



Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

**DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD
COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN
FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA**

Williams Francisco Monterroso Maradiaga

Asesorado por la Inga. Mirna Ivonne Aldana Larrazabal

Guatemala, agosto de 2020

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA



FACULTAD DE INGENIERÍA

**DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD
COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN
FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA**

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

WILLIAMS FRANCISCO MONTERROSO MARADIAGA
ASESORADO POR LA INGA. MIRNA IVONNE ALDANA LARRAZABAL

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, AGOSTO DE 2020

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA



NÓMINA DE JUNTA DIRECTIVA

DECANA	Inga. Aurelia Anabela Cordova Estrada
VOCAL I	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Br. Christian Moisés de la Cruz Leal
VOCAL V	Br. Kevin Vladimir Armando Cruz
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANA	Inga. Aurelia Anabela Cordova Estrada
EXAMINADOR	Ing. Sergio Leonel Gómez Bravo
EXAMINADORA	Inga. Floriza Felipa Ávila Pesquera de Medinilla
EXAMINADOR	Ing. Carlos Alfredo Azurdia Morales
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha 9 de enero de 2019.

A handwritten signature in black ink, consisting of a large, stylized 'W' followed by a series of loops and a long horizontal stroke at the bottom.

Williams Francisco Monterroso Maradiaga

Guatemala, 18 de mayo de 2020

Ingeniero
Oscar Argueta Hernández
Director de la Unidad de EPS
Facultad de Ingeniería
Universidad de San Carlos de Guatemala

Estimado Ingeniero Argueta:

Por medio de la presente doy por **finalizado satisfactoriamente** el proyecto de EPS titulado: "DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA". Además hago de su conocimiento que he **revisado y aprobado el informe final del mismo**.

Dicho proyecto e informe fue elaborado por el estudiante: WILLIAMS FRANCISCO MONTERROSO MARADIAGA quien se identifica con registro académico 201313815 y código único de identificación 2839 51176 0101, de la carrera en Ciencias y Sistemas de la Facultad de Ingeniería en la Universidad de San Carlos de Guatemala.

Sin otro particular me despido.

Atentamente,



Mirna Ivonne Aldana Larrazabal
INGENIERA EN CIENCIAS Y SISTEMAS
Colegiada No. 9567

Inga. Mirna Ivonne Aldana Larrazabal
Asesor de EPS
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 18 de mayo de 2020

Ing. Oscar Argueta Hernández
Director Unidad de EPS
Facultad de Ingeniería
Presente

Estimado Ingeniero Argueta Hernández:

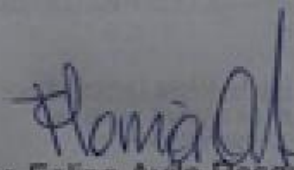
Por este medio atentamente le informo que como Supervisora de la Práctica del Ejercicio Profesional Supervisado, (E.P.S) del estudiante universitario de la Carrera de Ingeniería en Ciencias y Sistemas, **Williams Francisco Monterroso Maradiaga, Registro Académico 201313815 y CUI 2839 51176 0101** procedí a revisar el informe final, cuyo título es **DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA.**

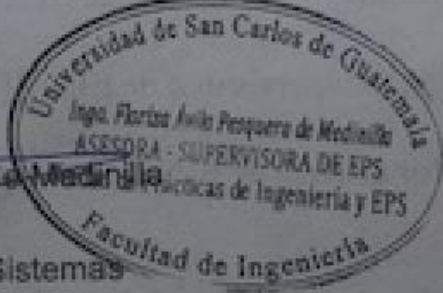
En tal virtud, **LO DOY POR APROBADO**, solicitándole darle el trámite respectivo.

Sin otro particular, me es grato suscribirme.

Atentamente,

"Id y Enseñad a Todos"


Inga. Floriza Felipa Avila Pesquera de Medina
Supervisora de EPS
Área de Ingeniería en Ciencias y Sistemas



Universidad de San Carlos de Guatemala
Inga. Floriza Avila Pesquera de Medina
ASESORA - SUPERVISORA DE EPS
Prácticas de Ingeniería y EPS
Facultad de Ingeniería



Guatemala, 18 de mayo de 2020
REF.EPS. D.223.05.2020

Ing. Carlos Gustavo Alonzo
Director Escuela de Ciencias y Sistemas
Facultad de Ingeniería
Presente

Estimado Ingeniero Alonzo:

Por este medio atentamente le envío el informe final correspondiente a la práctica del Ejercicio Profesional Supervisado, (E.P.S) titulado **DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA DE LA UNIVRSIDAD DE SAN CARLOS DE GUATEMALA**, que fue desarrollado por el estudiante universitario **Williams Francisco Monterroso Maradiaga**, CUI 2839 51176 0101 y **Registro Académico 201313815**, quien fue debidamente supervisado por la Inga. Floriza Felipa Ávila Pesquera de Medinilla, Por lo que habiendo cumplido con los objetivos y requisitos de ley del referido trabajo y existiendo la aprobación por parte de la supervisora, como director apruebo su contenido solicitándole darle el trámite respectivo.

