 2.3.23 WebStorm. Fuente:(Rivero Dorta, 2016)

El siguiente paso es tener listo un servidor donde se pueda desarrollar la aplicación.

```

1 var express = require('express'),
2   app      = express();
3
4 app.use(express.static(__dirname+'/public'))
5   .get('*', function(req, res){
6     res.sendFile('/public/index.html', {root:__dirname});
7   }).listen(3000);

```

Este código crea un archivo “*server.js*” usando ExpressJs con NodeJs. Al ejecutar el comando **node server.js** se accede a la aplicación en la maquina local por el puerto 3000 (localhost:3000). Todas las peticiones a la aplicación serán redirigidas a index.html que se encuentra en la carpeta public. De esta forma podremos usar el sistema de rutas de **AngularJS** con facilidad.

Otra opción es usar el servidor Apache ya sea instalado en local en el pc como servidor http o por las herramientas AMP. En la configuración de los sitios disponibles de apache crearemos un *virtualhost* como el ejemplo siguiente.

```
1 <VirtualHost *:80>
2     # DNS que servirá a este proyecto
3     ServerName miapp.dev
4     # La direccion de donde se encuentra la aplicacion
5     DocumentRoot /var/www/miapp
6     # Reglas para la reescritura de las direcciones
7     <Directory /var/www/miapp>
8         RewriteEngine on
9         # No reescribir archivos o directorios.
10        RewriteCond %{REQUEST_FILENAME} -f [OR]
11        RewriteCond %{REQUEST_FILENAME} -d
12        RewriteRule ^ - [L]
13
14        # Reescribir todo lo demás a index.html para usar el modo de rutas HTML5
15        RewriteRule ^ index.html [L]
16    </Directory>
17 </VirtualHost>
```

Después de haber configurado el host virtual para la aplicación necesitamos crear el DNS local para que responda a nuestra aplicación. en Windows está en la carpeta dentro de la carpeta del sistema **C:Windows\system32Drivers\etc\hosts**.

La herramienta *bower* funciona como un gestor de dependencias y a la vez permite compartir las creaciones, además se encarga de obtener las dependencias de la aplicación y las mantiene actualizadas. Lo primero es crear un archivo **bower.json** donde se define el nombre de la aplicación y las dependencias. El archivo tiene la siguiente estructura.

```
1  {  
2    "name": "miApp",  
3    "dependencies": {  
4      "angular": "~1.2.*"  
5    }  
6  }
```

En la línea 2 se define el nombre de la aplicación “miApp”. Una vez más en la consola se ejecuta `bower install` en la carpeta que tiene el archivo **bower.json**. Este creará una carpeta **bower_components** donde incluirá el framework para usar en la aplicación.

Una de las principales ventajas que nos proporciona **Bower** es que podremos distribuir la aplicación sin ninguna de sus dependencias. Podremos excluir la carpeta de las dependencias sin problemas ya que en cada lugar donde se necesiten las dependencias podremos ejecutar `bower install` y `bower` las gestionará por nosotros. Esto es muy útil a la hora de trabajar en grupo con sistemas de control de versiones como **GitHub** ya que en el repositorio solo estaría el archivo **bower.json** y las dependencias en las máquinas locales de los desarrolladores (Rivero Dorta, 2016).

AngularJS y sus características

Con este framework tendremos la posibilidad de escribir una aplicación de manera fácil, que con solo leerla podríamos entender qué es lo que se quiere lograr sin esforzarnos demasiado. Además de ser un framework que sigue el patrón MVC² nos brinda otras posibilidades como la vinculación de datos en dos vías y la inyección de dependencia. Sobre estos términos estaremos tratando más adelante.

Plantillas

AngularJS nos permite crear aplicaciones de una sola página, o sea podemos cargar diferentes partes de la aplicación sin tener que recargar todo el contenido en el navegador. Este

² (Model View Controller) Estructura de Modelo, Vista y Controlador introducido en los 70 y obtuvo su popularidad en el desarrollo de aplicaciones de escritorio.

comportamiento es acompañado por un motor de plantillas que genera contenido dinámico con un sistema de expresiones evaluadas en tiempo real.

Estructura MVC

La idea de la estructura MVC no es otra que presentar una organización en el código, donde el manejo de los datos (Modelo) estará separado de la lógica (Controlador) de la aplicación, y a su vez la información presentada al usuario (Vistas) se encontrará totalmente independiente.

Vinculación de datos

Desde que el DOM pudo ser modificado después de haberse cargado por completo, librerías como jQuery hicieron que la web fuera más amigable. Permitiendo de esta manera que en respuesta a las acciones del usuario el contenido de la página puede ser modificado sin necesidad de recargar el navegador. Esta posibilidad de modificar el DOM en cualquier momento es una de las grandes ventajas que utiliza AngularJS para vincular datos con la vista.

Directivas

Si vienes del dominio de jQuery esta será la parte donde te darás cuenta que el desarrollo avanza de forma muy rápida y que seleccionar elementos para modificarlos posteriormente, como ha venido siendo su filosofía, se va quedando un poco atrás comparándolo con el alcance de AngularJS. jQuery en si es una librería que a lo largo de los años ha logrado que la web en general se vea muy bien con respecto a tiempos pasados

Inyección de dependencia

AngularJS está basado en un sistema de inyección de dependencias donde nuestros controladores piden los objetos que necesitan para trabajar a través del constructor.

Luego AngularJS los inyecta de forma tal que el controlador puede usarlo como sea necesario. De esta forma el controlador no necesita saber cómo funciona la dependencia ni cuáles son las acciones que realiza para entregar los resultados.

Estructura de Ficheros

```
App
├─ css
├─ img
├─ index.html
├─ js
│  └─ app.js
│  └─ Registro
│     └─ Registro.js
│        └─ Config
│           └─ Controllers
│              └─ Directives
│                 └─ Filters
│                    └─ Models
│                       └─ Services
│  └─ Blog
│     └─ Blog.js
│        └─ Config
│           └─ Controllers
│              └─ Directives
│                 └─ Filters
│                    └─ Models
│                       └─ Services
│  └─ Tienda
│     └─ Tienda.js
│        └─ Config
│           └─ Controllers
│              └─ Directives
│                 └─ Filters
│                    └─ Models
│                       └─ Services
│  └─ Ayuda
│     └─ Ayuda.js
│        └─ Config
│           └─ Controllers
│              └─ Directives
│                 └─ Filters
│                    └─ Models
│                       └─ Services
├─ partials
│  └─ Registro
│  └─ Blog
│  └─ Tienda
```

Como podemos observar al establecer esta estructura en nuestro proyecto, no importa cuánto este crezca, siempre se mantendrá organizado y fácil de mantener.

En el directorio `app/js` es donde se guarda todo el código de nuestra aplicación, con excepción de la página principal de entrada a la aplicación y las plantillas (`partials`), que estarán a un nivel superior junto a los archivos de estilos y las imágenes. En el archivo `app.js` es donde declararemos la aplicación (módulo) y definiremos sus dependencias.

Módulos

Los módulos son una forma de definir un espacio para nuestra aplicación o parte de la aplicación ya que una aplicación puede constar de varios módulos que se comunican entre sí (Rivero Dorta, 2016).

```
1 <body ng-app="miApp">
2   <div class="container" ng-controller="miCtrl">
3     {{ mensaje }}
4   </div>
5   <script src="lib/angular.js"></script>
6   <script>
7     angular.module('miApp', [])
8       .controller('miCtrl', function ($scope) {
9         $scope.mensaje = 'AngularJS Paso a Paso';
10      });
11   </script>
12 </body>
```

La directiva **ng-app** es la encargada de auto arrancar una aplicación Angular, indica el elemento raíz y se debe colocar como atributo en la etiqueta que quieres que sea la raíz de la aplicación.

Incluir el archivo **angular.js** en la aplicación permite la disponibilidad del objeto **angular** en el entorno global o sea como propiedad del objeto **window**.

En la línea 7 se define un nuevo módulo para la aplicación, haciendo uso del método **module** del objeto **angular**. Este método tiene dos funcionalidades: crear nuevos módulos o devolver un módulo existente. Para crear un nuevo módulo es necesario pasar dos parámetros al método. El primer parámetro es el nombre del módulo que queremos crear y el segundo una lista de módulos necesarios para el funcionamiento del módulo que estamos creando. La segunda funcionalidad es obtener un módulo existente, en este caso sólo pasaremos un primer parámetro al método, que será el nombre del módulo que queremos obtener y este será devuelto por el método (Rivero Dorta, 2016).

jQuery & jQuery Mobile

jQuery

jQuery es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar páginas web. Fue presentada el 14 de enero de 2006 en el BarCamp NYC.

jQuery es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privados. jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

Sus características más importantes son:

- a) Selección de elementos DOM. Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3 y un plugin básico de XPath.
- b) Eventos, manipulación de la hoja de estilos CSS.
- c) Efectos y animaciones. Animaciones personalizadas.
- d) AJAX. Soporta extensiones.
- e) Utilidades varias como obtener información del navegador, operar con objetos y
- f) vectores, funciones para rutinas comunes, etc.
- g) Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari
- h) 3+, Opera 10.6+ y Google Chrome 8+.



Figura 2.3.24 Logo de jQuery.

jQuery Mobile

jQuery Mobile es una librería de Javascript optimizada para web en móviles que actualmente está siendo desarrollada por el equipo del proyecto jQuery. El desarrollo se centra en crear un framework compatible con el mayor número de smartphones o tablets debido a la gran heterogeneidad que hay en el mercado de estos dispositivos móviles. Es compatible con otros frameworks de aplicaciones para móvil como PhoneGap, Worklight y otros.



Figura 2.3.25 Logo de jQuery Mobile.

Sus características más importantes son:

- Compatible con la mayoría de las plataformas móviles como la mayoría de los navegadores: iOS, Android, BlackBerry, Web OS, Symbian, Windows Phone 7, y más.
- Construido sobre el núcleo jQuery, para que la curva de aprendizaje sea mínima para desarrolladores familiarizados con la sintaxis de jQuery.
- Un framework de temas que permite personalizar uno rápidamente.
- Dependencias limitadas y ligero que optimizan su velocidad..
- El mismo código de base escalará el tamaño automáticamente a cualquier
- pantalla.
- Posibilidad de navegación entre páginas por AJAX con animaciones en las
- transiciones de página
- UI widgets optimizados para táctil y para quien desconozca la plataforma.

HOJAS DE ESTILO EN CASCADA (CSS - CASCADING STYLE SHEETS)

Desde el inicio del desarrollo de páginas web, el lenguaje HTML se ha limitado al momento de darle forma y diseñar un documento, los primeros softwares o programas para desarrollar sitios web, eran editores de texto. Esto debido a que la web fue concebida para otros usos, científicos, sobre todo, más enfocados en un proceso distinto al actual.

Para ofrecer una solución a estas variables, los diseñadores han empleado técnicas tales como la utilización de tablas, con imágenes para ajustarlas a determinado diseño y medidas, la incorporación de lenguajes de programación entre las líneas de codificación del HTML, entre otras. Esta forma de trabajo ha creado los distintos problemas en las páginas en cuanto a su visualización e interpretación sobre distintas plataformas se refiere.

Esta mezcla de lenguajes, contenido, y el uso inapropiado de elementos propios del HTML para la diagramación o diseño, provocan inconvenientes en la interpretación del código HTML por parte los navegadores, haciéndolas indescifrables y difíciles de buscar y depurar; degradando su utilidad.

En función de los problemas presentados anteriormente, se busca separar el contenido del diseño y la programación. Es aquí donde se emplea la idea de desarrollar las páginas web por capas, una capa principal atribuida al contenido, una secundaria, la diagramación y diseño; y una tercera a la funcionalidad, esta última relacionada con los lenguajes de programación.

Refiriéndonos al diseño de las páginas en particular, aparecen las hojas de estilo en cascada tomando una doble intención en el desarrollo, generando una revolución en el proceso, permitiendo mejorar la calidad gráfica de las páginas web y separando el diseño del contenido.

Las hojas de estilo en cascada (*Cascading Style Sheets*, CSS), ofrecen una solución para proponer formatos a las páginas web; un lenguaje formal empleado para definir la presentación de un documento estructurado y escrito en HTML.

El World Wide Web Consortium (W3C) es el encargado de formular las especificaciones de las hojas de estilo que servirán de estándar para los navegadores; y define:

Las Hojas de Estilo en Cascada, es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos (Luzardo Alliey, 2009).

CSS proporciona tres caminos diferentes para aplicar las reglas de estilo a una página Web:

- Hoja de estilo del documento, donde los estilos se aplicarán solo a los elementos de ese documento o página; que puede servir para ejemplos o sitios pequeños de un solo documento, pero no es útil para un sitio completo, ya que se tendrían que repetir los estilos en cada página.
- Hoja de estilo interna, método para insertar el lenguaje de estilo directamente dentro de una etiqueta HTML; este caso sirve para combinarlo con otros lenguajes como ASP, PHP o Java. De esta manera indica a los navegadores que ese estilo se aplicara solamente a un elemento concreto. Este tipo de estilo no permite separar diseño y contenido, dado que se incrusta dentro del código HTML, lo que provocará posibles errores.
- Hoja de estilo externa: recomendada por los estándares de la W3C. Los códigos de estilo se encuentran en un archivo externo (.css), que se enlazará con cada documento HTML de un sitio; permitiendo cambios sin tener que modificar el código HTML; y a la vez, concediendo la factibilidad de separar la estructura del diseño.

Características y ventajas de las CSS

Existen varias versiones de hojas de estilo en cascada desarrolladas por el World Wide Web Consortium (W3C), entre ellas CSS 1.0, CSS 2.0 y CSS 2.1.

CSS 1.0, fue la primera especificación aparecida en 1996 pero no aprobada hasta 1999, que permitía dar formato visual y una primera aproximación al posicionamiento de elementos HTML, con propiedades de letras, márgenes, colores, entre otras, manejando casi todos los perfiles que necesitan las hojas de estilo. CSS 2.0 aparece seguidamente con nuevas

funcionalidades, conteniendo todas las propiedades del nivel de las CSS 1.0 y añade elementos completamente colocados, numeración automática, saltos de página y otros.

A finales de 2007, aparece el CSS 2.1 que corrige una serie de errores que se incluyeron en el CSS 2.0 y mejora algunos aspectos. Hoy día el nivel 3 de las CSS está en elaboración e incluye todo del nivel 2 y extiéndelo con nuevos selectores, bordes y fondos finos, texto vertical, interacción con el usuario, habla, entre otras.

Si antes el HTML permitía una mínima capacidad para maquetar las páginas, utilizando trucos para conseguir algo de diseño, ahora con estas propiedades inmersas en las hojas de estilo, nos permitirán definir la presentación de un documento, con un renglón de grandes ventajas.

Los navegadores actuales están soportando cada vez más los estándares desarrollados para las hojas de estilo, con algunas diferencias, pero permitiendo la universalidad entre ellos.

Por otro lado, la diversidad de dispositivos actuales desde los que nos podemos conectar; teléfonos móviles, Padas, computadores, entre otros, hace que sea necesario separar y estructurar el contenido, con simples cambios en las hojas de estilo, de esta forma podremos visualizar nuestra aplicación web en todos los dispositivos, adaptando su apariencia al dispositivo y permitiendo accesibilidad desde cualquier plataforma.

Las paginas construidas con CSS podrían llegar a reducir un 30% del tamaño página, en comparación con una página desarrollada de forma obsoleta (tabla, capas, etc.) reduciendo los tiempos de carga. Menos código permite que las páginas web tarden menos en cargar, mejorando la experiencia de usuario.

En aspectos de posicionamiento una página en Internet, diseñada en base a los estándares web aparecerá en mejor posición en los resultados de búsqueda. Debido a que el código es más limpio, las paginas sólo llevan contenido (no diseño), y semánticamente es correcto.

El uso de CSS permite realizar cambios de formato en todo el sitio editando un único archivo; a su vez ofrece control por parte del usuario, ya que permite la personalización de su navegador para visitar el sitio.

En conclusión, los navegadores se están adaptando a los estándares, y a su vez estos estándares son desarrollados con un concepto universal buscando la versatilidad de múltiples plataformas, de esta forma se garantiza la viabilidad de los proyectos a largo plazo. CSS 2.1

es compatible con la mayoría de los navegadores y con un empleo adecuado de sus capacidades, sirve para cualquier plataforma.

Las ventajas de utilizar un lenguaje de estilo como los CSS son infinitas, permitiendo rediseños más rápidos con cambios inmediatos, sitios accesibles e independientes del dispositivo que lo despliega, entre otros beneficios (Luzardo Alliey, 2009).

Maquetando con CSS

La maquetación es una de las etapas más importantes a la hora de crear una página web, que no se puede omitir. “Maquetar” refiere a la composición de una página y su compaginación con diferentes elementos. Maquetar un diseño consiste en dar un formato a los documentos y a todo el conjunto de elementos que lo componen como imágenes, textos, entre otros.

Con el inicio de las páginas web, la técnica para maquetar un sitio era basada en tablas, pero a medida que la web ha venido evolucionando, esta técnica ha representado numerosos problemas para los desarrolladores y diseñadores.; presentando aspectos como, poca flexibilidad sin poder cambiar la distribución de los elementos de una página de forma sencilla, mala interpretación por parte de los navegadores, necesidad de mayor ancho de banda y dificultad de ser depuradas por los buscadores.

Como anteriormente se ha mencionado, las hojas de estilo en cascada ayudan a separar el contenido de la forma, permitiendo ajustar de una manera precisa cualquier aspecto de cualquier elemento de la página. Para esto los CSS permiten definir condiciones a los elementos dentro de una página con el uso de capas, utilizando las etiquetas <DIV>.

Las capas, layouts o divs refieren a un mismo concepto; especie de contenedores o bloques donde es posible introducir diversos elementos, a los que se le asigna un ancho, alto y posición, para que de esta manera se construya la estructura deseada.

La separación del contenido y del estilo, proporcionará cantidad de beneficios, eliminando los inconvenientes presentados por la maquetación con tablas, obteniendo páginas más limpias y claras en códigos (Luzardo Alliey, 2009).

IONIC FRAMEWORK

Ionic es una herramienta, gratuita y open source, para el desarrollo de aplicaciones híbridas basadas en HTML5, CSS y JS. Está construido con Sass y optimizado con AngularJS. (Ionic: Advanced HTML5 Hybrid Mobile App Framework, 2013)

Características

Alto rendimiento

La velocidad es importante. Tan importante que sólo se nota cuando no está en tu app. Ionic está construido para ser rápido gracias a la mínima manipulación del DOM, con cero jQuery y con aceleraciones de transiciones por hardware.

AngularJS & Ionic

Ionic utiliza AngularJS con el fin de crear un marco más adecuado para desarrollar aplicaciones ricas y robustas. Ionic no sólo se ve bien, sino que su arquitectura central es robusta y seria para el desarrollo de aplicaciones. Trabaja perfectamente con AngularJS.

Centro nativo

Ionic se inspira en las SDK de desarrollo móviles nativos más populares, por lo que es fácil de entender para cualquier persona que ha construido una aplicación nativa para iOS o Android. Lo interesante, como sabéis, es que desarrollas una vez, y compilas para varios.

Bonito diseño

Limpio, sencillo y funcional. Ionic ha sido diseñado para poder trabajar con todos los dispositivos móviles actuales. Con muchos componentes usados en móviles, tipografía, elementos interactivos, etc.

Un potente CLI

Con un sólo comando podrás crear, construir, probar y compilar tus aplicaciones en cualquier plataforma.

JSON

JSON se corresponde con las siglas *JavaScript Object Notación* y es un formato ligero para el intercambio de datos entre distintas partes. Es un lenguaje muy sencillo de leer y escribir por las personas, y fácil de ser analizado sintácticamente y ser generado por máquinas. Aunque es un subconjunto de la notación de JavaScript, JSON es un formato de texto completamente independiente de JavaScript o cualquier otro lenguaje de programación.

Los mensajes JSON se construyen siguiendo dos estructuras:

- Una colección de pares nombre/valor. En varios lenguajes, esto equivale a un *object*, *record*, *struct*, *dictionary*, *hash table*, *keyed list* o *associative array*.
- Una lista de valores ordenados. En muchos lenguajes, equivale a un *array*, *vector*, *list* o *sequence*.

En JSON, los mensajes toman la forma de *object* y *array* respectivamente, y se describen del siguiente modo:

Un *object* es un conjunto no ordenado de pares nombre/valor. Un *object* comienza con { (llave izquierda) y termina con } (llave derecha). Cada nombre es seguido por : (dos puntos) y el par nombre/valor separados por , (coma) (Introducing JSON, 2008).

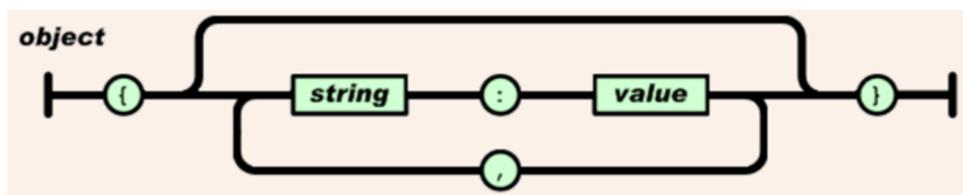


Figura 2.3.26 Construcción de un JSON object. Fuente: (Introducing JSON, 2008).

Un *array* es una colección ordenada de valores. Un *array* empieza con el carácter [(corchete izquierdo) y termina con] (corchete derecho). Los valores están separados entre sí por , (coma).

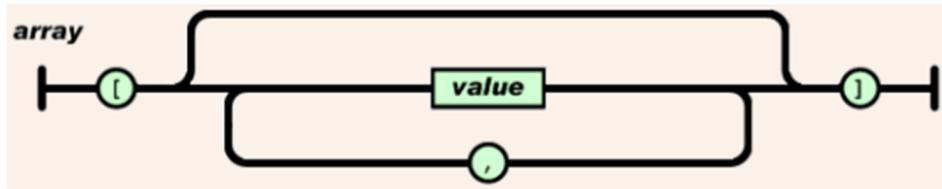


Figura 2.3.27 Construcción de un JSON array. Fuente: (Introducing JSON, 2008).

Los *objects* y *arrays* necesitan de *strings* y *values* para contener información útil. Se forman del siguiente modo:

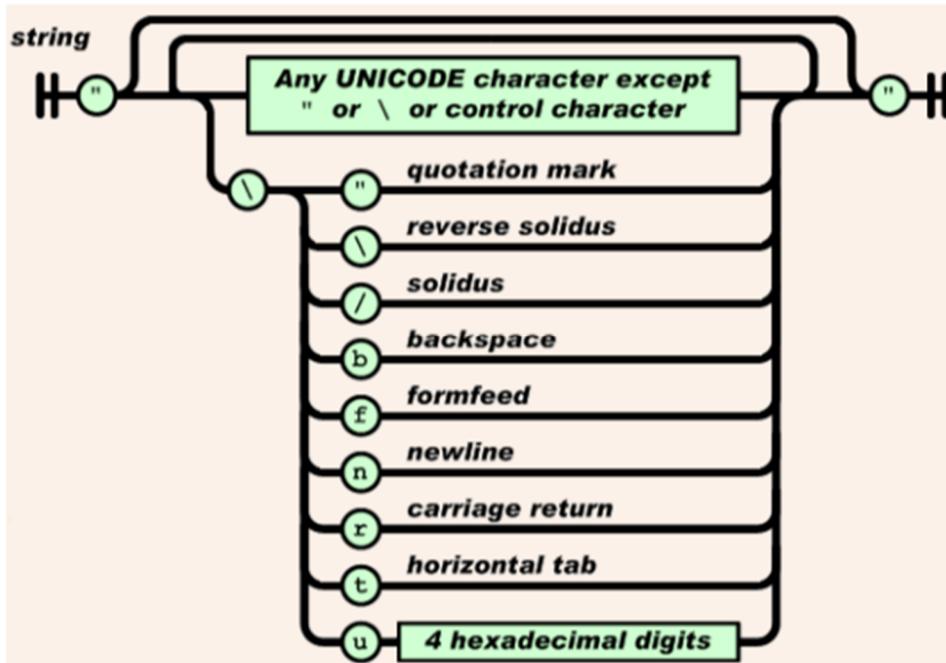


Figura 2.3.28 Formación de un string. Fuente: (Introducing JSON, 2008).