Sin otro particular, me es grato suscribirme.

Atentamente,

“Id y Enseñad a Todos”

Ing. Oscar Argueta Hernández
Director Unidad de EPS



OAH

Nota: esta carta es una copia de la original, la cual se sustituirá por la original al momento de que se normalicen las actividades en la Universidad.



Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

Guatemala, 19 de mayo de 2020

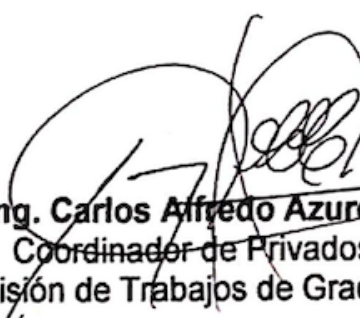
Ingeniero
Carlos Gustavo Alonzo
Director de la Escuela de Ingeniería
En Ciencias y Sistemas

Respetable Ingeniero Alonzo:

Por este medio hago de su conocimiento que he revisado el trabajo de graduación-EPS del estudiante **WILLIAMS FRANCISCO MONTERROSO MARADIAGA** carné 201313815 y CUI 2839 51176 0101, titulado: "DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA" y a mi criterio el mismo cumple con los objetivos propuestos para su desarrollo, según el protocolo.

Al agradecer su atención a la presente, aprovecho la oportunidad para suscribirme,

Atentamente,


Ing. Carlos Alfredo Azurdia
Coordinador de Privados
y Revisión de Trabajos de Graduación



UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA



FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN
CIENCIAS Y SISTEMAS
TEL: 24767644

*El Director de la Escuela de Ingeniería en Ciencias y Sistemas de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer el dictamen del asesor con el visto bueno del revisor y del Licenciado en Letras, del trabajo de graduación “**DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA**”, realizado por el estudiante, **WILLIAMS FRANCISCO MONTERROSO MARADIAGA** aprueba el presente trabajo y solicita la autorización del mismo.*

“ID Y ENSEÑAD A TODOS”



M

Director

Escuela de Ingeniería en Ciencias y Sistemas

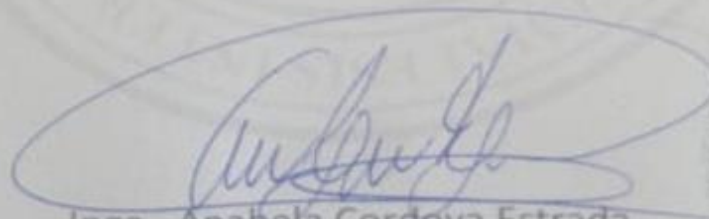
Digitally signed by Carlos Gustavo Alonzo
DN: 2.5.4.13=Profesional Titulado, c=GT,
l=Guatemala / Guatemala, street=Via 5
3-65 zona 4 Ed. El Angel 5to nivel of 52,
2.5.4.20=22347420, ou=NA, o=NA,
title=Ingeniero en Ciencias y Sistemas
Colegiado. 6358, serialNumber=2278
03167 0101, 2.5.4.45=29020980,
2.5.4.27=06/03/79,
email=alonzo@infoutiltygt.com,
cn=Carlos Gustavo Alonzo
Date: 2020.08.25 15:51:36 -06'00'

Guatemala, 23 de agosto de 2020

DTG. 191.2020.

La Decana de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería en Ciencias y Sistemas, al Trabajo de Graduación titulado: **DESARROLLO E IMPLEMENTACIÓN DEL MÓDULO DE ACTIVIDAD COMERCIAL EN EL SISTEMA INTEGRADO DE INFORMACIÓN FINANCIERA DE LA UNIVERSIDAD DE SAN CARLOS DE GUATEMALA**, presentado por el estudiante universitario: **Williams Francisco Monterroso Maradiaga**, y después de haber culminado las revisiones previas bajo la responsabilidad de las instancias correspondientes, autoriza la impresión del mismo.

IMPRÍMASE:



Inga. Anabela Cordova Estrada
Decana



Guatemala, agosto de 2020

ACTO QUE DEDICO A:

- Mi tío** Armando Miguel Maradiaga Turcios, por ser mi figura paterna durante mucho tiempo, por sus consejos, su apoyo incondicional y por todos esos recuerdos que hoy permanecen en mi corazón.
- Mi papi** Julio Enrique Monterroso Alfaro, por su esfuerzo y cariño mientras estuvo conmigo.
- Mi mami** Karen Johana Marisol Maradiaga Turcios, por ser padre y madre casi toda mi vida.
- Mi mamá** Melva Esperanza Turcios García, por todo su apoyo y su alegría.
- Mis tíos** Alex y Walter Maradiaga, por todas sus enseñanzas y recuerdos de felicidad que hoy forman parte de mí.

AGRADECIMIENTOS A:

Universidad de San Carlos de Guatemala	Por formarme como profesional y por todos los momentos de alegría que viví en ella.
Facultad de Ingeniería	Por plantar la semilla de un buen profesional y un mejor ciudadano.
Mi mami	Por su apoyo y amor incondicional durante toda mi vida.
Mi mamá	Por su alegría y apoyo en todo tipo de situaciones.
Mis hermanitas	Alexa, Camila y Karen Monterroso, por llenarme de vida con sus sonrisas.
Mis tíos	Alex, Miguel y Walter Maradiaga, Por ser una importante influencia en mi vida y el ejemplo de una buena persona.
Mis amigos	Juan Veleche, Gerson Rojas, Luis Ramírez y Willy Vásquez, por acompañarme durante mi carrera y apoyarme en los momentos difíciles.

Mi tía	Leticia Turcios, por sus consejos y por ser una referencia profesional para mí.
Mariana de León	Por su amor, su apoyo incondicional y todos los buenos momentos compartidos.
Mi asesora	Ivonne Aldana, por su amistad, sus consejos y su ayuda para formarme como un mejor profesional.
Mis mentores	Gerardo Esquivel y Marvin Mena, por ayudar formarme en una de las etapas más difíciles de mi vida.
Mis compañeros de la Universidad	Por cada momento y sonrisa que pude compartir con ustedes.
Mis hermanos de San Francisco de Sales	Por todo el cariño y los buenos momentos que compartimos en nuestra juventud.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES	V
LISTA DE SÍMBOLOS	IX
GLOSARIO	XI
RESUMEN	XV
OBJETIVOS.....	XVII
INTRODUCCIÓN	XIX
1. FASE DE INVESTIGACIÓN	1
1.1. Antecedentes de la empresa	1
1.1.1. Reseña histórica	1
1.1.2. Misión	2
1.1.3. Visión.....	2
1.1.4. Objetivos.....	3
1.1.5. Servicios que realiza.....	3
1.2. Descripción de las necesidades	5
1.3. Priorización de las necesidades	5
1.4. Diagnóstico FODA del proyecto.....	7
1.4.1. Análisis Interno	7
1.4.1.1. Fortalezas	7
1.4.1.2. Debilidades	8
1.4.2. Análisis externo	8
1.4.2.1. Fortalezas	8
1.4.2.2. Debilidades	8
1.5. Actividad comercial de la Universidad de San Carlos de Guatemala	9

2.	FASE TÉCNICO PROFESIONAL	13
2.1.	Ingeniería de software	13
2.1.1.	¿Qué es software?	13
2.1.2.	¿Qué es ingeniería de software?	13
2.2.	Crecimiento de la tecnología en los últimos años	14
2.2.1.	Internet en la actualidad	15
2.3.	Componentes web	17
2.4.	Shadow DOM	18
2.5.	Herramientas seleccionadas para el desarrollo.....	19
2.5.1.	Cliente	21
2.5.2.	Servidor web	21
2.6.	Angular.....	22
2.6.1.	Características de Angular	24
2.6.2.	Angular Material	25
2.6.3.	¿Por qué utilizar Angular para el proyecto?	26
2.7.	Node.js	27
2.7.1.	Java y Node.js	28
2.7.1.1.	Ventajas Java	28
2.7.1.2.	Desventajas Java	29
2.7.1.3.	Ventajas Node.js	29
2.7.1.4.	Desventajas Node.js.....	31
2.7.2.	Express.js para Node.js.....	31
2.7.3.	Base de datos	32
2.8.	Aplicación utilizada anteriormente.....	33
2.8.1.	Aplicación de nuevas herramientas.....	34
2.9.	Manejo de pagos actual en la USAC.....	34
2.10.	Preliminares de la aplicación.....	36
2.11.	Servidor del proyecto	37
2.11.1.	Archivo del servidor (admin_server.js)	39