Un *string* es una colección de cero o más caracteres Unicode, envueltos entre " (comillas dobles). Un *string* de JSON es muy similar a uno de C o Java, pudiéndose utilizar escapes de barra invertida (\).

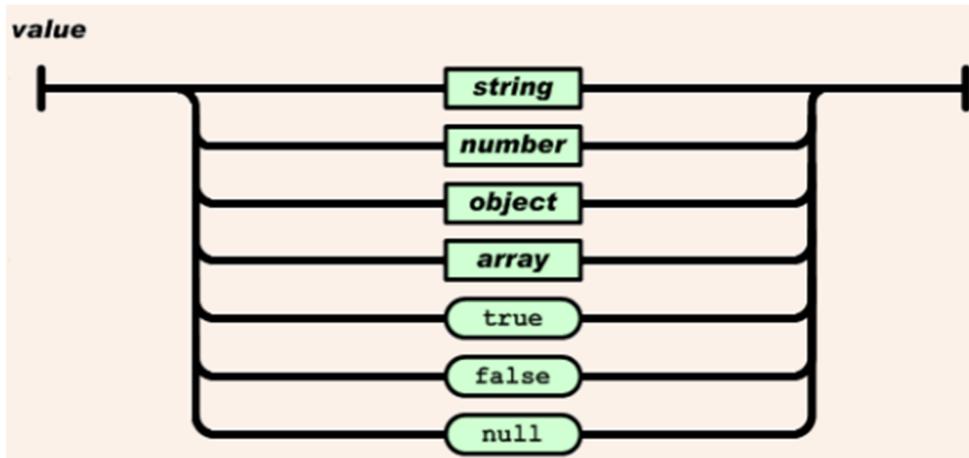


Figura 2.3.29 Opciones válidas para la formación de un value en JSON. Fuente: (Introducing JSON, 2008).

Un *value* puede ser un *string* entre " (comillas dobles), un *number*, *true* o *false* o *null*, un *object* o un *array*. Estas estructuras pueden anidarse.

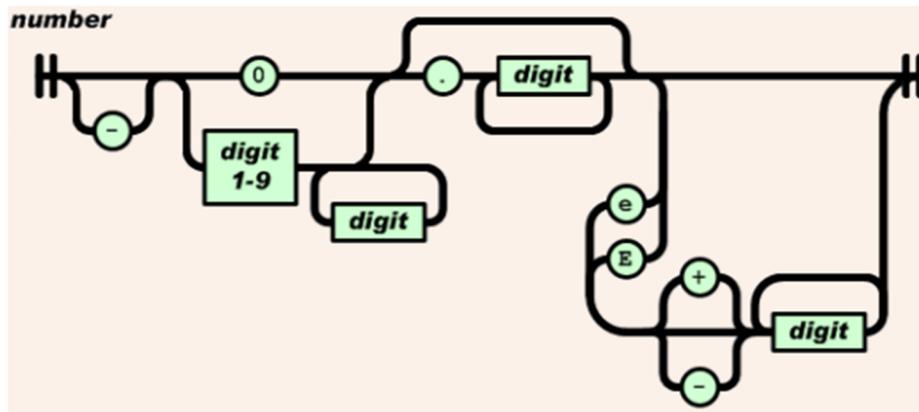


Figura 2.3.30 Formato de number en JSON. Fuente: (Introducing JSON, 2008).

Un *number* es muy similar a un número de Java o C, exceptuando que el formato octal y hexadecimal no se utilizan.

METODOLOGÍA RUP

El Rational Unified Process o Proceso Unificado de Racional. Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga la necesidad del usuario final dentro de un tiempo y presupuesto previsible. Es una metodología de desarrollo iterativo enfocada hacia los casos de uso, manejo de riesgos y el manejo de la arquitectural.

El RUP mejora la productividad del equipo ya que permite que cada miembro del grupo sin importar su responsabilidad específica acceda a la misma base de datos de conocimiento. Esto hace que todos compartan el mismo lenguaje, la misma visión y el mismo proceso acerca de cómo desarrollar software.

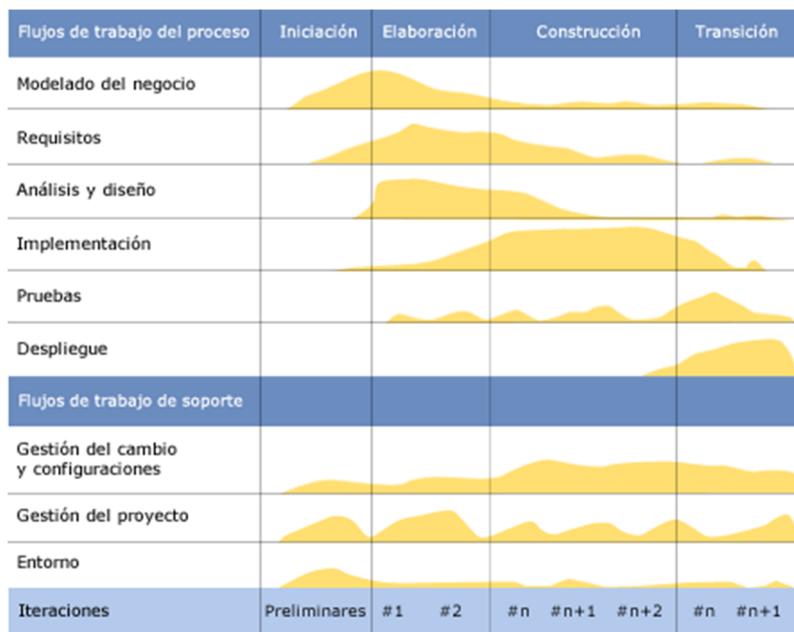


Figura 2.3.31 Ciclo de Vida de la Metodología RUP. Fuente: (https://jummp.files.wordpress.com/2011/04/rup_espanol.gif)

En el ciclo de vida RUP veremos una implementación del desarrollo en espiral. Con el ciclo de vida se establecen tareas en fases e iteraciones. El RUP maneja el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable

Las primeras iteraciones (en las fases de Inicio y Elaboración) se enfocan hacia la comprensión del problema y la tecnología, la delimitación del ámbito del proyecto, la eliminación de los riesgos críticos, y al establecimiento de una base de inicio.

Fases

- **Fase de Inicio**

En esta fase se definió y acordó el alcance del proyecto, identificando la factibilidad económica, técnica y operativa.

Se definieron los objetivos, la viabilidad, así como los riesgos relacionados del proyecto y los requerimientos.

- **Fase de Elaboración**

En esta fase se estableció la arquitectura base del sistema para proveer bases estables para el esfuerzo de diseño e implementación en la fase de construcción.

Se seleccionaron los casos de usos para la definición de la arquitectura base del sistema se realizó la especificación de los casos de uso seleccionados y el primer análisis del dominio del problema, diseñando la solución preliminar.

- **Fase de Construcción**

En esta fase se completó la funcionalidad del sistema, clasificando los requisitos pendientes, administrando los cambios de acuerdo a las evaluaciones y realizando las mejoras.

- **Fase de Transición**

Se hicieron las pruebas de funcionamiento, asegurando que el proyecto estuviera listo para los usuarios finales, posteriormente se hicieron unas pequeñas mejoras de configuración.

Principales Características

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo)
- Pretende implementar las mejores prácticas en Ingeniería de Software
- Desarrollo iterativo
- Administración de requisitos
- Uso de arquitectura basada en componentes
- Control de cambios
- Modelado visual del software

Verificación de la calidad del software El RUP es un producto de Rational (IBM). Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como, por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).

Especificación de las Fases

- Establece oportunidad y alcance
- Identifica las entidades externas o actores con las que se trata
- Identifica los casos de uso RUP comprende 2 aspectos importantes por los cuales se establecen las disciplinas:

✓ **Proceso:** Las etapas de esta sección son:

- Modelado de negocio
- Requisitos
- Análisis y Diseño

- Implementación
- Pruebas
- Despliegue

✓ **Soporte:** En esta parte nos conseguimos con las siguientes etapas:

- Gestión del cambio y configuraciones
- Gestión del proyecto
- Entorno

La estructura dinámica de RUP es la que permite que este sea un proceso de desarrollo fundamentalmente iterativo, y en esta parte se ven inmersas las 4 fases descritas anteriormente:

- Inicio (También llamado Incepción)
- Elaboración
- Desarrollo (También llamado Implementación, Construcción)
- Cierre (También llamado Transición)

Artefactos

RUP en cada una de sus fases (pertenecientes a la estructura estática) realiza una serie de artefactos que sirven para comprender mejor tanto el análisis como el diseño del sistema estos artefactos son los siguientes:

Inicio:

- ✓ Documento Visión
- ✓ Especificación de Requerimientos

Elaboración:

- ✓ Diagramas de caso de uso

Construcción:

- ✓ Documento Arquitectura que trabaja con las siguientes vistas:

Vista Lógica:

- ✓ Diagrama de clases
- ✓ Modelo E-R (Si el sistema así lo requiere)

Vista de Implementación:

- Diagrama de Secuencia
- Diagrama de estados
- Diagrama de Colaboración

Vista Conceptual:

- Modelo de dominio

Vista física:

- Mapa de comportamiento a nivel de hardware.

Implementación del RUP para el proyecto

La metodología RUP es más apropiada para proyectos grandes (Aunque también pequeños), dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas. En proyectos pequeños, es posible que no se puedan cubrir los costos de dedicación del equipo de profesionales necesarios.

DESARROLLO DEL PROYECTO

El desarrollo de la Plataforma autónoma para trámites y consultas de la Notaria Única de San Jacinto Bolívar está dividida en dos partes, el desarrollo de las aplicaciones web y móvil, y la programación e implementación del hardware.

Diseño del sistema hardware

Algunas consideraciones que se tuvieron en cuenta al momento de desarrollar el sistema fueron, desarrollar un sistema enfocado a personas con poca inmersión en el ámbito tecnológico y trabajar con hardware libre para la sencillez y economía del sistema. Es por esto, que se utilizaron elementos como Tablet (para visualizar), Arduino (con complemento) y una mini impresora de ticket (comunicación serial).

El siguiente bosquejo, muestra de una manera sencilla el diseño lógico del sistema:

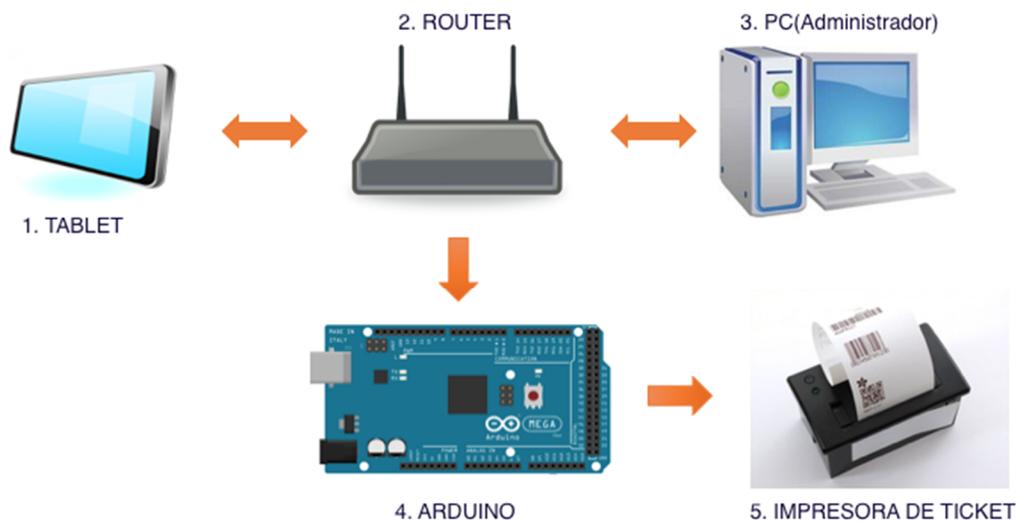


FIGURA 3.0.1 DISEÑO LÓGICO DEL SISTEMA. (DISEÑO PROPIO)

Funciones de los elementos del sistema.

1. Tablet.

Mediante una app, permite al usuario obtener información acerca de la documentación o preguntas guardadas en la base de datos.

2. Router.

Permite la comunicación del sistema.

3. PC (administrador).

De donde se administra toda la información, solo tiene el acceso el Notario o personal autorizado.

4. Arduino.

Con la condición de solicitar documentación, recibe la información del usuario, valor a cancelar y lo envía a la impresora de ticket.

5. Impresora de Ticket.

Imprime un recibo para recibir la documentación, indicando sus datos y el valor a cancelar en caja.

Implementación del Hardware.

Para el desarrollo del proyecto se escogió la plataforma de hardware libre Arduino, que permitirá procesar la información arrojada por el sistema luego que el usuario haya solicitado la impresión del documento. Por otra parte, se hace necesario la su uso para poner en marcha la impresora de ticket, ya que esta se comunica mediante la interfaz serie TTL.

Arduino.

Es una plataforma física de código abierto que se basa en una placa electrónica sencilla y con un software de entorno de desarrollo donde se puede escribir las instrucciones de ejecución de la placa. Esta placa es utilizada en el desarrollo de objetos interactivos, basados en las entradas y el control. Los proyectos Arduino, se pueden desarrollar de forma individual o pueden interactuar con otros softwares que se ejecutan en el ordenador como, por ejemplo: Flash, Processing, MaxMSP, etc.

Consideraciones puntuales de la plataforma Arduino tenidas en cuenta.

- Dado las condiciones de libertad, se hace ideal para el desarrollo de proyectos universitarios, de emprendimiento y empresariales sin violar los derechos de autor.
- La tarjeta Arduino permite el uso de sus puertos para hacer posible las conexiones con otros dispositivos para el desarrollo sistemas.
- Son de fácil adquisición en el mercado debido a su popularidad y bajo costo, al igual que los complementos que integran la plataforma Arduino.
- La gran cantidad de librerías, comunidades, foros, prototipos y documentación, permiten facilitar la programación, el uso y adaptación.
- Funciona en los sistemas operativos Windows, Macintosh OSX y Linux.
- El entorno de programación de Arduino, es fácil uso para personas que apenas están comenzando y flexible para usuarios avanzados.
- Permite la interacción con otros lenguajes y programas populares mediante la comunicación serial.

Para el proyecto se escogió la placa Arduino Mega2560.

Arduino Mega2560

Es una placa electrónica basada en el Atmega2560, que cuenta con 54 pines digitales de entrada / salida (de los cuales 15 se pueden utilizar como salidas PWM), 16 entradas analógicas, 4 UARTs (puertos seriales), un cristal oscilador de 16 MHz, una conexión USB,

un conector de alimentación, una cabecera ICSP, y un botón de reinicio. Lo cual contiene todo lo necesario para apoyar el microcontrolador, solo basta con conectarlo a un ordenador por medio de un cable USB o el poder con un adaptador de CA o la batería a CC para empezar a trabajar. La tarjeta Arduino Mega, es compatible con la mayoría de los escudos diseñados para el Arduino.

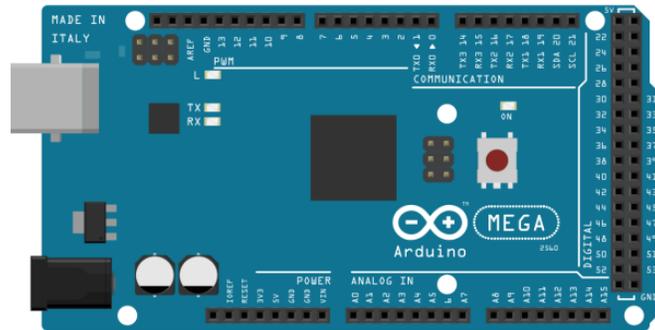


FIGURA 3.0.2 ARDUINO MEGA2560. FUENTE: ([HTTPS://WWW.ARDUINO.CC](https://www.arduino.cc))

Arduino Ethernet Shield

El Arduino Ethernet Shield permite que una tarjeta Arduino se conecte a Internet. Se basa en la Wiznet W5100 chip de ethernet (hoja de datos). El Wiznet W5100 proporciona una pila de red (IP) capaz de TCP y UDP. Soporta hasta cuatro conexiones de socket simultáneas. Usar la librería Ethernet para escribir bocetos que se conectan a Internet a través de la pantalla. El escudo ethernet se conecta a una placa Arduino usando cabezales largos que se extienden a través del blindaje. Esto mantiene el diseño del pasador intacto y permite que otro escudo se apile en la parte superior. (Arduino, 2016)

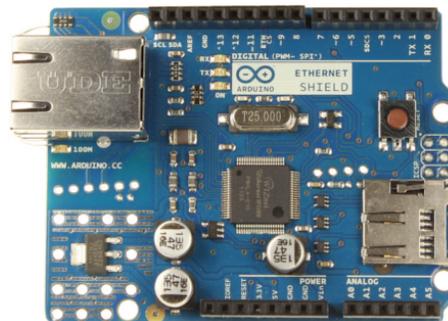


FIGURA 3.0.3 ARDUINO ETHERNET SHIELD. FUENTE: (ARDUINO ETHERNET SHIELD)

Impresora de Ticket



FIGURA 3.0.4. IMPRESORA DE TICKET. FUENTE([HTTPS://WWW.ADAFRUIT.COM/](https://www.adafruit.com/))

Especificaciones técnicas.

- Requiere 2.25 "de ancho, 50 pies o papel térmico más corto
- Requiere alimentación de 5-9VDC @ 1.5Amp durante la impresión
- Velocidad de impresión: 50-80mm / s
- Resolución: 8 puntos / mm, 384 puntos / línea
- Ancho de impresión efectiva: 48mm
- Conjunto de caracteres: ASCII, GB2312-80 (chino)
- Fuente de impresión: ANK: 5 × 7, chino: 12x24,24 × 24
- Tipo de papel: Papel térmico
- Ancho del papel: 57.5 ± 0.5mm
- Diámetro del rollo de papel: máx. 39mm
- Protocolo: Serie TTL, 19200 baudios
- Ciclos medios antes del fallo: 5 millones de líneas
- Fuente de alimentación (adaptador): DC5V-9V
- Esquema Dimensión (Ancho x Profundidad x Altura): 111x65x57mm
- Tamaño del puerto de instalación: 103 x 57mm
- De color negro
- Temperatura de funcionamiento: 5 ° C ~ 50 ° C
- Humedad de funcionamiento: 10% ~ 80%

- Temperatura de almacenamiento: $-20^{\circ}\text{C} \sim 60^{\circ}\text{C}$
- Humedad de almacenamiento: 10% ~ 90%
- Tornillos: M2.5

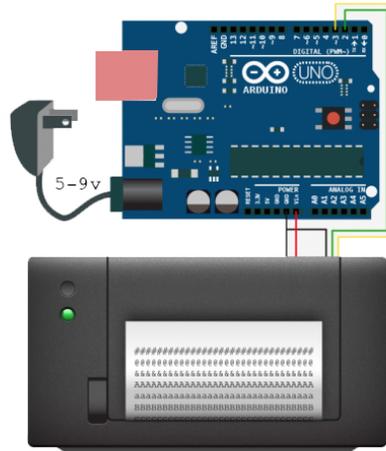


FIGURA 3.0.5. DIAGRAMA DE CONEXIÓN ARDUINO-IMPRESORA DE TICKET. FUENTE([HTTPS://WWW.ADAFRUIT.COM](https://www.adafruit.com))

En la figura 3.0.5, se muestra el diagrama básico para la conexión entre el Arduino Mega2560 y la impresora para la transmisión de información.

Códigos fuente

El código fuente creado para imprimir la información es el siguiente:

```
#include "Adafruit_Thermal.h"

#include "SoftwareSerial.h"
#define TX_PIN 6 // Arduino transmit  YELLOW WIRE  labeled RX on printer
#define RX_PIN 5 // Arduino receive  GREEN WIRE   labeled TX on printer

SoftwareSerial mySerial(RX_PIN, TX_PIN); // Declare SoftwareSerial obj first
Adafruit_Thermal printer(&mySerial);    // Pass addr to printer constructor

void setup() {

  pinMode(7, OUTPUT); digitalWrite(7, LOW);

  mySerial.begin(19200); // Initialize SoftwareSerial
  hardwareSerial
  printer.begin();      // Init printer (same regardless of serial type)

  // Set text justification (right, center, left) -- accepts 'L', 'C', 'R'

  printer.setSize('M');
  printer.println(F("-----"));
  printer.setSize('L');      // Set type size, accepts 'S', 'M', 'L'
  printer.justify('C');
  printer.println(F("NOTARIA UNICA"));
  printer.setSize('M');
  printer.println(F("San Jacinto"));
  printer.setSize('S');
  printer.println(F("Nit *****"));
  printer.setSize('M');
  printer.println(F("-----"));
  printer.setLineHeight(50);
  printer.setLineHeight(); // Reset to default

  printer.justify('L');
  printer.setSize('S');
  printer.println(F("Datos del solicitante"));
  printer.println(F("-----"));
  printer.println(F("Nombre:"));
  printer.println(F("N.I.:"));
  printer.println(F("Documento:"));
  printer.println(F("Valor:"));
  printer.setSize('M');
  printer.println(F("-----"));
  printer.setLineHeight(50);
  printer.println(F(""));
  printer.println(F(""));

  printer.sleep();      // Tell printer to sleep
  delay(3000L);         // Sleep for 3 seconds
  printer.wake();       // MUST wake() before printing again, even if reset
  printer.setDefault(); // Restore printer to defaults
}

void loop() {
}
```

Desarrollo de las aplicaciones Web y Móvil.

Para determinar las temáticas que llevan a la creación de aplicaciones web y móviles, se realizó inicialmente una investigación tanto bibliográfica como de páginas web, posteriormente se analizó la información y se seleccionaron las herramientas adecuadas para desarrollar con éxito el proyecto.

Elementos Utilizados

Para el desarrollo de este proyecto fue necesario la utilización de los siguientes equipos:

- Computador de Escritorio
 - Procesador Intel® Core™ i5-3210M CPU @ 2.50 GHz.
 - Memoria RAM de 2 GB.
 - Sistema operativo Windows 10 de 64 bits.

- Tablet
 - Procesador QuadCore 1.5Ghz A33 Cortex A7
 - Memoria RAM de 1GB y 8 GB de memoria Flash
 - Sistema Operativo Android 4.4
 - Teclado Inalámbrico.

- Software Sublime Text 2.
- MySQL Workbench.
- PHP 5.6
- Laravel 5.2
- JavaScript, Angular JS1

Selección De La Metodología Para El Desarrollo De Las Aplicaciones

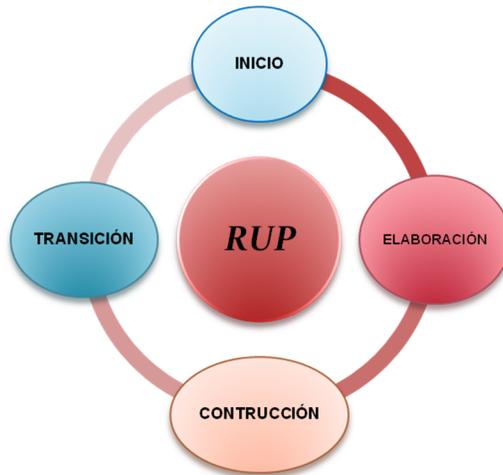


FIGURA 3.0.6. METODOLOGÍA RUP. FUENTE: ([HTTP://WWW.JLAYA.COM/METODOLOGIA-RUP-C](http://www.jlaya.com/metodologia-rup-c))

Existen numerosas metodologías para el desarrollo de software, en nuestro caso escogimos la Metodología RUP para el desarrollo de estas, ya que es una plataforma flexible en el proceso, que brinda guías consistentes y personalizadas de procesos para todo equipo de proyecto.