2.11.2.	Funciones middleware	41
2.11.2.1.	Autenticación por token	41
2.11.3.	Servicios	43
2.11.4.	Autenticación	45
2.11.4.1.	Autenticación de trabajadores	46
2.11.4.2.	Autenticación de arrendatarios	48
2.11.4.2.1.	Bcrypt	48
2.11.5.	Bitácora.....	49
2.11.5.1.	Registrar acceso del usuario	52
2.11.5.2.	Registrar salida del usuario	54
2.11.5.3.	Registrar inserción de registro	55
2.11.5.4.	Registrar modificación de registro	56
2.11.5.5.	Registrar eliminación de registro	57
2.12.	Cliente del proyecto	58
2.12.1.	Diseño de la aplicación	58
2.12.1.1.	Filtro de búsqueda y botón para crea ..	62
2.12.2.	Distribución de los componentes	65
2.12.3.	Lazy Loading	67
2.12.3.1.	Implementación de Lazy Loading	69
2.12.4.	AuthGuard	72
2.12.5.	RoleGuard	74
2.12.6.	Gateway.....	75
2.13.	Ambientes de la aplicación	76
2.13.1.	Desarrollo	76
2.13.2.	Pruebas y producción	76
2.14.	Atención a los requerimientos	77
2.14.1.	Evaluación de objetivos	79
2.14.2.	Ventajas del proyecto	90

3.	FASE SE ENSEÑANZA – APRENDIZAJE	93
3.1.	Estrategia de enseñanza.....	93
3.2.	Aplicación altamente intuitiva	93
3.2.1.	Iconos asociados a acciones.....	94
3.2.2.	Elementos de ayuda en formularios	95
3.2.2.1.	Anotaciones de error en formularios.....	95
3.2.2.2.	Listas de selección en formularios.....	96
3.2.3.	Activación de botones para realizar acción	97
3.2.4.	Anotaciones en procesos	98
3.3.	Fase de pruebas con usuarios	100
	CONCLUSIONES.....	103
	RECOMENDACIONES.....	105
	BIBLIOGRAFÍA.....	107
	APÉNDICES.....	109
	ANEXOS.....	115

ÍNDICE DE ILUSTRACIONES

FIGURAS

1.	Organigrama del Departamento de Procesamiento de Datos	4
2.	Distribución de archivos en la Oficina de Actividad Comercial	10
3.	Arquitectura de una aplicación web.....	20
4.	Descargas de Angular/core los últimos 2 años	23
5.	Interés a través del tiempo los últimos 5 años.....	23
6.	Servidor web mínimo con Node.js y Express.js.....	32
7.	Pantalla de aplicación anterior	33
8.	Arquitectura de la solución	36
9.	Estructura de archivos del servidor	37
10.	Scripts del servidor.....	38
11.	Encabezado del servidor.....	40
12.	Autenticación por token.....	42
13.	Parte superior router>index.js	43
14.	Estructura de servicios	44
15.	Estructura del router.....	45
16.	Autenticación usando LDAP.....	47
17.	Uso de bcrypt	49
18.	Registrar acceso del usuario.....	53
19.	Registrar salida del usuario.....	54
20.	Registrar inserción de registro.....	55
21.	Registrar modificación de registro	56
22.	Registrar eliminación de registro	57
23.	Diseño para menú del prototipo	58

24.	Diseño para menú de la aplicación	59
25.	Diseño de la aplicación existente para áreas de influencia.....	60
26.	Diseño de la aplicación para áreas de influencia	61
27.	Diseño de tabla del prototipo	63
28.	Servicio para búsqueda	64
29.	Uso del servicio para búsqueda desde componente	65
30.	Estructura de componente home-arrendatario.....	66
31.	Lazy Loading.....	68
32.	Lazy Loading router principal	69
33.	Lazy Loading HomeArrendatarioModule.....	70
34.	Lazy Loading cargar hijos	71
35.	AuthGuard.....	73
36.	RoleGuard.....	74
37.	Estructura de llaves para Gateway	75
38.	Pantalla para visualización de asociaciones	80
39.	Pagos disponibles para una asociación	80
40.	Visualización de área de influencia	81
41.	Crear una nueva área de influencia	82
42.	Editar un área de influencia	83
43.	Borrar un área de influencia.....	84
44.	Visualización de pagos por arrendatario	84
45.	Exoneración de pagos	85
46.	Filtros para consultar arrendatarios	86
47.	Resumen de cuenta corriente	86
48.	Selección de contrato para detalle de cuenta corriente	87
49.	Fragmento de detalle de cuenta corriente.....	88
50.	Filtros para consultar locales	89
51.	Icono acompañado de anotación en menú	94
52.	Anotación de error en campo de formulario	96

53.	Lista de selección en formulario	97
54.	Activación de botón para guardar.....	98
55.	Anotación en proceso de crear área de influencia	99
56.	Anotación en iniciar proceso de arrendamiento	100

TABLAS

I.	Categorización de necesidades	6
II.	Diferencias entre ingeniería de software y ciencias de la computación	14
III.	Diez países con más usuarios de Internet	16
IV.	Resumen de casos de prueba no satisfactorios.....	79
V.	Tiempos para generar acciones	101

LISTA DE SÍMBOLOS

Símbolo	Significado
GB	Gigabytes
GHz	Giga Hertz
TB	Terabytes

GLOSARIO

API	Interfaz de aplicación o <i>Application Programming Interface</i> . Es un conjunto de funciones que se crean con el fin de ser utilizadas por otro sistema.
Big data	Concepto que se refiere a cantidades muy grandes de información cuyo manejo necesita de procesos diferentes.
Callback	Es una función que se ejecuta al terminar otra y se manda como parámetro.
Chrome's V8 engine	Es un motor de código abierto hecho para JavaScript utilizado por el navegador de Google, Chrome.
CSU	Consejo Superior Universitario.
Deploy	En español significa desplegar. Se refiere al proceso de pasar una aplicación a un ambiente donde los usuarios puedan hacer uso de él.
Embebido	Es un sistema incrustado, que realiza algunas funciones dedicadas para otro sistema al que pertenece.

Endpoint	Dirección en la que se ubica un servicio
<i>Feedback</i>	En español, retroalimentación. Son respuestas que se transmiten con base en un mensaje recibido.
<i>Framework</i>	Esquema que se usa para desarrollar y organizar software.
JSON	Es un formato estándar basado en objetos del tipo atributo – valor.
<i>Out-of-the-box</i>	Listo para usar, en referencia al software se refiere a un programa computacional que puede ser utilizado por los usuarios.
<i>Plugin</i>	Aplicación que contiene un set de características y funciones que ayudan a mejorar un sistema.
Polymer	Set de librerías, herramientas y estándares para una mejor aplicación web.
<i>React</i>	Framework para JavaScript para desarrollo de aplicaciones web basada en componentes.
Renderizar	Es un proceso que desarrollan los navegadores para poder mostrar una página web a partir de código.
<i>Runtime</i>	Es un estado de máquina virtual que aprovisiona a los procesos para poder ejecutar un programa.

SIIF	Sistema Integrado de Información Financiera.
Typescript	Lenguaje de programación de código abierto desarrollado y mantenido por Microsoft.
UI	Interfaz de usuario, <i>User Interface</i> , es todo lo que el usuario observa.
USAC	Universidad de San Carlos de Guatemala.
Vue	Framework para JavaScript para desarrollo de aplicaciones web progresivas.
Widget	Es un programa pequeño que agrega una funcionalidad específica a un sistema.
Workers	Hilos de ejecución de un programa informático.

RESUMEN

La actividad comercial dentro de la Universidad de San Carlos de Guatemala crecido en los últimos años. La entidad que maneja esta actividad, la oficina de Actividad Comercial ha necesitado utilizar los recursos que tiene a su alcance para controlar las personas y negocios que existen dentro de la Universidad.

Con hojas de cálculo se ha llevado un control superficial; se ha invertido una gran cantidad de tiempo en la organización y actualización para tener información actualizada del estado de los arrendatarios. Además, estos deben hacer un proceso largo para realizar sus pagos, ya que deben recorrer varias entidades de la Universidad sin tener otra opción.

Debido a la situación en la que se encuentra la oficina de Actividad Comercial, se desarrolló una aplicación web que ayude a automatizar muchos de sus procesos, como el control de arrendatarios y pagos, que permite a los usuarios tener la información disponible en cualquier momento. Esta aplicación web será un módulo nuevo dentro del SIIF dedicado a la Actividad Comercial. Los arrendatarios también serán beneficiados, ya que se incorporó una alternativa de pago (en línea), para que no tengan que hacer el proceso antiguo, a menos que ellos lo deseen. Toda la aplicación recoge información sobre los pagos de los arrendatarios para luego distribuirlos a las entidades correspondientes dentro de la Universidad.

Para finalizar, el personal es capacitado y el sistema puesto en producción en los servidores del Departamento de Procesamiento de Datos de la

Universidad de San Carlos de Guatemala, para que los usuarios hagan uso de este.

OBJETIVOS

General

Expandir el Sistema Integrado de Información Financiera (SIIF) de la Universidad de San Carlos de Guatemala al agregarle el módulo de actividad comercial.

Específicos

1. Proveer de un módulo para consultar las asociaciones que pertenecen a cierta área de influencia y visualizar la cantidad de presupuesto que tienen disponible.
2. Proveer una interfaz web para registrar, editar, borrar y consultar las entidades de arrendatario, local, pago y área de influencia dentro de la Universidad de San Carlos de Guatemala.
3. Proveer un medio tecnológico debidamente documentado para actualizar la información del estado del contrato asociado a un local y que lleve el control de la cuenta corriente del mismo.
4. Proveer un módulo capaz de consultar el estado, en tiempo real, en el que se encuentra el proceso de adjudicación.

INTRODUCCIÓN

“Tengo todo perfectamente descontrolado”, una cita que muchos de nosotros hemos aplicado sin siquiera saberlo. Seguramente el descontrol es una actividad o un hábito que se ve disfrazado por la costumbre, mantenemos el descontrol diariamente sin saber de su existencia.