RUP permite seleccionar fácilmente el conjunto de componentes de proceso que se ajustan a las necesidades específicas del proyecto. Se pueden alcanzar resultados predecibles unificando el equipo con procesos comunes que optimicen la comunicación y creen un entendimiento común para todas las tareas, responsabilidades y artefactos.

Proceso de desarrollo

A continuación, se presentan los resultados obtenidos al aplicar la metodología RUP en el desarrollo de las aplicaciones del proyecto. Se describirán los artefactos(entregables) elaborados en cada una de las fases que comprenden la metodología antes mencionada.

Para el desarrollo del aplicativo web se realizó una instanciación de las cuatro fases que comprenden la metodología RUP.

A continuación, se presenta una breve descripción de cada una de las fases:

I. FASE DE INICIO

Esta es la primera fase de la metodología RUP. Durante el periodo que abarcó esta fase se elaboró un plan de proyecto, se identificaron los requerimientos del proyecto, así como su alcance. En esta fase se incluye también la lista de casos de uso no detallada, en los cuales tuvo base el desarrollo del proyecto, un glosario de términos y el contexto del negocio.

Visión del Proyecto

Este proyecto busca la optimización, así como la automatización y la mejora de la tramitología y consultas en la Notaria Única del municipio de San Jacinto Bolívar.

Con dicha automatización se busca que este proceso sea más eficiente para los usuarios.

Contexto del negocio

Para diligenciar documentos o realizar consultas en la Notaria, el usuario deberá tomar un turno, solicitar el documento por ventanilla, se hará búsqueda manual del documento por parte de personal de apoyo logístico, se procede a revisión y firma del Notario y finalmente es entregado al usuario el documento solicitado. El promedio de atención de un usuario tarda entre 15 y 30 minutos, dependiendo del tipo de documento solicitado, lo cual genera colas y demoras en la atención a los usuarios de la misma. Adicionalmente en muchas ocasiones los usuarios se acercan para realizar preguntas acerca de tramitología para un fin particular, sin embargo, también deben hacer las filas de espera para obtener la información requerida.

Debido a esto surgió la necesidad de automatizar el proceso. La idea es el desarrollo de un aplicativo web, complementado con uno móvil, que soporte este proceso de trámites y consultas, que sea amigable para el usuario y el personal administrativo, y que además permita agilizar dicho proceso.

A continuación, se presentan los requerimientos del proyecto:

1. Requerimientos

Los requerimientos fueron clasificados en dos tipos, requerimientos Funcionales y requerimientos No Funcionales. A continuación, se presenta una lista que contiene los requerimientos Funcionales; y seguida de esta lista se presenta una lista con los requerimientos No Funcionales.

a) Requerimientos Funcionales

Registro, consulta, modificación y eliminación de los datos básicos:

El sistema debe permitir la realización de las actividades antes mencionadas con los siguientes datos:

- Cedula
- Nombre
- Número de registro

Para más información diríjase al manual técnico sección Casos de usos.

b) Requerimientos No Funcionales

- *Requerimientos de Funcionalidad:*

El sistema debe permitir el acceso con distintos navegadores Web; como, por ejemplo: Mozilla, Google Chrome, entre otros.

- *Requerimientos de Usabilidad*

- El sistema debe ayudar a agilizar las consultas de los documentos de los usuarios y preguntas frecuentes.
- Las interfaces del sistema deben ser muy sencillas de utilizar (intuitivas) para cualquier usuario del mismo.

- *Requerimientos de Eficiencia*
 - El sistema debe tener alta disponibilidad, todos los días del año.

- *Requerimientos de Mantenibilidad*
 - El sistema debe permitir, la incorporación de nuevas funcionalidades (nuevos componentes).
 - A medida que pase el tiempo, la base de datos va ir creciendo, así que se debe asegurar un espacio adecuado para esta.

Modelo inicial de casos de usos

Actores:

A continuación, se da una breve descripción de la especialización de cada uno de los actores del modelo inicial de casos de uso del sistema.

Usuario: Es cualquier persona que tiene acceso al sistema, dependiendo del rol.

- a) Administrador del Sistema(Notario):** encargado de realizar las operaciones que realicen algún tipo de modificación (registro, consulta, modificación y eliminación) a los datos básicos, en este caso el personal administrativo o el Notario y será posible solo desde el aplicativo web.
- b) Consultor:** Usuario o persona del común que requiera realizar una consulta en la base de datos, por medio del aplicativo móvil instalado en el módulo.

Lista Inicial de Casos de Usos

- Gestionar información de personas
- Gestionar panel de preguntas
- Gestionar panel de documentos

- Consultar documentación
- Realizar preguntas

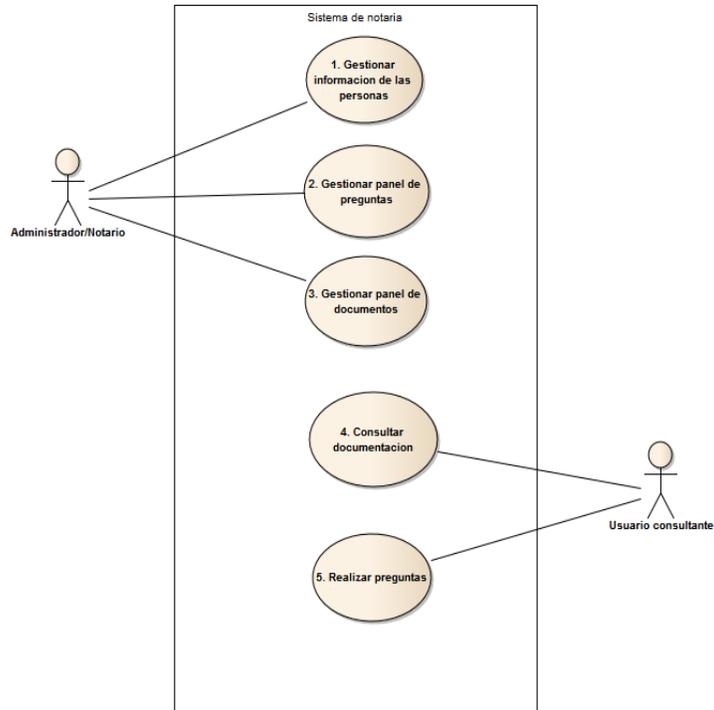


FIGURA 3.0.7. MODELO INICIAL DE CASOS DE USO. (DISEÑO PROPIO)

Para ver una breve descripción de cada caso de uso, vaya al manual técnico.

Lista de Riesgos

- Poco tiempo para el desarrollo del proyecto
- Recursos para el desarrollo del proyecto
- Distancia entre donde se estructuró el proyecto y donde se implementó.

Glosario del Sistema:

El glosario del sistema contiene los conceptos que van a ser utilizados a lo largo del desarrollo del sistema. Estos conceptos son definidos debido a que están presentes palabras no conocidas, o palabras que en algún momento puedan sonar y notarse confusas.

II. FASE DE ELABORACION

Segunda fase de la metodología RUP. Durante el periodo que abarcó esta fase en el proyecto, se realizó la identificación de los requerimientos complementarios, así como también el refinamiento de los casos de uso del sistema, también se realizó la descripción de la arquitectura de software.

a) *Requerimientos Complementarios:*

En este punto se plantean los requerimientos complementarios que surgieron luego de haber presentado la lista de requerimientos, elaborada en la fase de Inicio al Notario.

- El modulo debe entregar un ticket que contenga el valor del documento solicitado, así como los datos personales del usuario.
- Para el acceso a la manipulación de los datos por parte del aplicativo web, este debe solicitar usuario y contraseña.

b) *Lista revisada de Riesgos*

En esta sección se presenta una lista revisada de los riesgos que fueron encontrados en la fase de Inicio.

Nombre	Poco tiempo para el desarrollo del proyecto
Variación	Disminuyó
Descripción	Se verificaron los aspectos mínimos que debe alcanzar la aplicación, para así dejar claro los requerimientos que se deben cumplir para el momento de la entrega, adicionalmente se alargó el tiempo de entrega.

TABLA 1. POCO TIEMPO PARA EL DESARROLLO DEL PROYECTO.

Nombre	Recursos para el desarrollo del proyecto
Variación	Disminuyó
Descripción	Una vez decidida la herramienta de desarrollo y estudiados los requerimientos de la misma se procedió a adquirir el hardware para la comodidad de desarrollo, mientras se completaba el análisis del sistema. Para así comenzar el desarrollo a tiempo.

TABLA 2. RECURSOS PARA EL DESARROLLO DEL PROYECTO. (DISEÑO PROPIO)

Nombre	Distancia entre donde se estructuró el proyecto y donde se implementó.
Variación	Se mantuvo
Descripción	Se cumplieron las visitas previstas como se organizó en el plan del proyecto.

TABLA 1. DISTANCIA ENTRE DONDE SE ESTRUCTURÓ EL PROYECTO Y DONDE SE IMPLEMENTÓ. (DISEÑO PROPIO)

III. FASE DE CONSTRUCCION

a) Diseño de las Interfaces

El aplicativo está dividido en dos partes, la aplicación móvil, que será la interfaz para el usuario y la aplicación web que será manipulada por el Notario y personal administrativo que este considere, Por esto es necesario que dicha interfaz sea amigable y fácil de usar.

Los usuarios, podrán acceder a la información almacenada en la base de datos privada de la Notaria atreves de el aplicativo móvil que está instalado en la Tablet, la cual representa la pantalla del módulo.

Por su parte los usuarios en el aplicativo móvil podrán acceder a dos opciones que son, Buscar Documentos y Realizar una Pregunta, solo basta con ingresar el

número de documento del solicitante en la primera opción o escribir una palabra en la opción de preguntas para que el sistema lo relacione con la información en la base de datos.

Se diseñó una aplicación móvil con el framework Ionic ya que permite realizarlas de manera híbrida para que corra en diferentes sistemas operativos móviles.

En la pantalla inicial solo existirán dos botones, que como anteriormente se dijo son para buscar documentos por medio de la cédula de ciudadanía y el otro para realizar una consulta o inquietud relacionado con temas Notariales.

Por otra parte, para el aplicativo web, que es manejado por el Notario o personal administrativo, lleva más partes. Se diseñó una especie de panel de control, donde el administrador podrá gestionar la información que solicitara el Usuario, como son documentos y preguntas. Por un lado, se estableció que el administrador, en este caso el Notario, sea el encargado de ingresar a su parecer las preguntas y las respuestas que serán mostradas en el aplicativo web.

El prototipo funcional desarrollado en este proyecto abarca los requerimientos especificados en la fase de Inicio junto con los requerimientos complementarios establecidos en la fase de elaboración.

b) Mapa de Navegación de los aplicativos.

El mapa de Navegación es una idea de cómo el usuario puede navegar en las aplicaciones.

(Esta información se encuentra detallada en el manual técnico).

c) Descripción del Prototipo por Actor, con Pantallas:

En este caso existen dos Roles o Actores uno para cada aplicación, y para cada una existen distintas funcionalidades y por ende distintas pantallas; debido a eso la descripción de las pantallas se hará para cada Actor, para que se vean claras las funcionalidades y quien las puede ejecutar.

Para el aplicativo web, que es donde el Notario administra la información es una pantalla que presenta unos “tabs” en la parte superior que indican el acceso a las distintas funcionalidades.

(Para tener mayor información, consultar el manual del usuario).

IV. Fase de Transición

En esta fase se realizaron las pruebas integradas del sistema para medir su calidad y además se elaboraron el manual técnico y del usuario, estos últimos con el fin de permitir a las personas relacionadas con las aplicaciones familiarizarse con el sistema y para futuras revisiones del mismo.

1. Pruebas Integradas:

Se realizaron pruebas funcionales y pruebas no funcionales, es decir, se realizaron pruebas sobre las funcionalidades del sistema y se hicieron pruebas de usabilidad, eficiencia, entre otras; las cuales son llamadas pruebas no funcionales. Con estas pruebas se busca analizar la calidad.

2. Pruebas Funcionales:

Para la realización de estas pruebas se consideraron los Casos de Uso, es decir, las pruebas fueron realizadas con base en los Casos de Uso: para cada Caso de Uso se realizaron distintas pruebas, para observar la completitud de las funcionalidades. En estas las aplicaciones no presentaron fallas.

Los resultados obtenidos se clasificaron en:

- Éxito: Cuando se cumple con la función.
- Fallas leves (FL): Falla del sistema que no afecta la funcionalidad.
- Fallas graves (FG): Falla que afecta la funcionalidad del sistema.

En la siguiente tabla se puede observar los resultados de las pruebas por caso de uso.

CU	Pruebas	Éxitos	FL	FG	%E	%FL	%FG
CU-1	50	50	0	0	100	0	0
CU-2	50	50	0	0	100	0	0
CU-3	50	50	0	0	100	0	0
CU-4	50	50	0	0	100	0	0
CU-5	50	50	0	0	100	0	0
CU-6	50	50	0	0	100	0	0
CU-7.1	50	50	0	0	100	0	0
CU-7.2	50	50	0	0	100	0	0
CU-7.3	50	50	0	0	100	0	0
CU-7.4	50	50	0	0	100	0	0
CU-8.1	50	50	0	0	100	0	0
CU-8.2	50	50	0	0	100	0	0
CU-8.3	50	50	0	0	100	0	0
CU-8.4	50	50	0	0	100	0	0
CU-9	50	50	0	0	100	0	0
CU-10	50	50	0	0	100	0	0

TABLA 2. RESULTADO DE LAS PRUEBAS. (ELABORACIÓN PROPIA)

Leyendas:

CU	CASO DE USO
FL	FALLAS LEVES
FG	FALLAS GRAVES
%E	EXITOS
%FL	FALLAS LEVES
%FG	FALLAS GRAVES

Promedio de éxitos por Caso de Uso: 100%

Promedio de FL por caso de Uso: 0%

Como se puede notar que no se encontraron fallas para ningún caso de uso, por lo cual no hubo correcciones para esta etapa.

3. *Pruebas No Funcionales:*

Para estas pruebas se consideraron los Requerimientos no Funcionales, es decir, se hicieron pruebas sobre las categorías *Usabilidad, Eficiencia y Portabilidad*; la categoría *Funcionalidad* es considerada la más importante, debido a que identifica la capacidad del software para cumplir con las funciones.

4. *Material Soporte para los Usuarios Finales:*

Se elaboraron 2 manuales para los usuarios finales con la finalidad de guiar a los usuarios en el uso, operación y mantenimiento el producto. A continuación, se muestra una breve descripción del contenido de estos manuales.

a. Manual del Usuario

El propósito principal de este manual es para guiar al usuario en cómo usar el producto de software. En éste se describe como es la navegación entre las pantallas del sistema, para las distintas funcionalidades y por ende para los distintos actores, es decir, describe como llevar a cabo las distintas funcionalidades a través de las pantallas del sistema.

b. Manual Técnico

El propósito de este manual técnico es proporcionar al lector, la lógica con la que se ha desarrollado las aplicaciones, la cual se sabe que es propia de cada programador, por lo que se considera necesario documentarla.

Arquitectura del sistema

ARQUITECTURA MVC(Modelo-Vista-Controlador): El sistema se desarrolló bajo la arquitectura MVC basándose en los requerimientos funcionales y no funcionales, con el fin de separar la lógica de negocio tanto de los datos como de la vista de los usuarios, dada la flexibilidad para el acceso. Teniendo como finalidad la generación de un sistema enfocado interoperabilidad, escalabilidad y la posibilidad de tener aplicaciones distribuidas en varias ubicaciones.

(Para información detallada, consulte el manual técnico).

RESULTADOS

Pruebas

Se realizaron pruebas de funcionamiento durante cinco semanas seguidas, poniendo el modulo a disposición de los Usuarios de la Notaria de San Jacinto.



FIGURA 4.1 NOTARIA ÚNICA DE SAN JACINTO BOLÍVAR.

- Base de Datos

Se realizó el ingreso de documentos directamente y se realizaron consultas desde la app del módulo para verificar la operatividad, resultando normal y sin inconvenientes.

- Módulos

Al terminar cada módulo, se realizaron las pruebas correspondientes para la verificación de la funcionalidad de cada uno y que cumplieran con las tareas de forma correcta, de los cuales se obtuvo 100% de efectividad y funcionalidad.

Las pruebas realizadas sobre la plataforma autónoma para tramites y consultas Notariales se hizo de manera periódica durante el tiempo que estuvo a disposición del personal, se evaluó su funcionamiento y se proporcionaron críticas constructivas acerca de lo que se tiene por mejorar.

Primero se evaluó el funcionamiento de la base de datos, ingresando por parte del administrador documentos digitalizados, la cual no presentó ningún inconveniente.

Posteriormente se probó la app del módulo antes de poner en disposición al público.



FIGURA 4.2 MODULO INSTALADO EN LA NOTARIA

En la Figura 4.2 se muestra la plataforma finalmente instalada en la Notaria de San Jacinto.

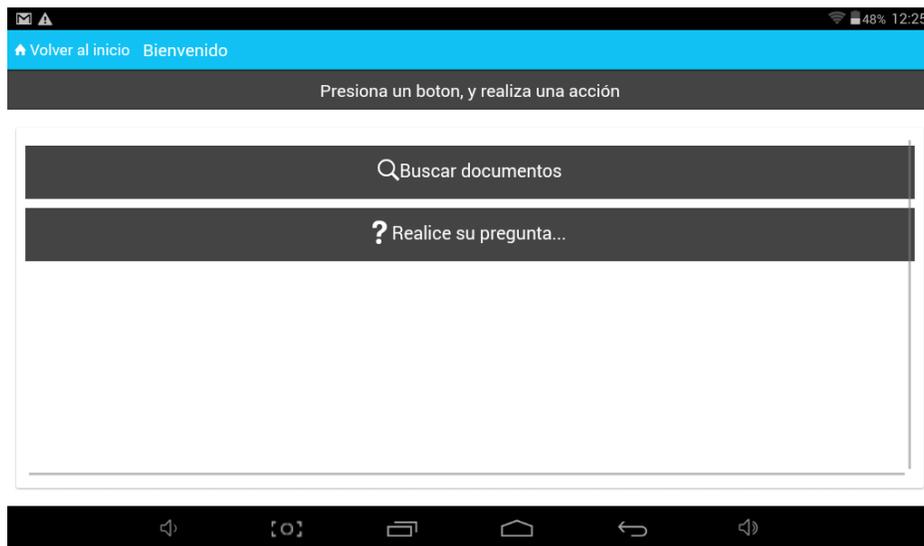


FIGURA 4.3 INTERFAZ GRÁFICA DEL USUARIO

El aplicativo móvil instalado en una Tablet para la interacción del usuario con la información, consta de una interfaz gráfica sencilla y amigable, que solo consta de dos botones, lo cual funcionó perfectamente puesto que resultó sencillo para las personas, cumpliendo el objetivo planteado.

Este aplicativo muestra dos botones como se ve en la figura 4.3, recordando que este se intercomunica por medio de una red interna de la Notaria para acceder a la base de datos.

Por tratarse de un proyecto basado en aplicaciones web y móvil, una de las pruebas más importantes fue la navegación sobre estas y los tiempos de respuesta, las cuales se realizaron en diferentes navegadores y dieron como resultado un funcionamiento óptimo.

RESULTADOS OBTENIDOS

El proyecto pasó con éxito las pruebas, su desempeño fue mejor de lo esperado y en general se lograron los siguientes resultados:

- Se obtuvo el conocimiento necesario acerca de las bases de datos que se usaron, logrando así que pudieran servir de fuentes de datos para el proyecto.
- Se realizaron un conjunto de consultas interesantes que probaron la efectividad del sistema, validando así la funcionalidad del servicio.
- En ningún momento el sistema arrojó errores.

CONCLUSIONES

Al realizar este proyecto de tesis, titulado Diseño e implementación de una plataforma para tramites y consultas de la Notaria Única de San Jacinto Bolívar, se pusieron a prueba las habilidades adquiridas durante nuestra formación como ingeniero para resolver problemas, tanto en la obtención de información, así como en el desarrollo del mismo.

Con la elaboración de este sistema se pudo comprender mejor algunos conceptos relacionados al área de desarrollo de software, así como adquirir nuevos conocimientos y el recordar algunos otros aprendidos durante el transcurso de nuestra estancia académica en la Facultad de Ingeniería, que fueron la base para poder desarrollar dicho sistema.

El objetivo fundamental de este proyecto es la implementación de un dispositivo capaz de agilizar la tramitología con respecto a documentación y preguntas relacionadas con temas Notariales, realizado con elementos de bajo costo.

Mediante la implementación del sistema en la Notaria de San Jacinto se pudieron sacar las siguientes conclusiones:

- Al utilizar la plataforma en la Notaria en un tiempo de tres meses, se logró total aceptación y generó curiosidad por parte de los usuarios.
- Se diseñó el sistema para Notaria Única de San Jacinto que cumple con los requerimientos y se desarrolló un prototipo funcional del mismo, que abarca todos los Casos de Uso pautados, el cual fue acordado con el Notario
- El uso de la plataforma autónoma ayuda a brindar un mejor servicio en tiempos de respuesta, además de ofrecer una interfaz amigable y fácil de utilizar.
- La utilización de un aplicativo web como control de la información por parte del administrador se mostró favorable en el sentido de la versatilidad y comodidad para la manipulación de dicha información.

- La metodología de desarrollo usada para llevar a cabo el proyecto, *Rational Unified Process (RUP)*, es adaptable al desarrollo, garantizando que sólo se lleven a cabo aquellas actividades necesarias o útiles para el proyecto.

Este proyecto cumple las expectativas esperadas, sin embargo, se considera que es la primera etapa y está expuesto a mejoras innovadoras.

RECOMENDACIONES

Si bien se ha comentado dentro del desarrollo del proyecto, este se realiza con el fin de brindar una herramienta de apoyo para la Notaria Única del municipio de San Jacinto del departamento de Bolívar. El trabajo a futuro puede considerarse a fin de las labores que se efectúan durante la vida útil del proyecto. Cualquiera sea el objeto de perfeccionamiento e inclusión de nuevas tecnologías, es muy importante contar con la participación de los involucrados en el funcionamiento del proyecto.

A continuación, se presentan algunas ideas que pueden ser desarrolladas como trabajo futuro sobre la temática mostrada en este proyecto de investigación.

- Implementar un sistema de reconocimiento de voz que permita a los usuarios realizar preguntas por medio del habla, con el fin de mejorar la interacción con la plataforma y por estar expuesto a público, servir de complemento para las personas que presentan analfabetismo funcional o incapacidad visual.
- Implementar un sistema para la detección del código de barras para cédulas e incluir o desarrollar un generador de código para la autenticación de los documentos de tal manera que dificulte la falsificación.

PRESUPUESTO

Recursos Personales			
Personal	Tiempo(meses)	Horas(semanas)	Costo
Investigador Javier Castañeda	7	30	-
Investigador Holman Ortega	7	30	-
Asesor de Investigación	9	2	1.200.000

TABLA 7.1 RECURSOS PERSONALES.

Recursos Institucionales			
Equipo	Tiempo	Cantidad	Costo Total

TABLA 7.2 RECURSOS INSTITUCIONALES

Recursos Adicionales			
Servicios	Cantidad	Precio Und.	Costo Total
Transporte	4	100.000	400.000
Librería			
Componentes	3	-	500.000
Electrónicos	1	250.000	250.000
Estructura			

TABLA 7.3 RECURSOS ADICIONALES.