Algo similar pasa con muchos procesos, como es el caso de la actividad comercial de la Universidad de San Carlos de Guatemala. No se cuenta con un mecanismo bien implementado para llevar el control de la misma. Además, el proceso actual, que se ha propuesto como primera respuesta y de forma inmediata, es bastante mecánico e involucra muchos recursos por parte de la oficina y de las personas que arriendan algún local dentro de la Universidad.

Como respuesta y con el uso de la tecnología se desarrolló un sistema para que automatice el proceso actual, lo mejore y logre descentralizar toda la actividad que actualmente se realiza dentro de la oficina de Actividad Comercial. El sistema será capaz de visualizar toda la información necesaria para el administrador y el arrendatario, tomando en cuenta los sistemas de información actuales y sus propuestas para tener software de calidad.

Tal sistema tendrá varias fases para su eficiente desarrollo. Al entregarlo, los usuarios involucrados en el proceso podrán ser capaces de utilizarlo, enfocar sus esfuerzos en otros procesos y, de esta forma, mejorar toda la actividad comercial dentro de la Universidad de San Carlos de Guatemala

1. FASE DE INVESTIGACIÓN

1.1. Antecedentes de la empresa

La actividad comercial dentro de la Universidad de San Carlos de Guatemala, USAC es una de las actividades que ha tenido más crecimiento en los últimos años debido al aumento en la población estudiantil; en consecuencia, se han tomado decisiones para tener un mejor control de esta.

1.1.1. Reseña histórica

A diario, un aproximado de 90 000 personas circulan en las instalaciones de la USAC. Mientras están dentro buscan satisfacer distintas necesidades, como alimentación, adquisición de material para desarrollar sus actividades, transporte, entre otras.

Dentro del campus hay una gran variedad de productos y servicios que buscan satisfacer dichas necesidades. Con el crecimiento de la población estudiantil de la Universidad ha crecido en conjunto el comercio. Con el fin de regular toda esta actividad comercial, el Consejo Superior Universitario (CSU) aprobó en 2013 un reglamento.

Además del reglamento se creó la Comisión para la Actividad Comercial, la cual se encarga de evaluar periódicamente la prestación de los bienes y servicios a los consumidores; recibe denuncias, aplica medidas correctivas y sanciones, emite resoluciones sobre la autorización y renovación de comercios. Cada dos años propone al CSU las tarifas que se deben cobrar por los arrendamientos y

se encarga de la desocupación y desalojo de los comercios en caso de ser necesario. La Oficina de Actividad Comercial hace efectivas las decisiones que se toman dentro de la comisión.

Actualmente, para evitar la mala distribución de los puestos comerciales se lleva a cabo una reorganización. Han sido clausurados los negocios que no cumplían con lo estipulado en el reglamento y han demolido locales que no tenían una infraestructura adecuada. El objetivo es revitalizar y remodelar los espacios comerciales para brindar un mejor servicio y tener un mayor control sobre la actividad comercial dentro de la Universidad.

Con base en lo anterior, la Oficina de Actividad Comercial solicitó al Departamento de Procesamiento de Datos un software para apoyar con la organización y control de los espacios comerciales, el cual debe ser capaz de apoyar las actividades que actualmente se desarrollan dentro de dicha oficina.

1.1.2. Misión

“Somos el Departamento, de la Dirección General Financiera de la Universidad de San Carlos de Guatemala, referente por excelencia en tecnologías de información y comunicación, a través del desarrollo de software a la medida, para coadyuvar al desarrollo de los procesos docentes, de investigación, de extensión y administrativos y a la vez, administrar los servicios de red y telecomunicaciones –red de datos-.”¹

1.1.3. Visión

“En concordancia con los fines universitarios de gestión actualizada, dinámica, efectiva y con recursos óptimamente utilizados, ser el ente que proporcione las directrices, infraestructura, asesoría y desarrollo en sistemas de información, telecomunicaciones y tecnología en general para la Universidad de San Carlos de Guatemala.”²

¹ USAC. *Departamento de Procesamiento de Datos*. <http://dpd.usac.edu.gt>.

² *Ibíd.*

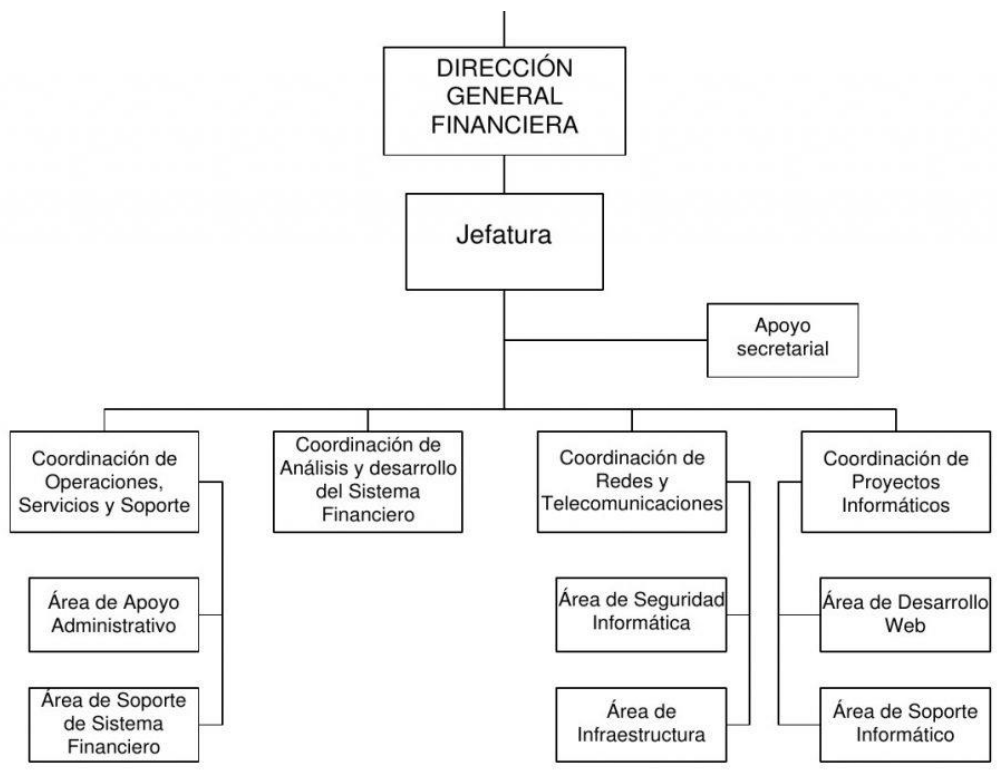
1.1.4. Objetivos

- Mantener y mejorar el sistema informático, modular y extensible, para el manejo del sistema financiero.
- Brindar asesoría en informática y telecomunicaciones orientada a buscar las mejores soluciones integrales para el beneficio de la comunidad universitaria.
- Administrar la red troncal de datos, brindando acceso a las distintas dependencias.
- Promover el uso de tecnología idónea para satisfacer los planes, programas y proyectos de la Universidad de San Carlos de Guatemala.

1.1.5. Servicios que realiza

El Departamento de Procesamiento de Datos está organizado de la siguiente forma:

Figura 1. **Organigrama del Departamento de Procesamiento de Datos**



Fuente: Departamento de Procesamiento de Datos. *Organigrama*. <http://dpd.usac.edu.gt>.
 Consulta: 1 de mayo de 2019.

El proyecto pertenece a la Coordinación de Análisis y Desarrollo del Sistema Financiero, la cual se encarga de desarrollar software a la medida para las entidades dentro de la Universidad, con el fin de satisfacer las necesidades de sistematización y automatización de procesos académicos, docentes y administrativos a través de su área de desarrollo web.

1.2. Descripción de las necesidades

Para identificar las necesidades y el alcance del proyecto se llevaron a cabo reuniones frecuentes con el responsable del proyecto, licenciado Aldrin Fuentes, y la arquitecta Wendy Estrada, profesional de la Oficina de Actividad Comercial, quienes plantearon las siguientes necesidades:

- Una aplicación web que sea capaz de dar seguimiento a la cuenta corriente de los locales, además de brindar la opción de pago por la banca virtual.
- Evaluación de herramientas que mejor se adapten a las necesidades de la aplicación, tomando en cuenta que deben ser de código abierto para que no sea necesario pagar por usarlas y, de esta forma, complicar el desarrollo.
- Documentación suficiente sobre la aplicación para futuro mantenimiento y posible continuidad.
- Brindar capacitación y acompañamiento durante un período corto para que el usuario final sea capaz de utilizar el software de forma correcta.

Para observar los requerimientos iniciales se presenta una lista completa en el apéndice 1 del presente documento.

1.3. Priorización de las necesidades

La priorización se delimitó con la técnica MoSCoW, la cual clasifica los requisitos en las siguientes categorías, ordenadas de mayor a menor prioridad:

- **Must:** son los requisitos que debe tener; no son negociables de ningún modo. Sin estos requisitos el producto no podría ser usado por los usuarios.
- **Should:** son todos los requisitos que deben estar para tener una buena experiencia más allá de la usabilidad del producto.
- **Could:** son aquellos requisitos cuya ausencia no perjudicará en gran medida al producto final, pero su desarrollo será agradecido.
- **Would/Won't:** son los requisitos de muy baja prioridad, que aportan muy poco al producto final y, sin ellos, el producto será bien recibido.