Costo Total del Proyecto	
Concepto	
Recursos Personales	1.200.000
Recursos Institucionales	0
Recursos Adicionales	1.150.000
Costo Total	2.350.000

TABLA 7.4 COSTO TOTAL DEL PROYECTO

BIBLIOGRAFÍA

- Indep Consultores, S.L. (s.f.). *Aplicaciones Web a Medida*. Recuperado el 15 de 03 de 2016, de Indep: http://www.indep.es/aplicaciones_web
- REDACCIÓN TECNÓSFERA, EL TIEMPO . (14 de septiembre de 2015). *Periodico EL TIEMPO*. Obtenido de Tramites del estado en internet: www.eltiempo.com
- Ian Sommerville, M. I. (2005). *Ingeniería del Software* . Pearson Educación.
- Plattini, M. G., Calvo-Manzano, J. A., Cervera, J., & Fernández, L. (2004). *Análisis y diseño de Aplicaciones Informáticas de Gestión* (1ª ed.). Mexico D.F.: Alfaomega.
- Tanenbaum, A. S. (2009). *Sistemas Operativos Modernos* (3ª Edición ed.). México D.F., México: Prentice Hall, Pearson.
- Luján Mora, S. (2002). *Programación de Aplicaciones Web: Historia, Principios básicos y Clientes Web*. San Vicente, Alicante, España: Editorial Club Universitario.
- Kioskea. (Junio de 2014). *Intranet y Extranet*. Recuperado el 26 de Junio de 2016, de Kioskea : <http://es.kioskea.net>
- Schmuller, J. (2001). *Aprendiendo UML en 24 horas*. Prentice-Hall.
- Elmasri, R., & Navathe, S. (2002). *Fundamentals Of Database Systems* (3ra Edición ed.). Madrid, España: Pearson Educación.
- Piñero Gómez, J. (2004). *Definición y manipulación de datos*. Bilbao, España: Ediciones Paraninfo.
- Velázquez Medina, R., & Hueyo Chino, M. C. (2015). *Construcción de una página web con PHP y LATEX para el aprendizaje de las matemáticas*. Tesis Profesional, Universidad Autónoma de Puebla, Puebla.
- Sánchez, J. (2003). Manual de referencia de JavaScript. *JavaScript*.
- Solis, C. (2015). *Manual del Guerrero: AngularJS*. Obtenido de Manual del Guerrero: http://manualdelguerrero.com/angularjs/adelanto/manual_angularjs_preview.pdf

Rivero Dorta, M. J. (2016). *AngularJs Paso a Paso*. Leanpub.

Luzardo Alliey, A. M. (2009). *Diseño de la interfaz grafica web en función de los dispositivos móviles*. Tesis, Universidad de Palermo, Buenos Aires.

Ionic: Advanced HTML5 Hibrid Mobile App Framework. (2013). Recuperado el 05 de 2016, de Ionic: <http://ionicframework.com>

Introducing JSON. (2008). Recuperado el 05 de 2016, de JSON: <http://www.json.org/>

REDACCIÓN TECNÓSFERA, EL TIEMPO. (s.f.).

Pressman, R. S. (2010). *La Naturaleza Única de las Webapps (7ª Edición ed.)*. (V. C. Olguín, & J. E. Brito, Trads.) Mexico D.F.: Mc Graw Hill.

Berzal , F., & Cortijo, F. (s.f.). *Desarrollo Profesional de Aplicaciones Web con ASP.NET*. (i. Consulting, Ed.)

TRIESA. (s.f.). *Desarrollo de Sistemas* . Obtenido de Tecnología en Redes e informática Especializada S.A. de C.V.: <http://www.triesa.com.mx/frames/desarrollo.htm>

Fernandez Moreno, D. (2006). *Diseño e implementación de una aplicación web para la gestión y control de visitas a centros educativos*. Proyecto de Fin de Carrera, Madrid.

Historia de PHP. (s.f.). Recuperado el 26 de 05 de 2016, de PHP: Hypertext Preprocessor: <http://php.net/manual/es/history.php.php>

HostaliaWhitepapers. (s.f.). *Laravel, un framework de PHP*. Recuperado el 2016, de Hostalia: http://pressroom.hostalia.com/wp-content/themes/hostalia_pressroom/images/framework-laravel-wp-hostalia.pdf

Arduino. (2016). *Arduino - ArduinoEthernetShield*. Obtenido de Arduino: <https://www.arduino.cc/en/Main/ArduinoEthernetShield>

Capítulo 9

ANEXOS



Figura 9.1 Modulo en prueba



Figura 9.2 Modulo en prueba



Figura 9.3 Modulo en prueba



Figura 9.4 Modulo en prueba



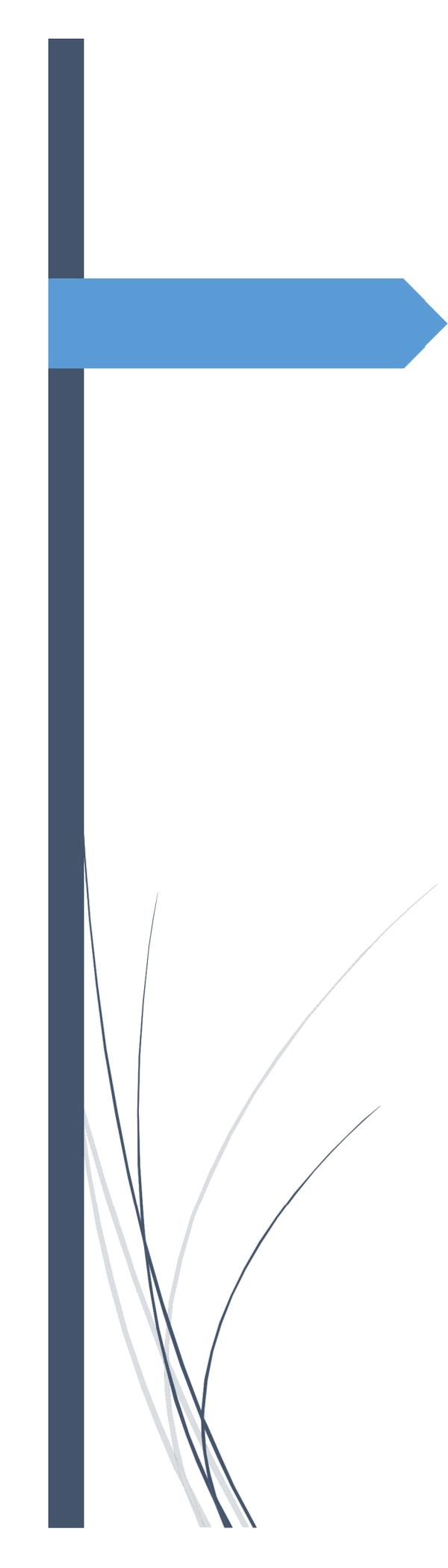
Figura 9.5 Modulo en prueba



Figura 9.6 Modulo en prueba



Figura 9.7 Modulo en prueba



Manual Del Usuario

Holman Ortega;Javier Castañeda

TABLA DE CONTENIDO

PAGINA PRINCIPAL	3
AGREGAR PERSONAS	4
LISTADO DE PERSONAS	7
BUSCAR PERSONAS.....	8
ACTUALIZAR DATOS PERSONAS	10
BORRAR PERSONAS	13
AÑADIR DOCUMENTOS.....	16
PANEL DE PREGUNTAS.....	18
PANEL DE TIPO DOCUMENTOS.....	24

- PAGINA PRINCIPAL

En la Imagen 1 vemos la pagina principal en la cual encontramos diversas opciones para la administracion y manejo del sistema. En la pagina principal tenemos como primera opcion “Agregar Personas” aquí se pueden registrar los usuarios para acelerar tramites en el futuro.

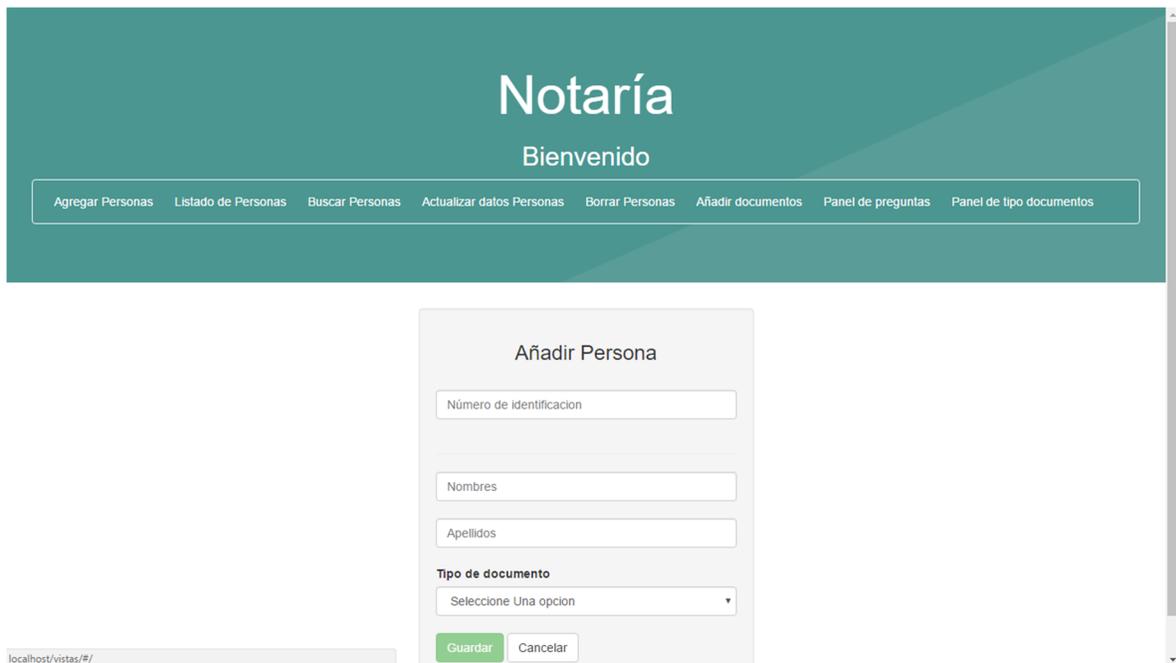


Imagen 1

- AGREGAR PERSONAS

Para agregar un nuevo usuario llenamos las casillas con los datos que nos solicitan, los cuales son:

- Numero de identificacion (1)
- Nombres (2)
- Apellidos (3)

The image shows a web application interface for a notary office. The main header is green and contains the text "Notaría" and "Bienvenido". Below the header is a navigation bar with several menu items: "Agregar Personas", "Listado de Personas", "Buscar Personas", "Actualizar datos Personas", "Borrar Personas", "Añadir documentos", "Panel de preguntas", and "Panel de tipo documentos".

In the foreground, a modal form titled "Añadir Persona" is displayed. It contains four input fields, each highlighted with a red box and a number to its right:

1. "Número de identificación" (Text input)
2. "Nombres" (Text input)
3. "Apellidos" (Text input)
4. "Tipo de documento" (Dropdown menu with the text "Seleccione Una opcion")

At the bottom of the form, there are two buttons: "Guardar" (green) and "Cancelar" (white).

Imagen 2

Despues de digitar los datos solicitados procedemos a dar click en la opcion “tipo de documento (4)”

Una vez hacemos click en “tipo de documento” se nos despliega un menu de opciones en este caso seleccionamos “Cedula” y damos click en “Guardar”, inmeditamente vemos un mensaje en pantalla que nos dice que el usuario fue registrado exitosamente como podemos apreciar en la “Imagen 4”



Imagen 3



Imagen 4

- LISTADO DE PERSONAS

Haciendo click en el apartado “Listado de personas” podremos ver todos usuarios registrados en nuestra base de datos organizados por su numero de identificacion.

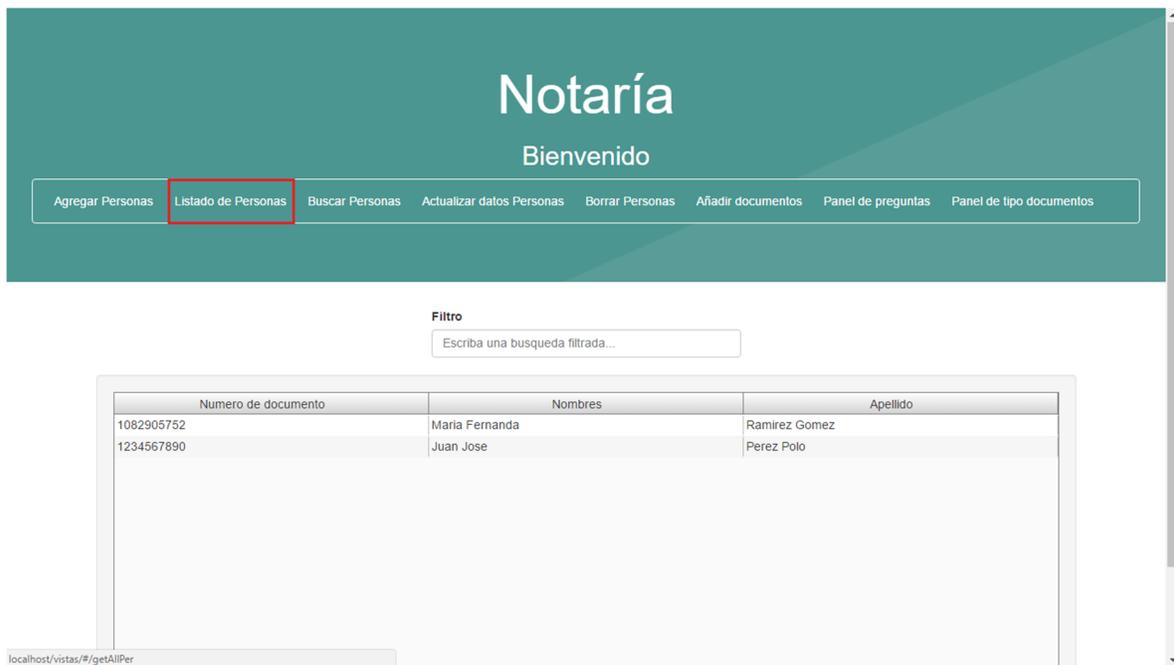


Imagen 5

- **BUSCAR PERSONAS**

Haciendo click en el boton “Buscar Personas” tenemos la opcion de buscar en nuestra base de datos los usuarios registrados, para esto digitamos su numero de documento en la casilla y presionamos “Buscar”

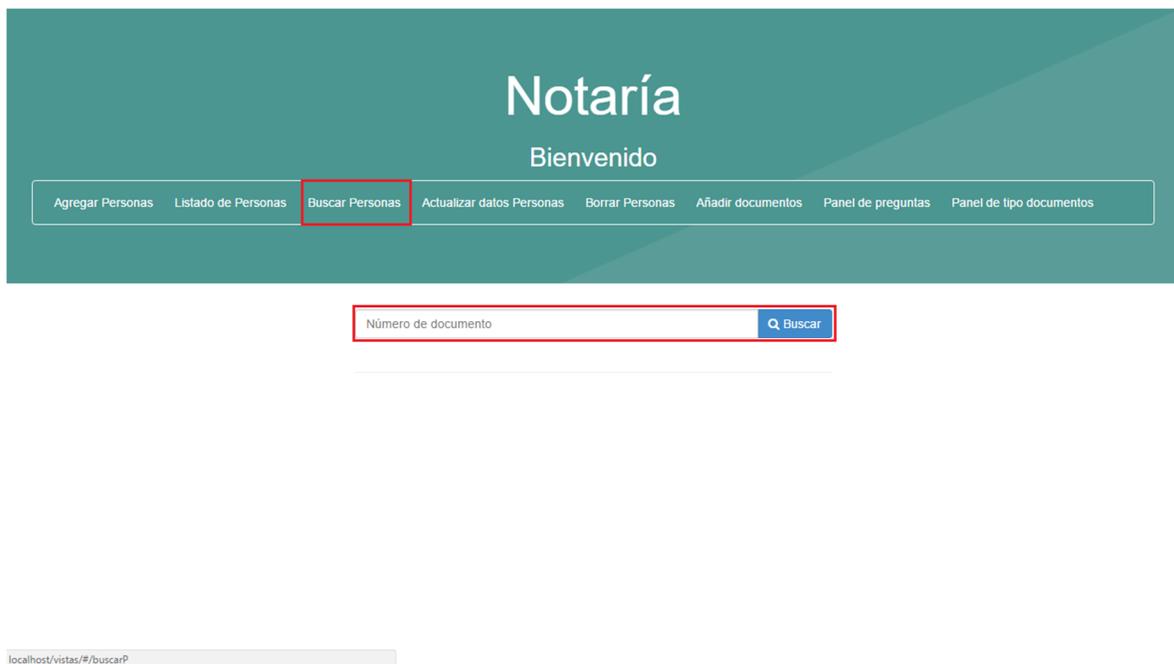


Imagen 6

Despues de hacer click en “Buscar” podemos observar la informacion del usuario solicitado, nombre, numero de identificacion y documentos asociados al usuario.



Imagen 7

- ACTUALIZAR DATOS PERSONAS

En este apartado tenemos la opción de modificar los datos de los usuarios anteriormente registrados ya sea por la existencia de algún error o por la falta de algún dato. Haciendo click en “Actualizar datos personas” se nos solicita ingresar el número de identificación de la persona a la cual se le desea modificar los datos, una vez ingresemos el número de identificación hacemos click en el botón buscar y el sistema inmediatamente nos muestra los datos del usuario y la opción de modificarlos, en este caso se hizo el cambio de Cedula a Registro civil y dando click en “Guardar” nos arroja un mensaje de éxito

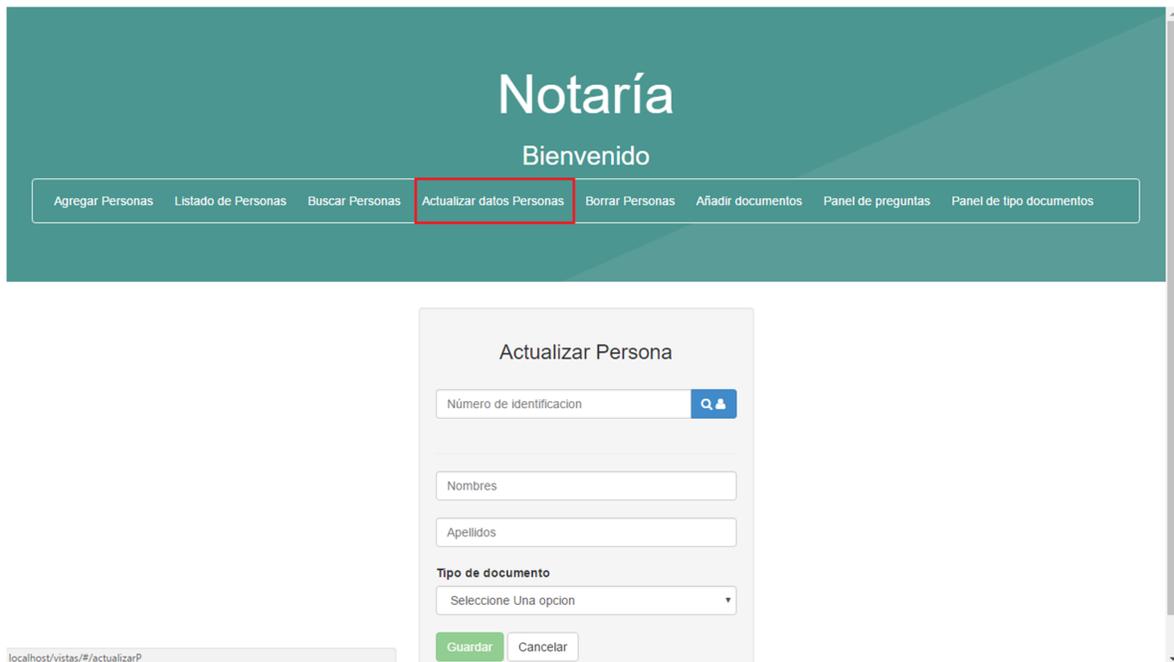


Imagen 8



Actualizar Persona

1234567890

Nombres

Apellidos

Tipo de documento
Seleccione Una opcion

Imagen 9



Actualizar Persona

1234567890

Juan Jose

Perez Polo

Tipo de documento
Registro civil

Imagen 10



Imagen 11

- BORRAR PERSONAS

Haciendo click en “Borrar Personas” tenemos la opcion de eliminar cualquier usuario de la base de datos, para esto digitamos el numero de identificacion del usuario y damos click en “Eliminar”, el sistema nos arroja un mensaje solicitandonos confirmar la eliminacion del usuario, hacemos click en “Si” y recibimos un mensaje indicandonos que el proceso de eliminacion fue exitoso.



localhost/vistas/#/borrarP

Imagen 12



1234567890

Imagen 13



1234567890

Desea eliminar los datos de la persona?

ID:1234567890

Nombres:Juan Jose

Apellidos:Perez Polo

Imagen 14

Notaría

Bienvenido

Agregar Personas Listado de Personas Buscar Personas Actualizar datos Personas Borrar Personas Añadir documentos Panel de preguntas Panel de tipo documentos

Número de documento

 Borrado
Eliminado 4x

Imagen 15

- AÑADIR DOCUMENTOS

Haciendo click en el boton “Añadir documentos” tenemos la opcion de subir documentos y relacionarlos con cualquier usuario en nuestra base de datos, para hacer esto digitamos el numero de identificacion del usuario al cual queremos registrarle un documento, el sistema inmediatamente nos muestra el nombre del usuario registrado con ese numero lo cual nos permite varificar que sea el usuario deseado, luego hacemos click en “Elegir documento” se abrira una ventana que nos permite navegar en nuestro equipo y buscar el documento a subir, una vez encontrado el documento le damos click y luego click en abrir, podemos agregar alguna descripcion si se desea y hacemos click en “Guardar” para subir el documento.

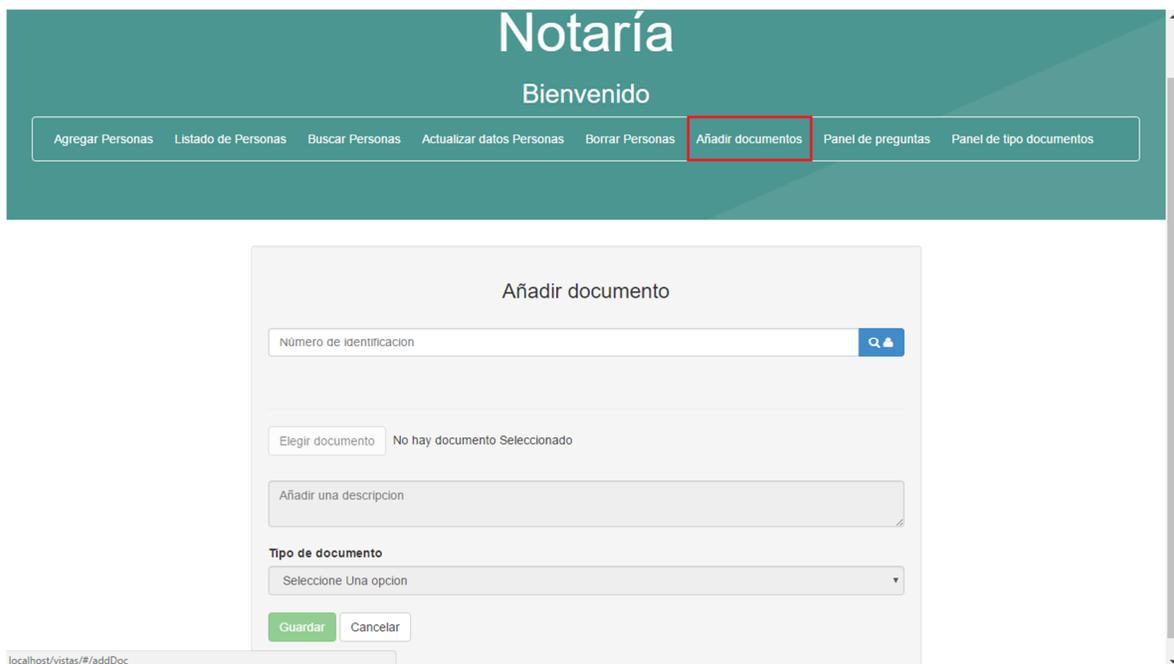


Imagen 16

Añadir documento

1082905752

Datos de la persona identificada con el numero:1082905752
Maria Fernanda Ramirez Gomez

Elegir documento No hay documento Seleccionado

Añadir una descripción

Tipo de documento
Seleccione Una opcion

Guardar Cancelar

Imagen 17

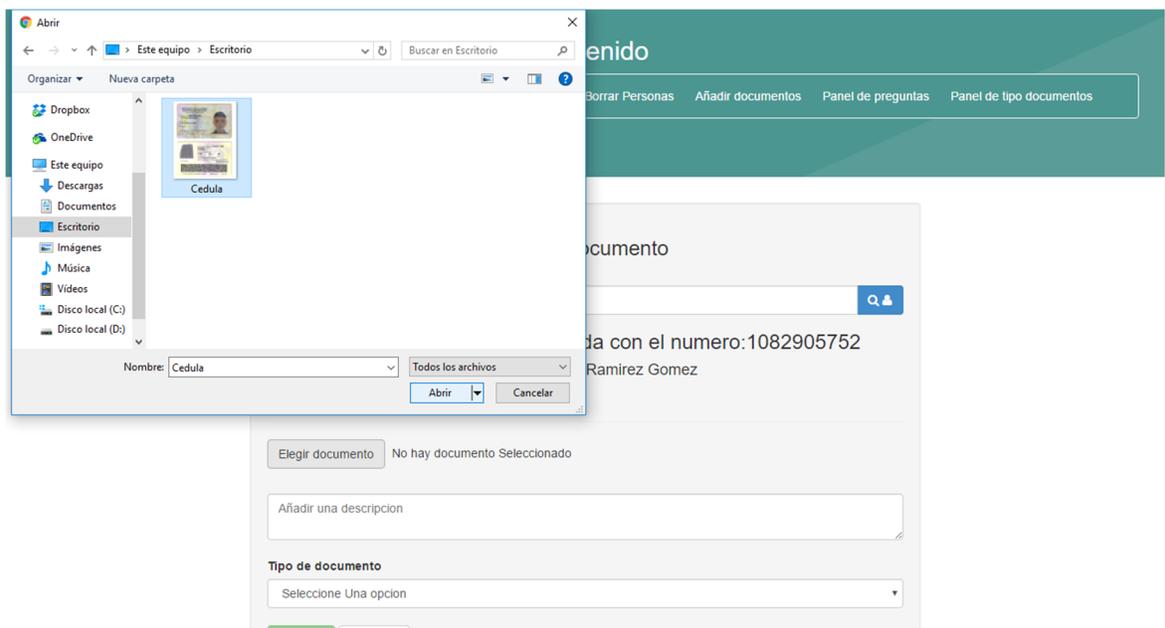


Imagen 18

- PANEL DE PREGUNTAS

En la opción “Panel de preguntas” encontramos algunas de las preguntas mas frecuentes en realizadas en una notaria publica, las cuales han sido registradas con anterioridad. Tambien contamos con las opciones:

- Añadir nueva pregunta
- Editar pregunta
- Eliminar pregunta

Notaría
Bienvenido

[Agregar Personas](#)
[Listado de Personas](#)
[Buscar Personas](#)
[Actualizar datos Personas](#)
[Borrar Personas](#)
[Añadir documentos](#)
[Panel de preguntas](#)
[Panel de tipo documentos](#)

[Añadir Nueva pregunta +](#)
[Editar Pregunta](#)
[Eliminar Pregunta](#)

Escriba una búsqueda filtrada...

ID	Preguntas	Respuestas	Palabras Claves
1	¿Que es una Capitulación Matrimonial?	Las capitulaciones matrimoniales son una conve...	Capitulación Matrimonial
2	¿Cuándo puede hacerse el documento de Capitu...	Se pueden hacer capitulaciones matrimoniales a...	Capitulación Matrimonial, Matrimonio
3	¿Cuales son los requisitos Notariales para la real...	Los únicos requisitos el acuerdo y la comparecen...	Capitulación Matrimonial
4	¿Que son los bienes mancomunados?	Quiere decir que los bienes que se tienen al mo...	bienes mancomunados
5	¿Que son los bienes separados?	Quiere decir que los bienes de cada uno de los e...	bienes separados
6	Ventajas y desventajas de casarme por bienes s...	Toda adquisición material que se haga antes y d...	bienes separados
7	¿Donde puedo dirigirme para efectuar una separ...	Puede dirigirse directamente a un notario. Tenga ...	separación de bienes
8	¿La separación de bienes impide que las deudas...	En régimen de separación de bienes las deudas ...	separación de bienes
9	¿Que es una sociedad Conyugal?	Es la unión de los bienes materiales que se obt...	Sociedad Conyugal

localhost/vistas/#/preguntas

Imagen 19

Haciendo click en el boton “Añadir pregunta” se despliega una ventana que nos permite hacer el registro de nuevas preguntas a la base de datos, en esta se solicita que se “escriba la pregunta” (1), “la respuesta” (2) y “palabras claves” (3), una vez ingresada esta informacion el sistema nos habilita el boton “Guardar pregunta” para registrar la pregunta en la base de datos.

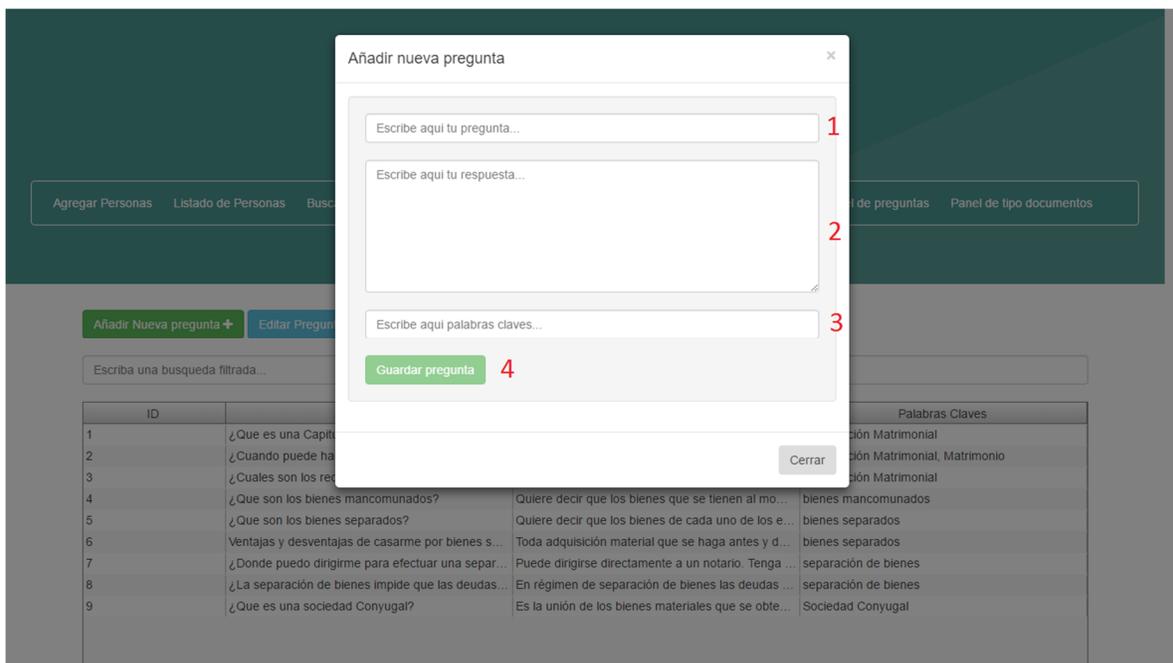


Imagen 20

Para hacer uso de la opción “Editar pregunta” debemos hacer click en la pregunta que se quiere modificar (1) y luego damos click en “Editar pregunta” (2), se despliega una ventana en la cual vemos la pregunta con su respuesta y palabras claves, realizamos los cambios correspondientes y hacemos click en “Guardar pregunta” para guardar la pregunta con sus respectivos cambios.

2

Añadir Nueva pregunta + Editar Pregunta Eliminar Pregunta

Escriba una búsqueda filtrada...

ID	Preguntas	Respuestas	Palabras Claves
1	¿Que es una Capitulación Matrimonial?	Las capitulaciones matrimoniales son una conve...	Capitulación Matrimonial
2	¿Cuándo puede hacerse el documento de Capitu...	Se pueden hacer capitulaciones matrimoniales a...	Capitulación Matrimonial, Matrimonio
3	¿Cuales son los requisitos Notariales para la real...	Los únicos requisitos el acuerdo y la comparecen...	Capitulación Matrimonial
4	¿Que son los bienes mancomunados?	Quiere decir que los bienes que se tienen al mo...	bienes mancomunados
5	¿Que son los bienes separados?	Quiere decir que los bienes de cada uno de los e...	bienes separados
6	Ventajas y desventajas de casarme por bienes s...	Toda adquisición material que se haga antes y d...	bienes separados
7	¿Donde puedo dirigirme para efectuar una separ...	Puede dirigirse directamente a un notario. Tenga ...	separación de bienes
8	¿La separación de bienes impide que las deudas...	En régimen de separación de bienes las deudas ...	separación de bienes
9	¿Que es una sociedad Conyugal?	Es la unión de los bienes materiales que se obt...	Sociedad Conyugal

1

Imagen 21

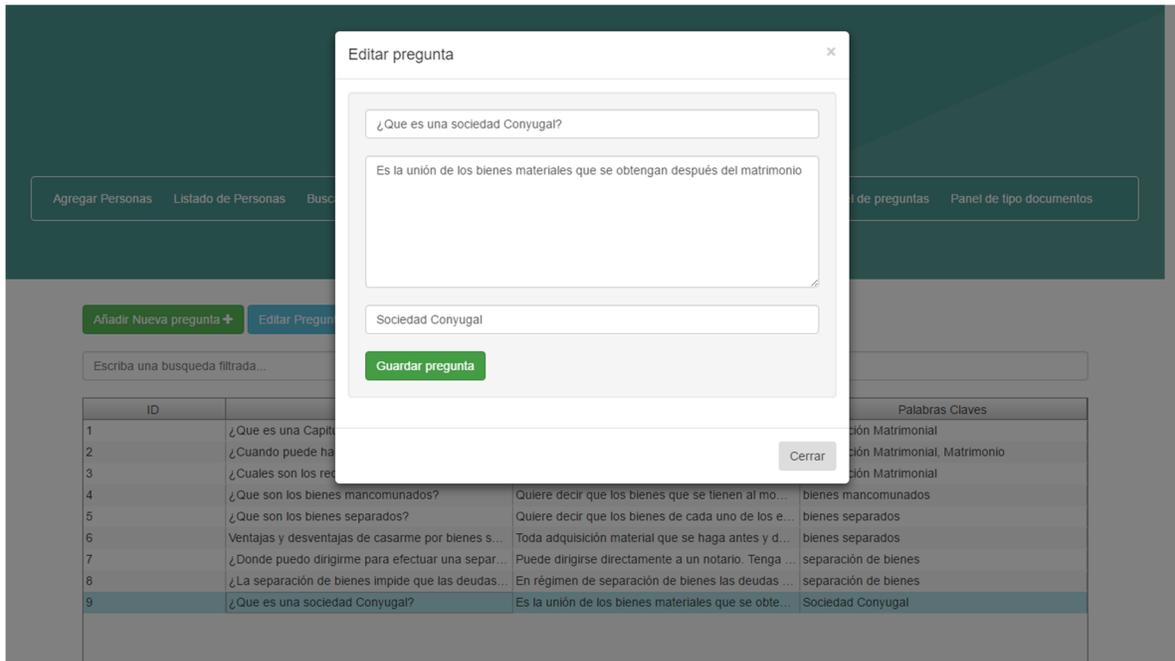


Imagen 22

Para eliminar una pregunta de la base de datos hacemos click en la pregunta que se desea eliminar (1) y luego hacemos click en “Eliminar pregunta” (2), el sistema nos arroja una ventana en la cual debemos confirmar que queremos eliminar la pregunta, esto se hace para evitar borrar alguna pregunta por error, si realmente se desea borrar la pregunta hace click en el boton “Eliminar pregunta” de la ventana de confirmacion.

2

Añadir Nueva pregunta + Editar Pregunta Eliminar Pregunta

Escriba una búsqueda filtrada...

ID	Preguntas	Respuestas	Palabras Claves
1	¿Que es una Capitulación Matrimonial?	Las capitulaciones matrimoniales son una conve...	Capitulación Matrimonial
2	¿Cuando puede hacerse el documento de Capitu...	Se pueden hacer capitulaciones matrimoniales a...	Capitulación Matrimonial, Matrimonio
3	¿Cuales son los requisitos Notariales para la real...	Los únicos requisitos el acuerdo y la comparecen...	Capitulación Matrimonial
4	¿Que son los bienes mancomunados?	Quiere decir que los bienes que se tienen al mo...	bienes mancomunados
5	¿Que son los bienes separados?	Quiere decir que los bienes de cada uno de los e...	bienes separados
6	Ventajas y desventajas de casarme por bienes s...	Toda adquisición material que se haga antes y d...	bienes separados
7	¿Donde puedo dirigirme para efectuar una separ...	Puede dirigirse directamente a un notario. Tenga ...	separación de bienes
8	¿La separación de bienes impide que las deudas...	En régimen de separación de bienes las deudas ...	separación de bienes
9	¿Que es una sociedad Conyugal?	Es la unión de los bienes materiales que se obt...	Sociedad Conyugal

1

Imagen 23

Eliminar pregunta

¿Seguro que desea eliminar esta pregunta?

Cerrar Eliminar pregunta

Agregar Personas Listado de Personas Buscador de preguntas Panel de tipo documentos

Añadir Nueva pregunta + Editar Pregunta Eliminar Pregunta

Escriba una búsqueda filtrada...

ID	Preguntas	Respuestas	Palabras Claves
1	¿Que es una Capitulación Matrimonial?	Las capitulaciones matrimoniales son una conve...	Capitulación Matrimonial
2	¿Cuando puede hacerse el documento de Capitu...	Se pueden hacer capitulaciones matrimoniales a...	Capitulación Matrimonial, Matrimonio
3	¿Cuales son los requisitos Notariales para la real...	Los únicos requisitos el acuerdo y la comparecen...	Capitulación Matrimonial
4	¿Que son los bienes mancomunados?	Quiere decir que los bienes que se tienen al mo...	bienes mancomunados
5	¿Que son los bienes separados?	Quiere decir que los bienes de cada uno de los e...	bienes separados
6	Ventajas y desventajas de casarme por bienes s...	Toda adquisición material que se haga antes y d...	bienes separados
7	¿Donde puedo dirigirme para efectuar una separ...	Puede dirigirse directamente a un notario. Tenga ...	separación de bienes
8	¿La separación de bienes impide que las deudas...	En régimen de separación de bienes las deudas ...	separación de bienes
9	¿Que es una sociedad Conyugal?	Es la unión de los bienes materiales que se obt...	Sociedad Conyugal

Imagen 24

- PANEL DE TIPO DOCUMENTOS

En el apartado “Panel de tipo documentos” podemos registrar los tramites que se realizan en la notaria y sus respectivos valores, tambien podemos modificar la informacion de los tramites o borrar dicha informacion. Para agregar un tramite hacemos click en “Añadir un nuevo tipo de documento” (1) se despliega una ventana en la cual debemos ingresar el nombre del tramite o documento y su valor, hacemos click en “Guardar pregunta” y el tramite queda registrado y relacionado con su respectivo valor.

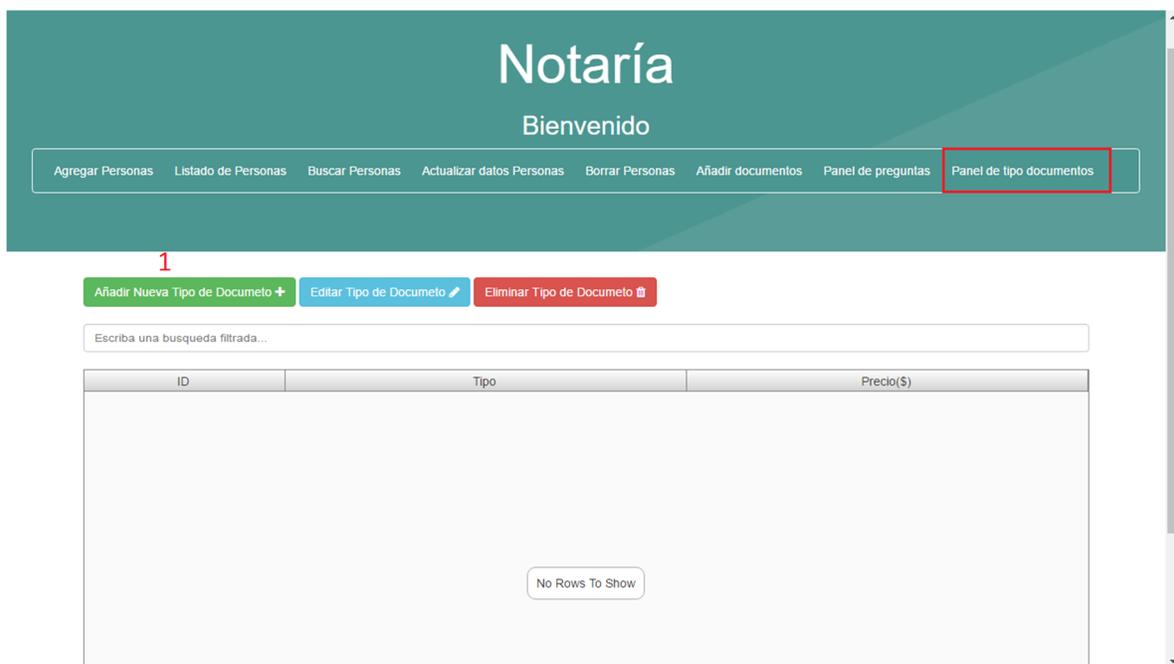


Imagen 25

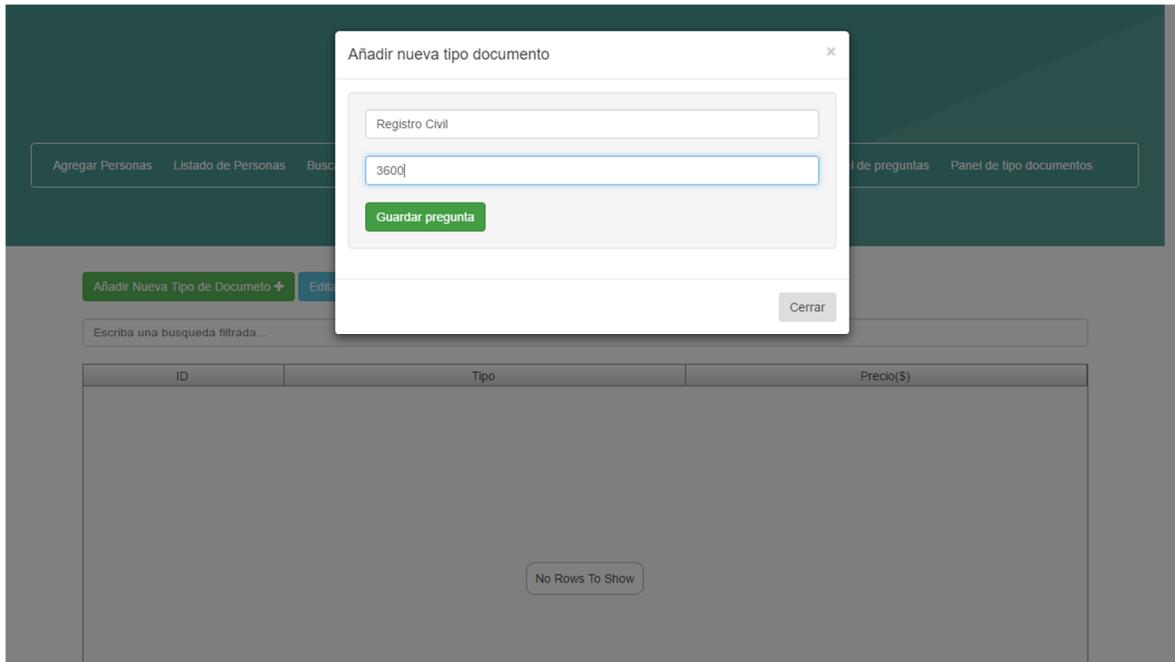


Imagen 26



Imagen 27

Para editar la información existente de un trámite o documento debemos hacer click en la opción “Editar tipo de documento” (1), se despliega una ventana emergente en la cual podemos realizar las modificaciones deseadas, ya sea nombre del trámite y/o documento o su valor, una vez realizados los cambios damos click en “Guardar pregunta” y con esto los cambios se verán efectuados.



Imagen 28

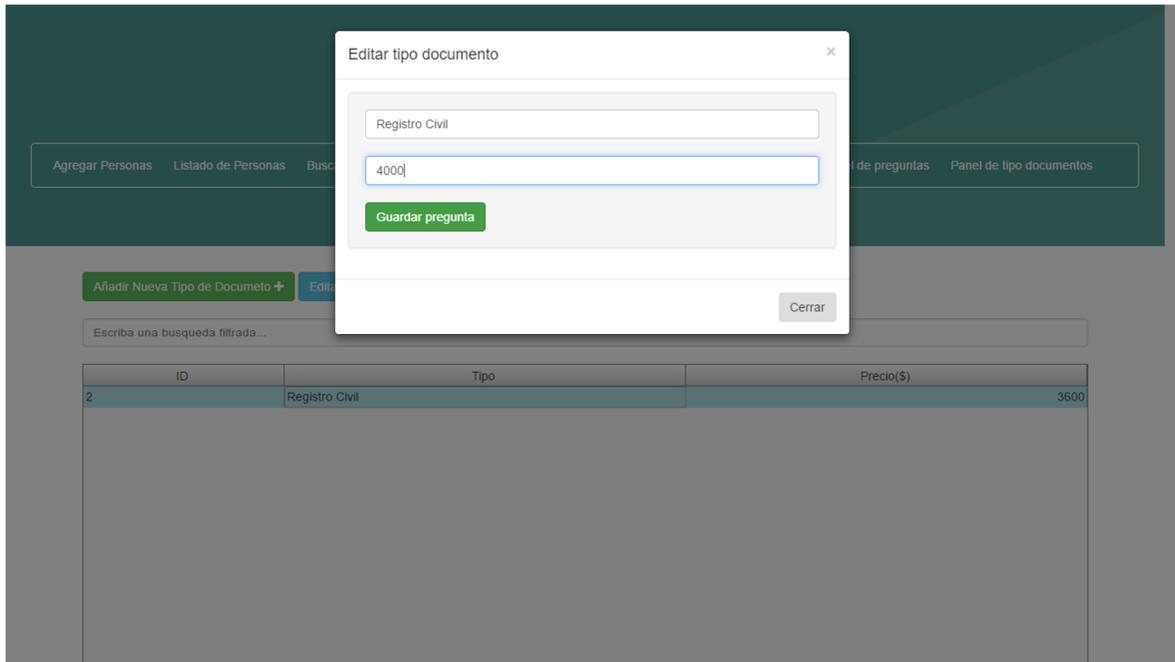


Imagen 29



Imagen 30

Para eliminar un documento debemos seleccionar el documento a eliminar y hacer click en “Eliminar tipo de documento” (1), se despliega una ventana de confirmacion en la cual debemos confirmar que queremos eliminar el documento dando click en “Eliminar pregunta” y con esto el documento sera eliminado.



Imagen 31

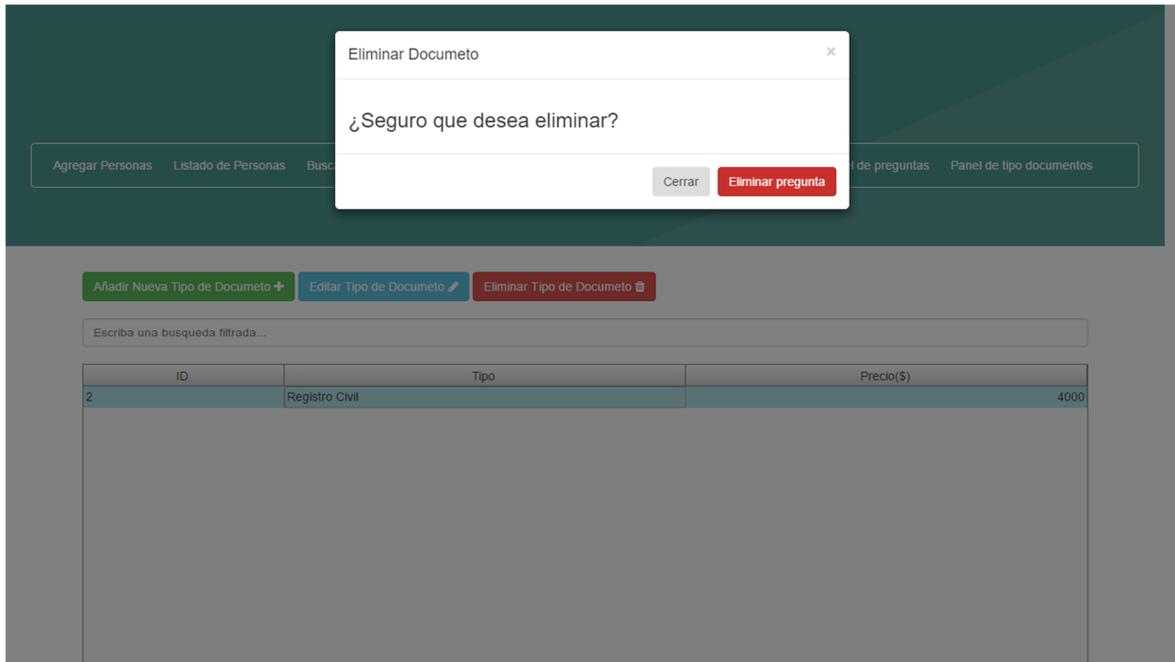


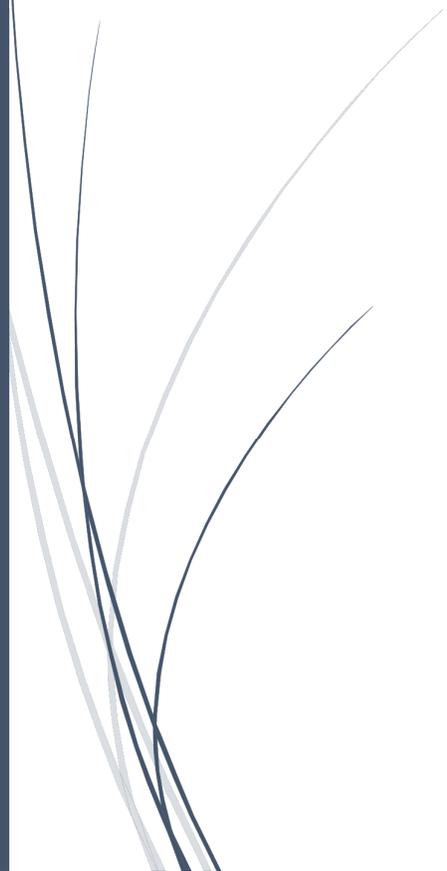
Imagen 32



Imagen 33



Manual Técnico



Holman Ortega;Javier Castañeda

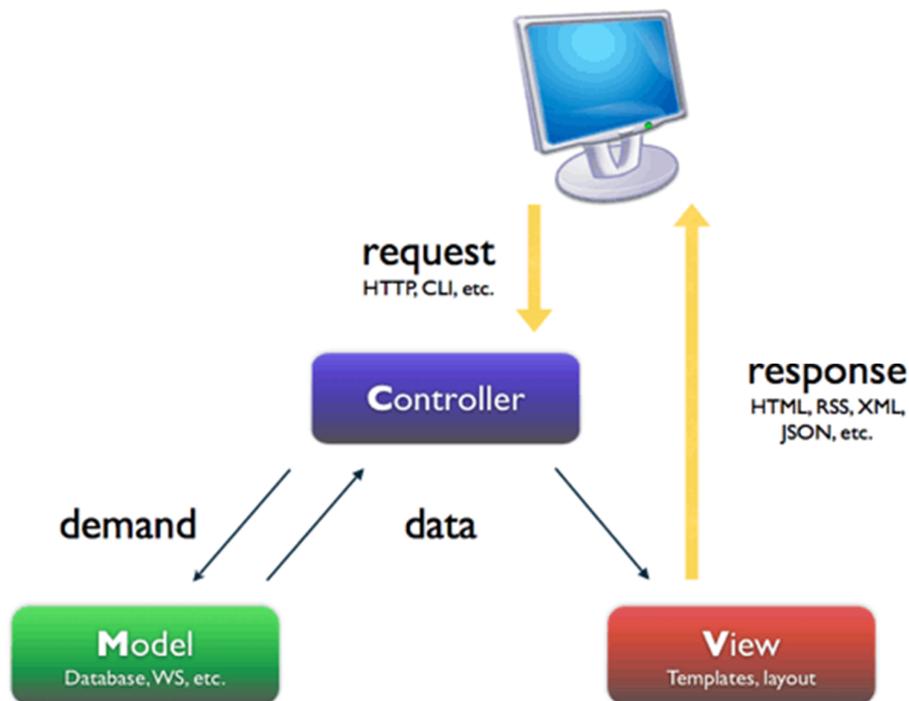
Tabla de Contenidos

1. Arquitectura del sistema	1
2. Descripción del patrón	2
2.1 El Modelo	2
2.2 El Controlador	2
2.3 La Vista	2
3. Casos de uso	3
3.1 Actores del sistema	3
3.1.1 Administrador/Notario	3
3.1.2 Usuarios consultantes	4
3.2 Casos de uso	4
3.3 Descripción de casos de uso	7
4. Modelo de datos	23
4.1 Modelo entidad relación	23
4.2 Diccionario de datos	24
5. Instalación y configuración del Servidor Local	28

1. Arquitectura del sistema

ARQUITECTURA MVC(Modelo-Vista-Controlador): El sistema se desarrolló bajo la arquitectura MVC basándose en los requerimientos funcionales y no funcionales, con el fin de separar la lógica de negocio tanto de los datos como de la vista de los usuarios, dada la flexibilidad para el acceso. Teniendo como finalidad la generación de un sistema enfocado interoperabilidad, escalabilidad y la posibilidad de tener aplicaciones distribuidas en varias ubicaciones.

En la siguiente figura se puede observar cómo está conformada la arquitectura con la se construyó la aplicación.



2. Descripción del patrón

- 2.1. El Modelo: Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la 'vista' aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al 'modelo' a través del 'controlador'.

- 2.2. El Controlador: Responde a eventos (usualmente acciones del usuario) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a su 'vista' asociada si se solicita un cambio en la forma en que se presenta el 'modelo' (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos), por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo' .

- 2.3. La Vista: Presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario) por tanto requiere de dicho 'modelo' la información que debe representar como salida

3. Casos de uso

3.1. Actores del sistema: el sistema de información solo contará con dos actores con los cuales interactúa, se describe a continuación sus perfiles y actividades a desempeñar.

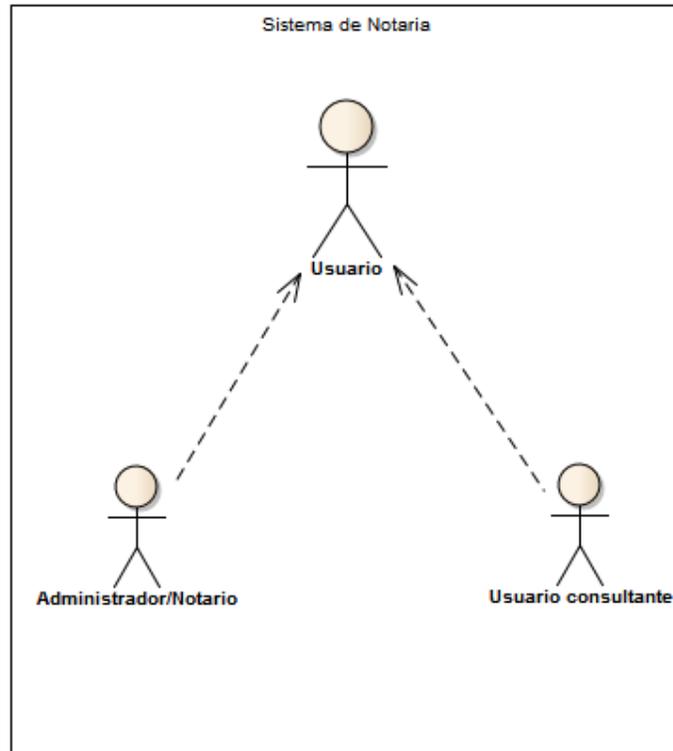


Ilustración 1 Actores del sistema

3.1.1. Administrador/Notario: Es quien tiene el control total del sistema, encargado de controlar toda la información contenida en él, además es el único que puede Agregar, modificar y eliminar la información contenida en la base de datos.

3.1.2. Usuarios consultantes: Tienen el acceso a consultar la información correspondiente a ellos con su número de identificación, además de escogerla y enviarla a impresión, este actor puede solicitar al Administrador cargar información de el al sistema.

Por otra parte este actor puede realizar preguntas al sistema sobre la documentación requerida.

3.2. Casos de uso: a continuación se ilustran las funciones de cada uno de los actores del sistema.

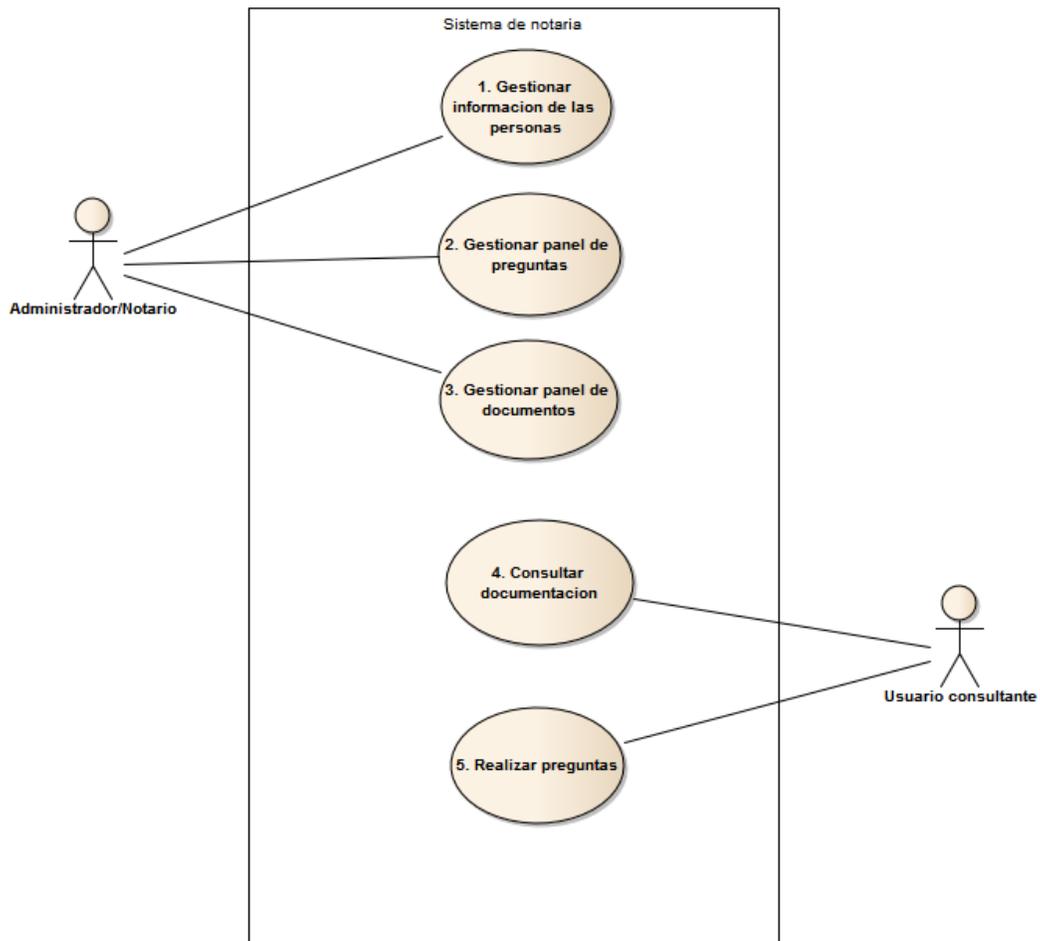


Ilustración 2 Administración y consulta del sistema (caso de uso de primer nivel)

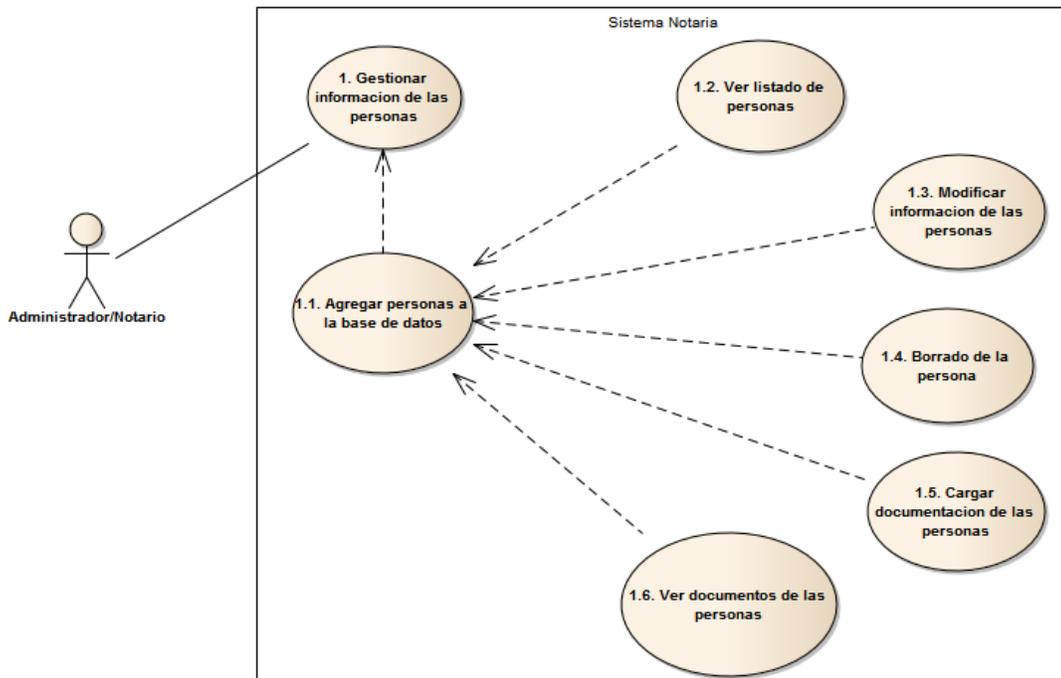


Ilustración 3 Gestión de la información de las personas

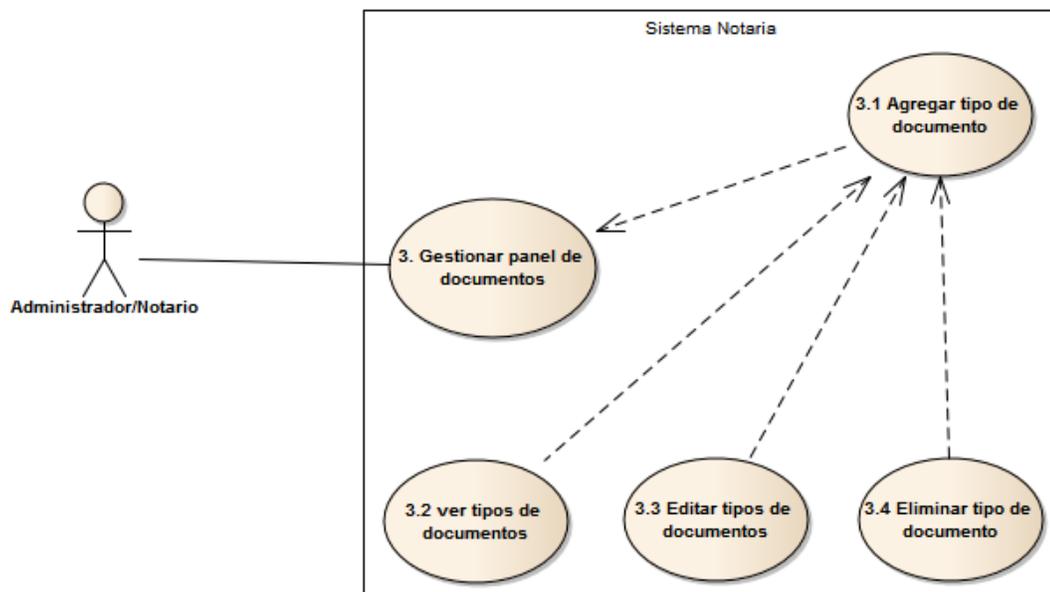


Ilustración 4 Gestión de la documentación de las personas

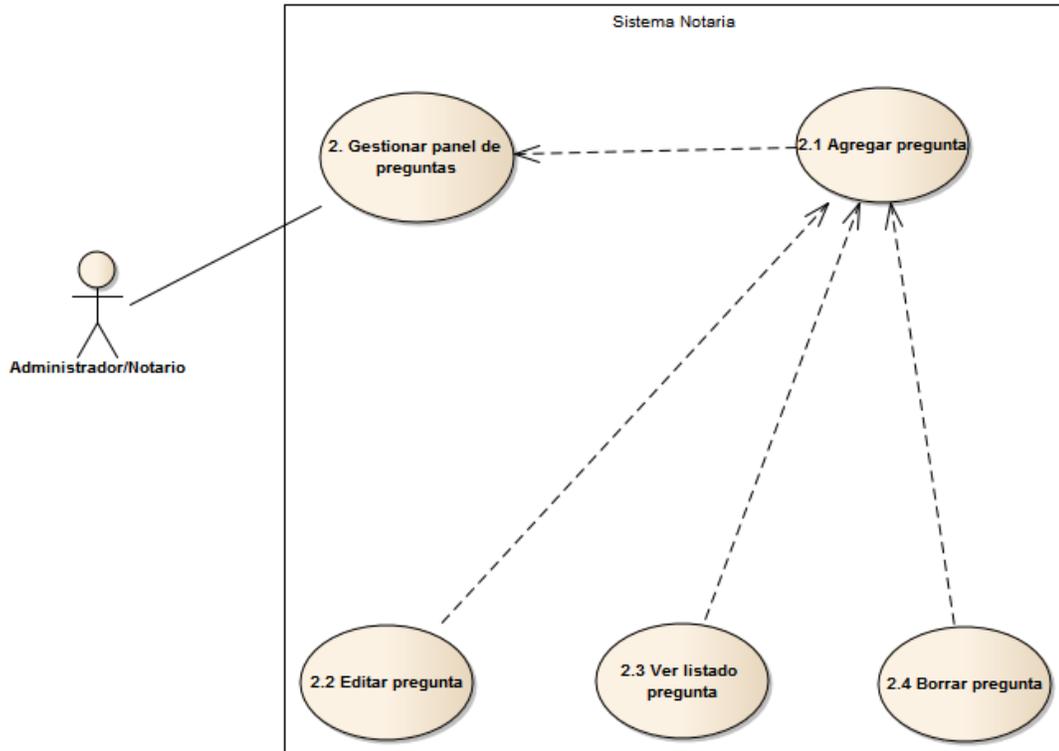


Ilustración 7 Gestión del panel de preguntas

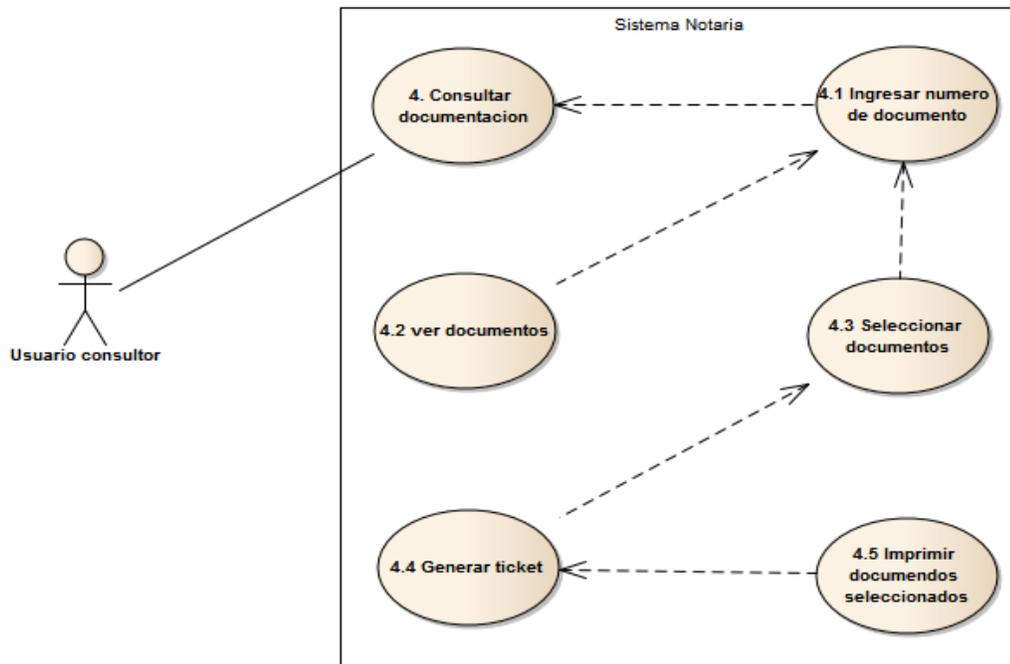


Ilustración 5 Consulta de la documentación

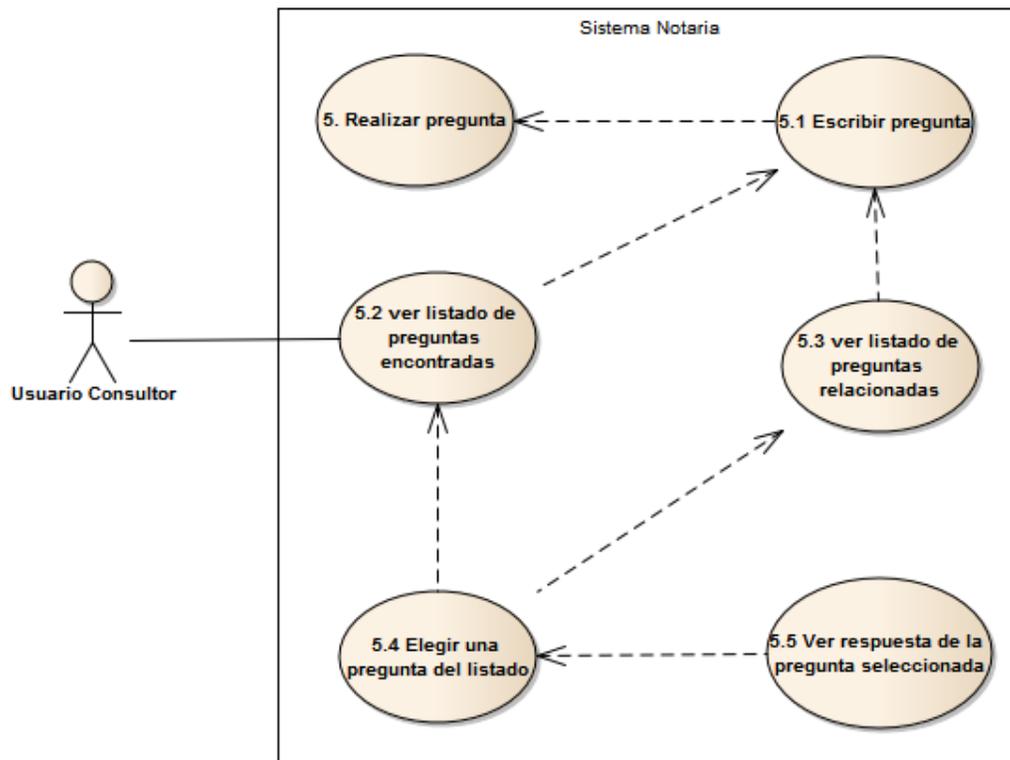


Ilustración 8 Consulta de preguntas

3.3. Descripción de casos de uso

Agregar personas a la base de dato

El notario o administrador ingresa, actualiza, consulta y elimina la información de las personas.

Restricciones

- Pre-condición
Ser el administrador general del sistema.

- Post-condición

La información de la persona quedara registrada en el sistema

Escenarios

Escenario normal

1. El sistema despliega una pantalla con el formulario de información de la persona, para el administrador digite la información de la persona en cuestión
2. El notario debe completar el formulario con la siguiente información
 - identificación de la persona
 - Nombres
 - Apellidos
 - Tipo de identificación
 - a. Si al escribir el número de identificación de la persona, este existe no se podrá hacer el registro de un dato repetido.
3. El sistema valida q la información sea correcta.
4. El notario elige guardar la información en el sistema
5. El sistema almacena los datos y muestra un mensaje indicando que se guardó correctamente la información.

Escenario alternativo

4. El sistema no dejará realizar el guardado de la información si este no tiene los campos requeridos completos.
 - a. Volver al paso 2.

Ver listado de personas

El notario puede ver el listado de las personas registradas en el sistema.

Restricciones

- Pre-condición
Ser el administrador general del sistema.

Escenarios

Escenario normal

1. El sistema despliega el listado de las personas registradas en el sistemas
2. El notario podrá buscar en el listado, mediante el número de identificación de la persona
3. El sistema le arrojará una fila con la información de la persona buscando.

Escenario alternativo

3. Si el número de identificación de la persona no existe, la búsqueda aparecerá vacía
 - a. Volver al paso 2.

Buscar personas

El notario escribe el número de identificación de la persona y pueda observar la información de este.

Restricciones

- Pre-condición
 - Ser el administrador general del sistema.
 - Que exista el registro en la base de datos
- Post-condición
 - Se podrá observar la información de las personas

Escenarios

Escenario normal

1. El notario digita el número de identificación de la persona en cuestión
2. El sistema busca a la información y documentos de las personas

3. El sistema despliega en pantalla la información de la persona y su documentación
4. El notario podrá ver los documentos de la persona

Escenario alternativo

3. Si no se encuentra la información de la persona el sistema arrojará un mensaje diciendo que no hay información encontrada
 - a. Volver al paso 2.

Actualizar datos de personas

El notario podrá actualizar los datos de una persona, siempre y cuando este registrado en el sistema.

Restricciones

- Pre-condición
 - Ser el administrador general del sistema.
 - Que exista el registro en la base de datos

- Post-condición
 - Se podrá observar la información de las personas
 - Se podrá actualizar la información de las personas

Escenarios

Escenario normal

1. El sistema muestra un formulario de actualización de información de personas, bastante similar al de registro.
2. El notario ingresa el número de identificación de la persona
3. Se carga la información de la persona de forma automática en los campos.
4. Se modifican los campos que se deseen, y se guarda la información actualizada.
5. El sistema arroja una notificación informando que ha guardado la información

Escenario alternativo

3. Si no se encuentra información de la persona, los campos quedan en blanco.
 - a. Volver al paso 2.

Borrar información de personas.

Se borra total y completamente la información de la persona y no queda registro alguno en el sistema.

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se podrá observar la información de las personas
Se podrá eliminar la información de las personas

Escenarios

Escenario normal

1. El sistema muestra un campo donde se debe digitar el número de identificación de la persona.
2. El notario digita el número de identificación de la persona.
3. El sistema arroja la información de la persona, además de un botón de confirmación de eliminación
4. Después de presionar la confirmación el sistema elimina dicho registro.

Escenario alternativo

3. El sistema no encuentra la información de la persona y arroja un mensaje diciendo q no se encontraron registros de ese número de identificación.
 - a. Volver al paso 2.

Escenario alternativo 2.

4. Si se presiona el botón no, se borra la información mostrada en pantalla, además de la información digitada en el campo.

Cargar documentos persona

Se almacena la documentación de la persona en el sistema para su posterior consulta.

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se cargara la documentación de cada persona.

Escenarios

Escenario normal

1. El sistema muestra una pantalla con un formulario, el cual se debe completar con la información allí requerida
2. Se completa el formulario con la siguiente información
 - Documento de la persona
 - fotografía del documento

-Descripción del documento

-Tipo de documento

3. Después de haber completado todos los campos, el notario guarda la información
4. El sistema almacena la información y arroja una notificación donde se informa que los datos fueron almacenados correctamente

Escenario alternativo

3. Si no se encuentra información el sistema muestra un mensaje diciendo q no se ha encontrado la información
 - a. Volver al paso 2.

Añadir preguntas

El sistema guarda las nuevas preguntas que el administrado desee agregar

Restricciones

- Pre-condición
Ser el administrador general del sistema.
- Post-condición
Se guardara la pregunta.

Escenarios

Escenario normal

1. El sistema arroja una venta flotante con el formulario correspondiente para agregar las preguntas, el cual se deberá completar para realizar la inserción de la pregunta en el sistema
2. Cuando el notario termine el formulario de edición se deberá guardar la información
3. Al ser editado el sistema arrojará una notificación informando que se guardó correctamente la pregunta

Editar preguntas

El sistema permite editar las preguntas anteriormente guardadas

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se guarda la edición de la pregunta

Escenarios

Escenario normal

1. Se elige la pregunta a editar, se presiona el botón de edición.
2. El sistema arroja una ventana flotante con el formulario correspondiente para agregar las preguntas, el cual se deberá completar para realizar la edición de la pregunta en el sistema
3. Cuando el notario termine el formulario se deberá guardar la información
4. Al ser guardado el sistema arrojará una notificación informando que se guardó correctamente la pregunta

Listar preguntas

El sistema muestra el listado de las preguntas con sus respuestas

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se cargara la lista de preguntas.

Escenarios

Escenario normal

1. El sistema arroja el listado de todas las preguntas guardadas en el sistema

Borrar preguntas

El sistema muestra borra la pregunta seleccionada

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se borrara la lista de preguntas.

Escenarios

Escenario normal

1. Se elige la pregunta a borrar, se presiona el botón de borrado.

2. El sistema arroja una venta flotante con un mensaje de confirmación de eliminación
3. Después de presionar el botón de confirmación se elimina la pregunta del sistema

Añadir tipo de documento

El sistema guarda los nuevos tipos de documentos.

Restricciones

- Pre-condición
Ser el administrador general del sistema.
- Post-condición
Se guardara los tipos de documentos.

Escenarios

Escenario normal

1. El sistema arroja una venta flotante con el formulario correspondiente para llenar el tipo de documento, el cual se cargara completado con la información para realizar la carga en el sistema
2. Cuando el notario termine el formulario de se deberá guardar la información

3. Al ser editado el sistema arrojará una notificación informando que se guardó correctamente el tipo de documento

Editar tipo de documento

El sistema permite editar los tipos de documentos anteriormente guardados

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se guarda la edición los tipos de documentos.

Escenarios

Escenario normal

1. Se elige el tipo de documento a editar, se presiona el botón de edición.
2. El sistema arroja una ventana flotante con el formulario correspondiente para editar el tipo de documento.
3. Cuando el notario termine el formulario se deberá guardar la información

4. Al ser guardado el sistema arrojará una notificación informando que se guardó correctamente.

Listar tipo de documento

El sistema muestra el listado de los tipos de documentos

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se cargará la lista de tipos de documentos.

Escenarios

Escenario normal

1. El sistema arroja el listado de todos los tipos de documentos guardados en el sistema

Borrar los tipos de documentos

El sistema muestra elimina los tipos de documentos seleccionados

Restricciones

- Pre-condición
Ser el administrador general del sistema.
Que exista el registro en la base de datos
- Post-condición
Se borrar la lista de preguntas.

Escenarios

Escenario normal

1. Se elige la pregunta a borrar, se presiona el botón de borrado.
2. El sistema arroja una ventana flotante con un mensaje de confirmación de eliminación
3. Después de presionar el botón de confirmación se elimina la pregunta del sistema

Buscar documento de personas

El sistema muestra la información de la persona que busca sus datos.

Restricciones

- Pre-condición
Estar registrado en el sistema
- Post-condición
Se Mostrara la información de la persona.

Escenarios

Escenario normal

1. El sistema muestra una pantalla con un campo, el cual se debe completar con la identificación de la persona
2. Después de haber completado todos los campos, el usuario realiza la búsqueda
3. El sistema muestra una pantalla con la información requerida por el usuario, documento, el tipo, y el precio de este, además podrá ver el documento antes de elegirlo
4. El usuario elige los documentos a imprimir y el sistema muestra el precio total de los documentos
5. se envían a imprimir los documentos elegidos por el usuario.

Escenario alternativo

3. Si no se encuentra información el sistema no carga ninguna información
 - a. Volver al paso 2.

Realizar preguntas frecuentes

El sistema muestra la respuesta la pregunta realizada por el usuario

Restricciones

- Post-condición

Se Mostrara la respuesta a la pregunta realizada por el usuario.

Escenarios

Escenario normal

1. El sistema muestra una pantalla con un campo, el cual se debe completar con la pregunta realizada por el usuario.
2. Después de haber completado todos los campos, el usuario realiza la búsqueda
3. El sistema muestra una pantalla con la las preguntas que se asemejan a la que el usuario realizo, además muestra unas preguntas con palabras claves encontradas en la pregunta en cuestión.
4. El usuario elige la pregunta del listado que le aparece
5. El sistema muestra una pantalla con la pregunta elegida por el usuario y una respuesta.

Escenario alternativo

3. Si no se encuentra información el sistema el sistema arroja un mensaje donde dice que no se ha encontrado la información
 - a. Volver al paso 2.

4. Modelo de datos
 - 4.1. Modelo entidad relación: A continuación se representan las entidades primordiales para el sistema de información, denotando las relaciones existentes entre las mismas.

Se utilizó MySQL Workbench el cual es una herramienta para el modelado de base de datos, y donde se construyó el modelo E-R del sistema.

En el diccionario de datos se detallan todas las tablas que conforman la base de datos, así como los atributos de cada una de ellas.

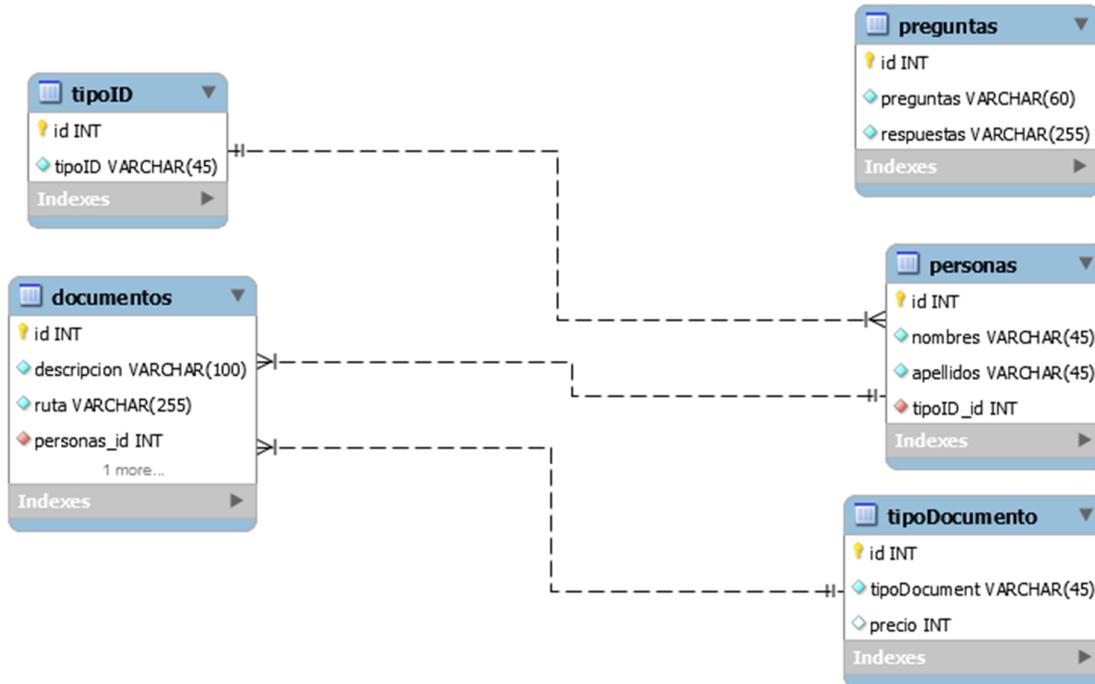


Ilustración 9 Modelo de base de datos

4.2. Diccionario de datos:

Luego de la representación gráfica del modelo entidad relación, se hace necesario tener una descripción más detallada de su estructura.

Se pretende ofrecer precisión sobre los datos que maneja el sistema, detallando sus características lógicas, complementando así, los diagramas de flujo.

Tabla TipoID

Acciones
Almacena el tipo de identificación con el cual será registrado un usuario(definido por ID)

Campo	Tipo de dato	IsNul l	P K	F K	Descripción
IdGuia	Int	No	Si	N o	Llave primaria
tipodID	VarChar (45)	No	No	N o	Guarda el nombre del tipo de ID

Tabla personas

Acciones					
Almacena la información general de cada persona, luego de ejecutarse las acciones de almacenamiento, la persona queda registrada en el sistema					
Campo	Tipo de dato	IsNull	PK	FK	Descripción
Id	Int	No	Si	No	Llave primaria, además de ser el número de identificación de cada persona registrada en el sistema
tipoID_id	int	No	No	Si	Llave foránea, que donde se registra el tipo de id con el que fue almacenada la persona
nombres	VarChar (45)	No	No	No	Guarda el nombre de cada una de las personas
apellidos	VarChar (45)	No	No	No	Guarda los apellidos de cada una de las personas

Tabla Tipo Documento

Acciones
Almacena la información la información general de cada tipo de documento que puede tener en físico la notaría

Campo	Tipo de dato	IsNul l	P K	F K	Descripción
Id	Int	No	Si	N o	Llave primaria
tipoDocumento	VarChar(45)	No	No	N o	Llave foránea, que donde se registra el tipo de id con el que fue almacenada la persona
Precio	int	No	No	N o	Guarda el precio de impresión que tendrá el documento

Tabla Documentos

Acciones					
Almacena cada uno de los documentos que pertenecen a las personas registradas en el sistema					
Campo	Tipo de dato	IsNull	P	F	Descripción
		l	K	K	
Id	Int	No	Si	No	Llave primaria
personas_id	int	No	No	Si	Llave foránea, hace referencia a que persona pertenece el documento guardado
tipodDocumento_id	int	No	No	Si	Llave foránea, hace referencia al tipo de documento que se ha almacenado
descripcion	VarChar (255)	no	No	No	Guarda una breve descripción del documento

Ruta	VarChar (45)	no	No	o	Guarda la ruta del servidor donde se ha almacenado dicho documento
------	--------------	----	----	---	--

Tabla Preguntas

Acciones
Almacena cada una de las preguntas frecuentes con su respectiva respuesta.

Campo	Tipo de dato	IsNull	P	F	Descripción
		I	K	K	
Id	Int	No	Si	No	Llave primaria
preguntas	VarChar(60)	No	No	No	Guarda la pregunta
respuestas	VarChar(255)	No	No	No	Contiene en detalle la respuesta a dicha pregunta

5. Instalación y configuración del Servidor Local (Apache, MySQL, Phpmyadmin) XAMPP.

Descarga XAMPP

Descargamos XAMPP de su página oficial:

<https://www.apachefriends.org/es/index.html> y seleccionamos en descargar “XAMPP para Windows” como se muestra a continuación:



Ilustración 10 descarga del servidor

Una vez este descargado lo ejecutamos y primero nos aparece si está activado el Control de Cuentas de Usuario y recuerda que algunos directorios tienen permisos restringidos:

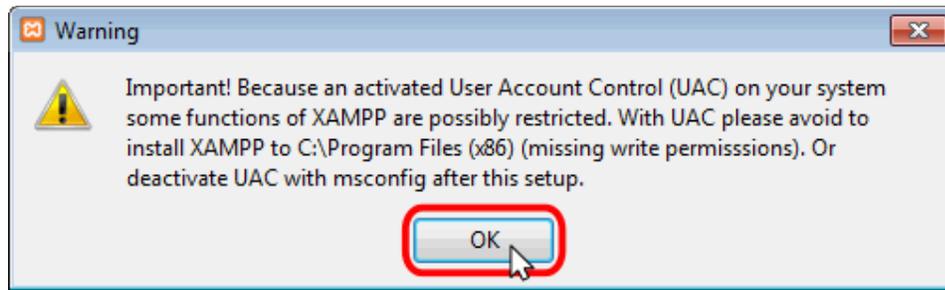


Ilustración 11 pasos de instalación

A continuación se inicia el asistente de instalación. Para continuar, hay que hacer clic en el botón "Next":

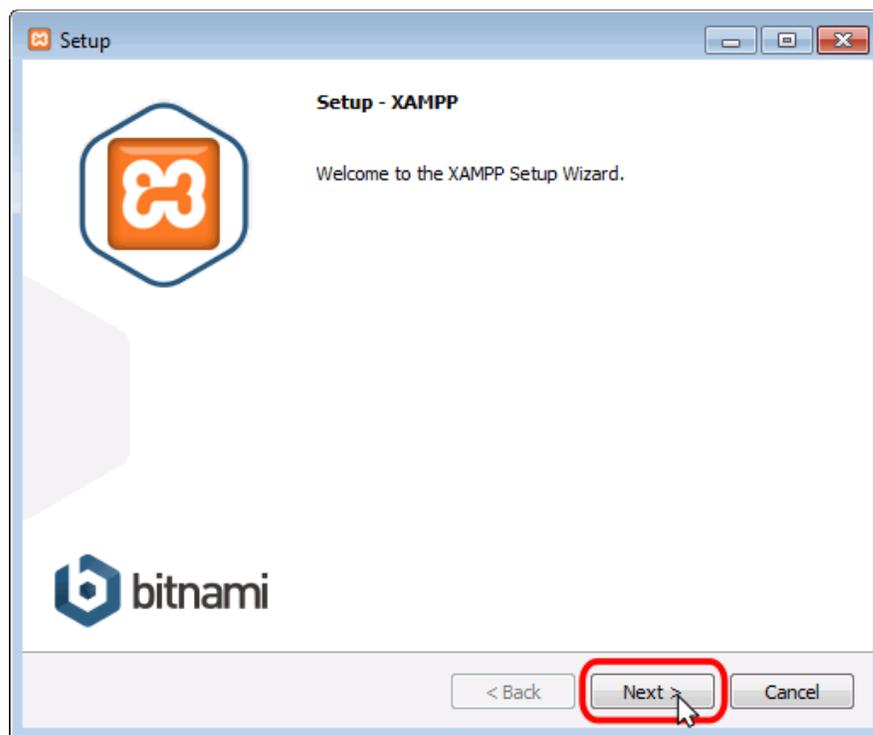


Ilustración 12 pasos de instalación

Los componentes mínimos que instala XAMPP son el servidor Apache y el lenguaje PHP, pero XAMPP también instala otros elementos. En la pantalla de selección de componentes puede elegirse la instalación o no de estos componentes. Para este curso se necesita al menos instalar MySQL y phpMyAdmin:

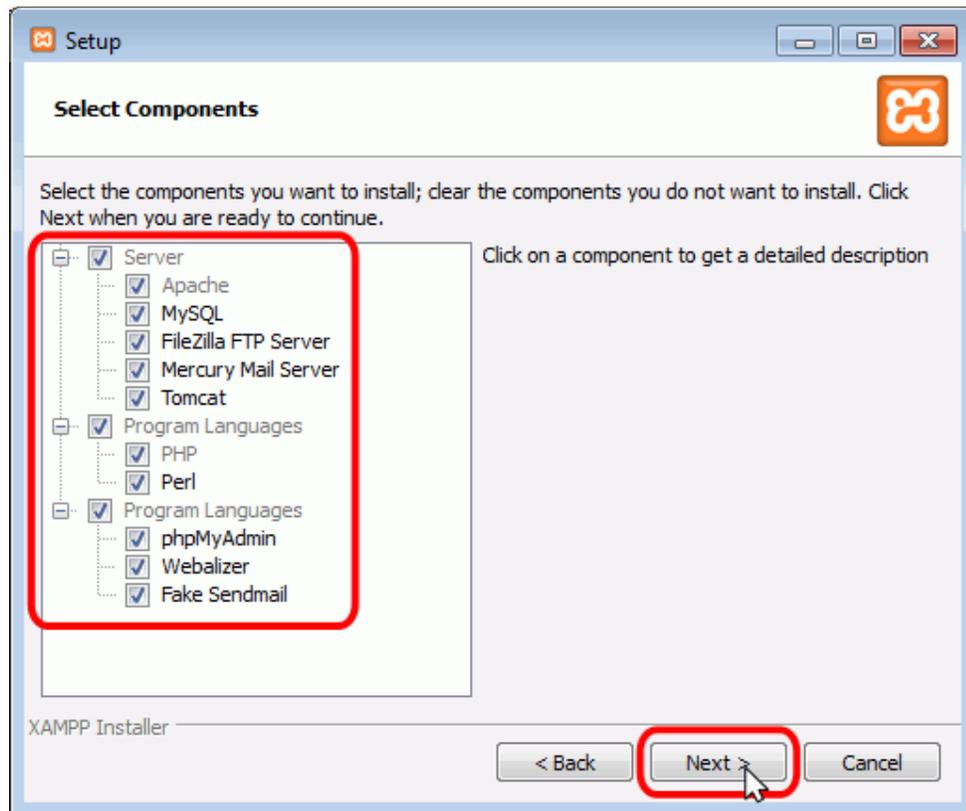


Ilustración 13 pasos de instalación

En la siguiente pantalla se puede elegir la carpeta de instalación de XAMPP. La carpeta de instalación predeterminada es **C:\xampp**. Si se quiere cambiar, hay que hacer clic en el icono de carpeta y seleccionar la carpeta donde se quiere instalar XAMPP. Para continuar la configuración de la instalación, hay que hacer clic en el botón "Next":

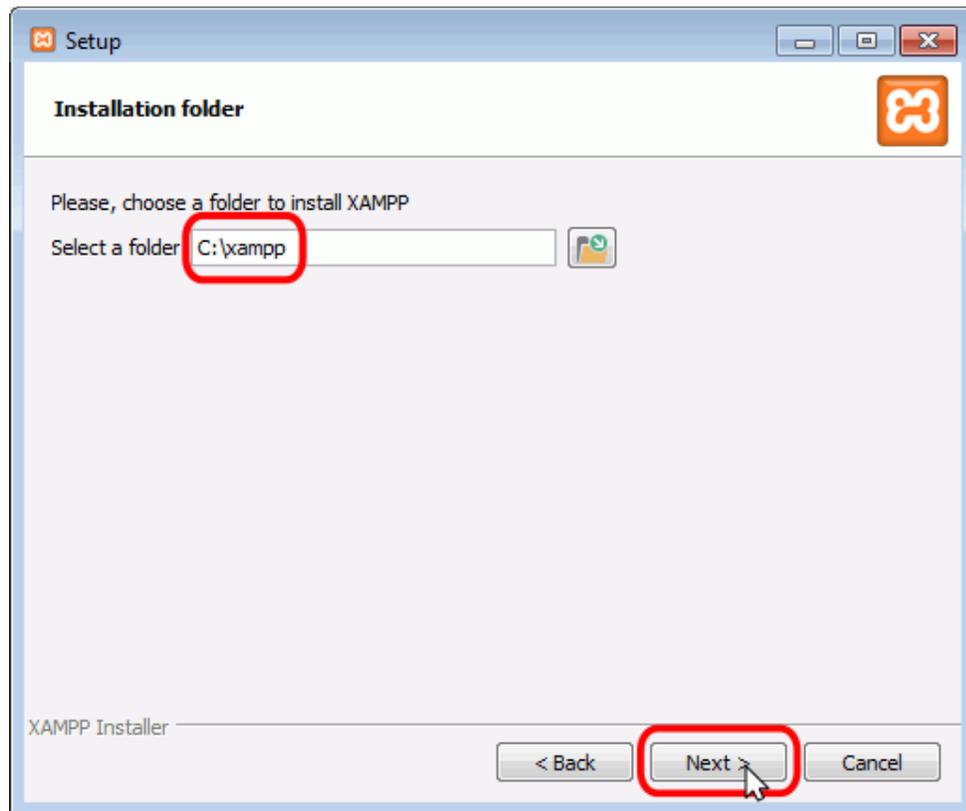


Ilustración 14 pasos de instalación

La siguiente pantalla nos ofrece información sobre los instaladores de aplicaciones para XAMPP creados por Bitnami. Para que no se abra la página web de Bitnami, habría que desmarcar la casilla correspondiente:

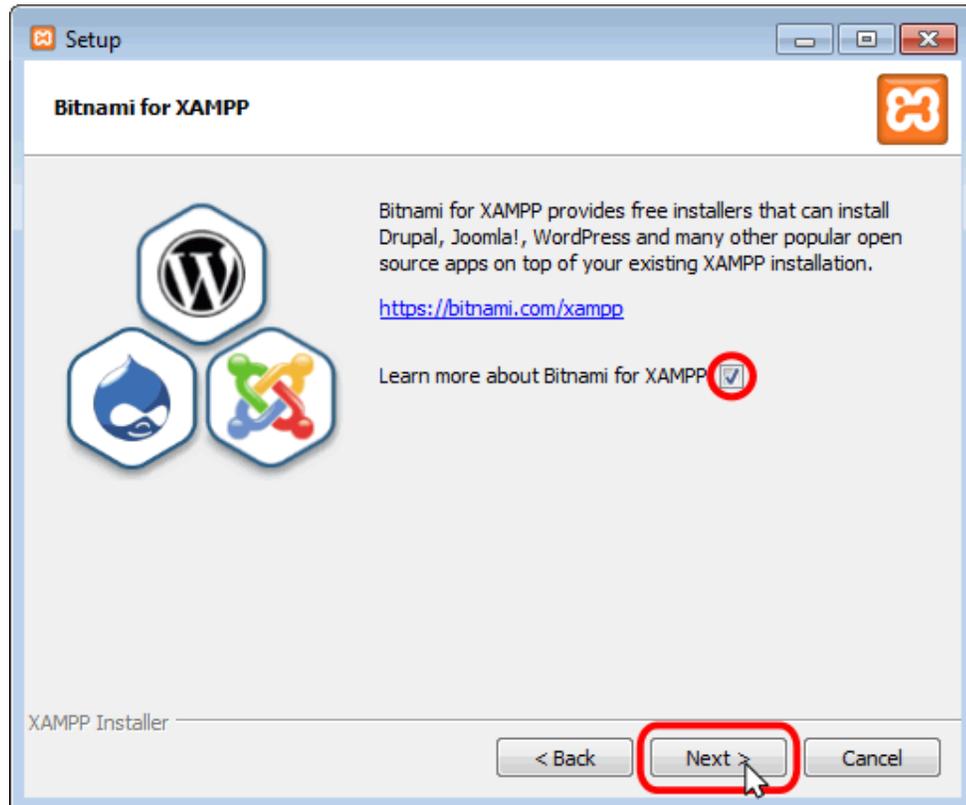


Ilustración 15 pasos de instalación

Para empezar la instalación de XAMPP, hay que hacer clic en el botón "Next" en la pantalla siguiente:

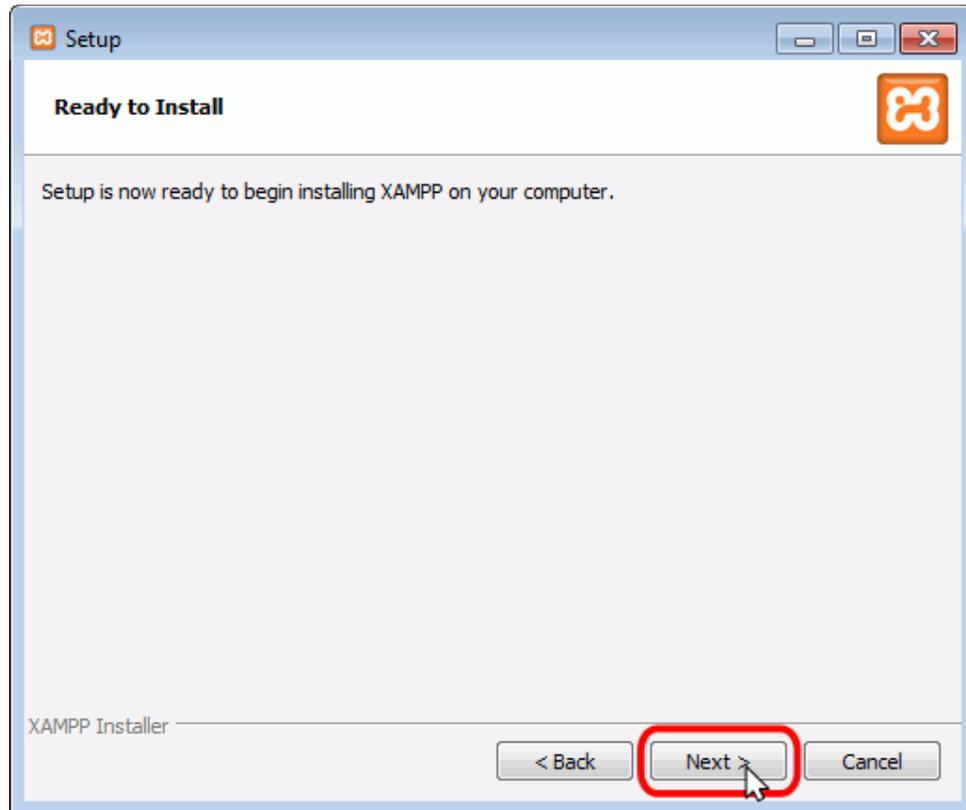


Ilustración 16 pasos de instalación

A continuación, se inicia el proceso de copia de archivos, que puede durar unos minutos en finalizar:



Ilustración 17 pasos de instalación

Durante la instalación, si en el ordenador no se había instalado Apache anteriormente, se mostrará un aviso del cortafuegos de Windows para autorizar a Apache para comunicarse en las redes domésticas o de trabajo, lo que debemos permitir haciendo clic en el botón "Permitir acceso":

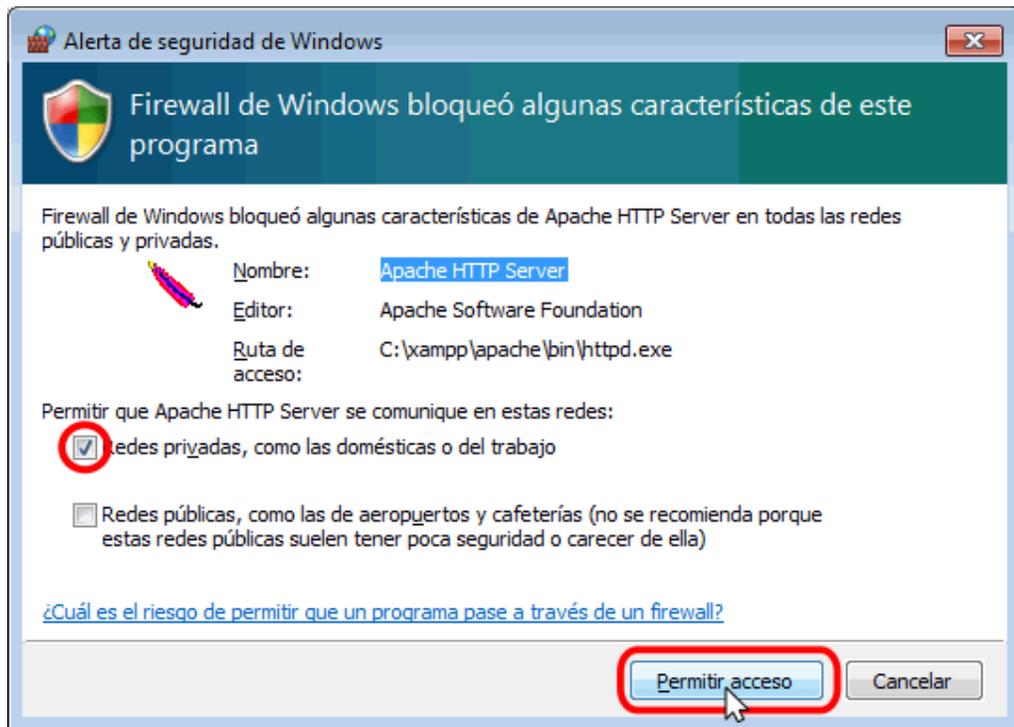


Ilustración 18 configuración para compartir en redes

Una vez terminada la copia de archivos, se muestra la pantalla que confirma que XAMPP ha sido instalado. Hay que hacer clic en el botón "Finish". Para no abrir a continuación el panel de control de XAMPP habría que desmarcar la casilla correspondiente:

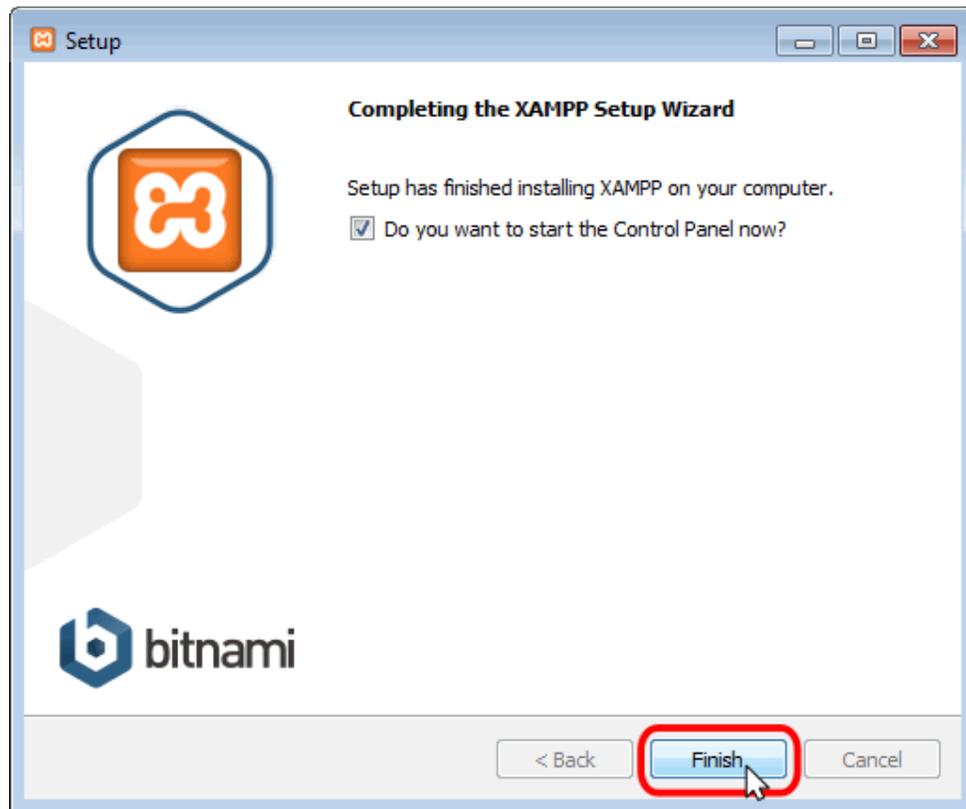


Ilustración 19 finalización de instalación

Una vez instalado procedemos a hacer dos configuraciones básicas, para que el servidor se ejecute cada que inicie el sistema operativo.

1. Ejecutamos XAMPP Control Panel en Inicio>Todos los Programas>XAMPP>XAMPP Control Panel:
2. Una vez inicie el XAMPP Control Panel le damos click en config, luego de esto se despliega una ventana de configuración, en la sección de “Autostart of modules” seleccionamos Apache, MySQL, Start Control Panel Minimized, Enable Tomcat output Windows y Check default ports on startup. Como se muestra a continuación:

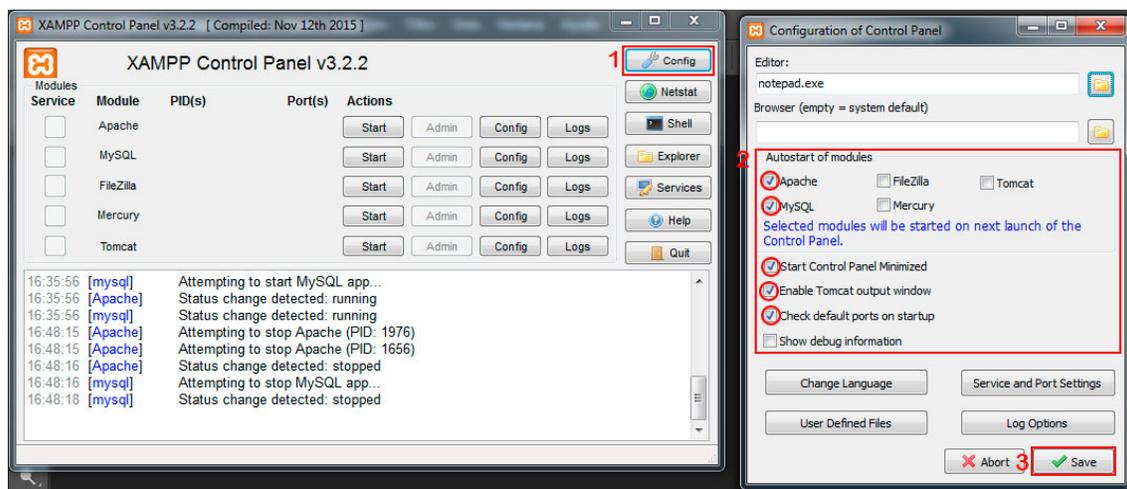


Ilustración 20 panel de control de Xampp

3. Luego de hacer esto vamos a Inicio>Todos los programas >Accesorios >Herramientas del sistema >Programador de tareas, al cargar nos muestra una ventana de tareas y en la parte derecha (Menú Acciones), le damos “Crear tarea básica”, como se muestra:

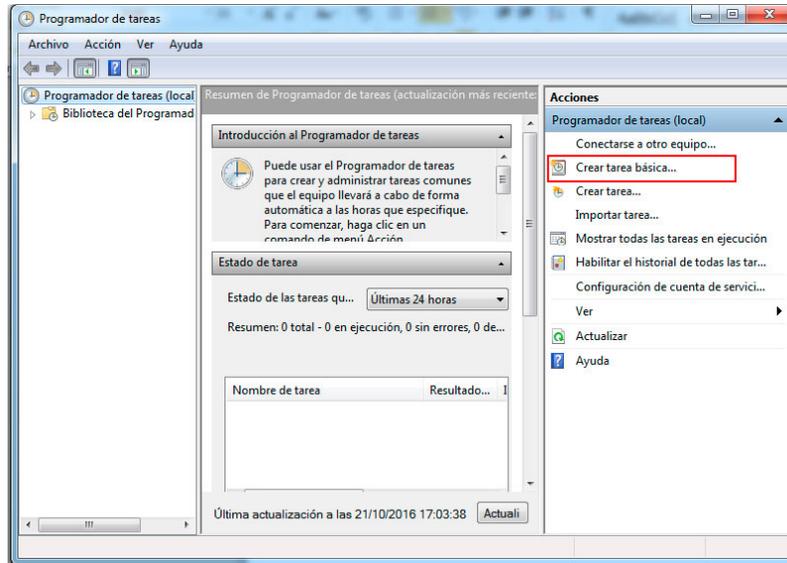


Ilustración 21 creación de la tarea para que el servidor se ejecute al iniciar el sistema

4. Luego de esto se muestra un Asistente donde seguimos los siguientes pasos:

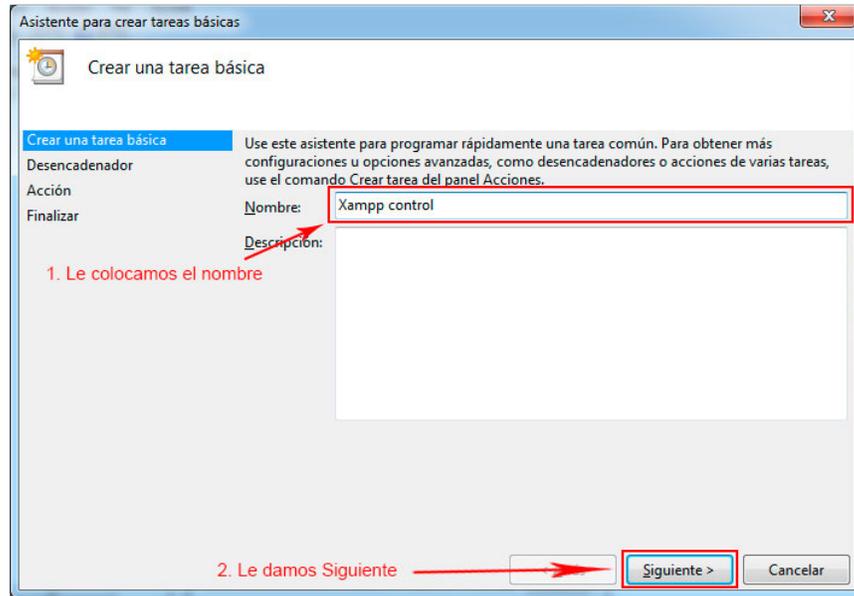


Ilustración 22 configuración para el inicio automático del servidor

5. En el siguiente apartado seleccionamos Al iniciar el equipo y damos siguiente:

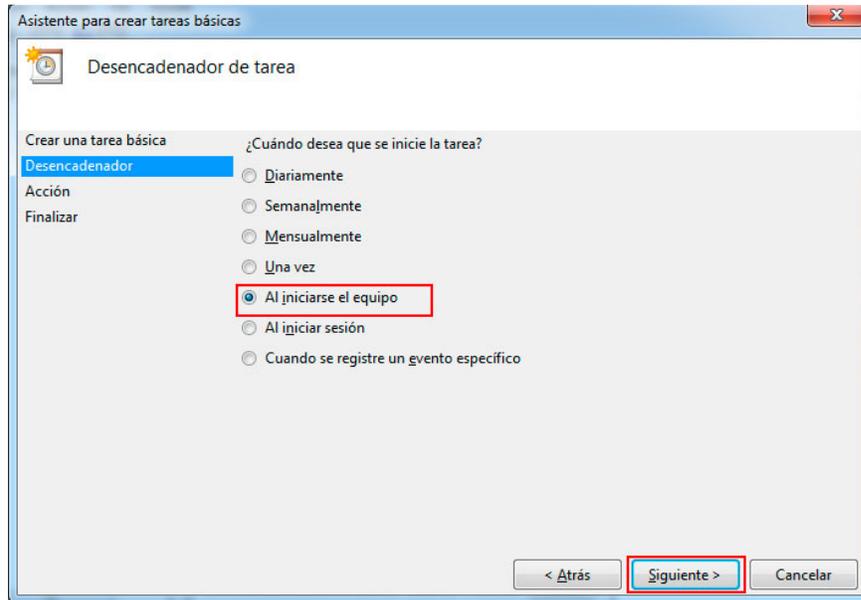


Ilustración 23 configuración de inicio automático del servidor

6. Luego le damos en Iniciar un programa y siguiente:

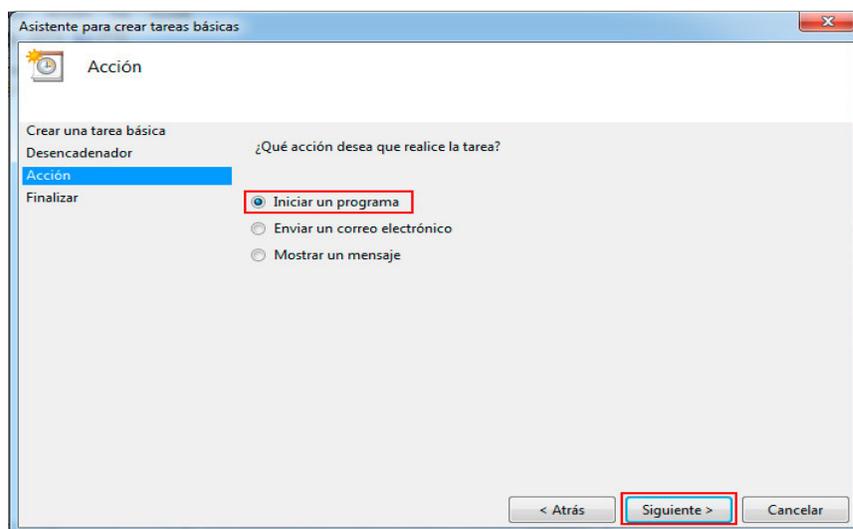


Ilustración 24 Inicio automático del servidor con el sistema

7. Ahora en esta le damos examinar, seleccionamos xampp en el directorio C:\xampp\xampp-control.exe así:

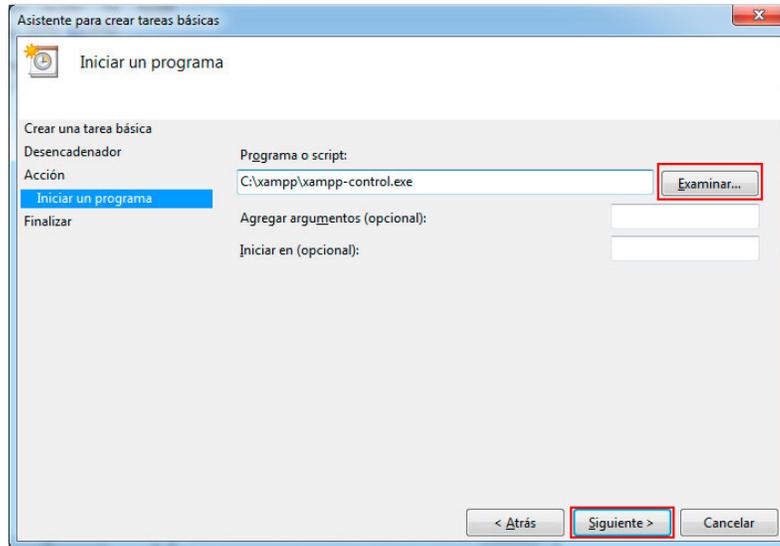


Ilustración 25 Configuración final del arranque del servidor

8. Luego de esto solamente pulsamos finalizar:

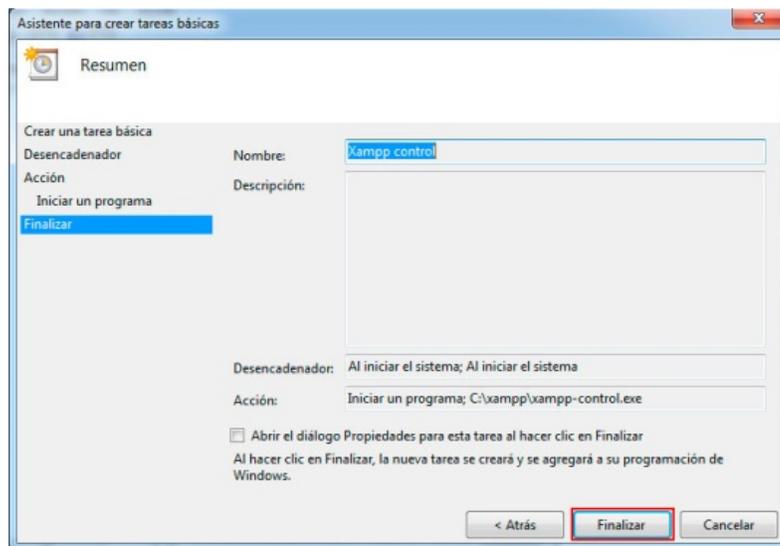


Ilustración 26 finalización de configuración del servidor en el equipo