Para las necesidades generales de la aplicación se presenta la siguiente categorización:

Tabla I. **Categorización de necesidades**

Necesidad	Categoría
Implementación de base de datos en Oracle.	Must.
Mantenimiento, por usuario administrador, de la configuración general.	Must.
Actualizar el pago realizado por un arrendante automáticamente.	Must.
Generar boleta de pago desde aplicación, por usuario arrendante.	Should.
Generación de notas automáticas.	Should.
Aplicación para control de presupuesto por arrendante según área de influencia.	Could.
Documentación suficiente de la aplicación, técnica y de usuario.	Must.

Fuente: elaboración propia.

1.4. Diagnóstico FODA del proyecto

Para tener un mejor panorama sobre el proyecto se realizó un diagnóstico previo al desarrollo del proyecto dividido en análisis interno y externo.

1.4.1. Análisis Interno

Se evaluaron las fortalezas y debilidades internas del proyecto, tomando en cuenta a los interesados de este.

1.4.1.1. Fortalezas

Las fortalezas internas del proyecto son las siguientes:

- Se tiene la infraestructura a disposición para llevar a cabo el proyecto.
- El departamento cuenta con proyectos de informática que apoyan la realización de este. Se cuenta con personas afines a la carrera de Ciencias y Sistemas.
- Hay interés por parte de las personas del departamento en la realización del proyecto, y también de la unidad de Actividad Comercial.
- Infraestructura del SIIF.
- Ambiente de despliegue listo.

1.4.1.2. Debilidades

Las debilidades internas del proyecto son las siguientes:

- El proceso de arrendamiento no depende únicamente de la actividad comercial.
- Se depende de otras unidades para el flujo del proceso.
- Existe rotación del personal cada cambio de autoridades.

1.4.2. Análisis externo

Se evaluaron las fortalezas y debilidades externas del proyecto, tomando en cuenta a los interesados y el contexto de este.

1.4.2.1. Fortalezas

La fortaleza externa del proyecto es la posibilidad de interconectar el módulo de pagos en línea con el sistema.

1.4.2.2. Debilidades

Las fortalezas externas del proyecto son las siguientes:

- Dependier de la disponibilidad del sistema.
- Cambio de autoridades cada 4 años.
- Cierre de instalaciones por manifestaciones.
- Resistencia al cambio por parte de las personas de Actividad Comercial.

1.5. Actividad comercial de la Universidad de San Carlos de Guatemala

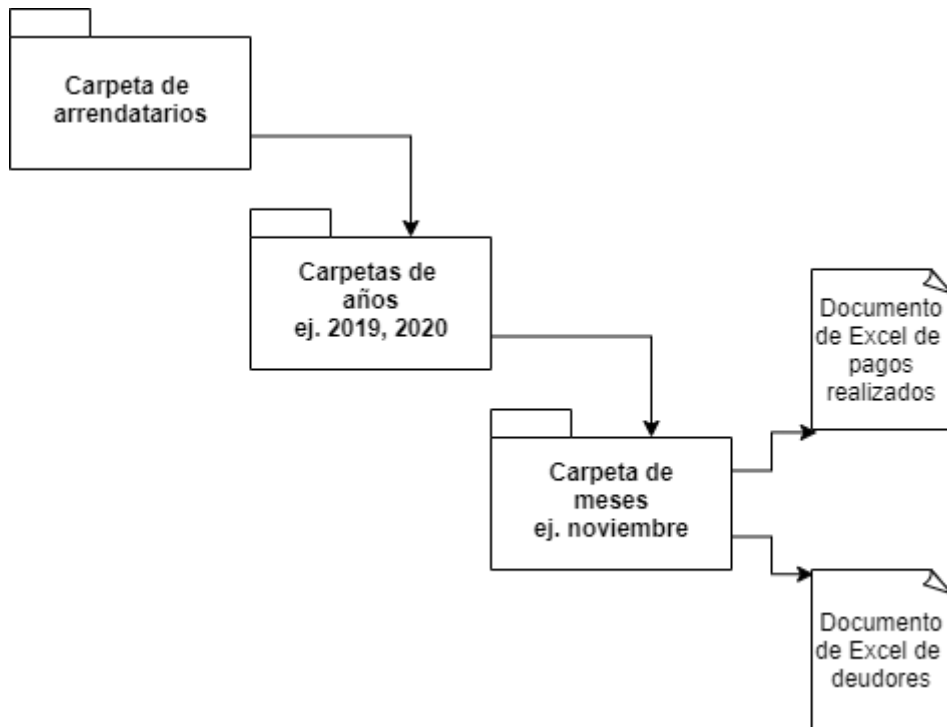
La actividad comercial dentro de la USAC se controla por medio de la Comisión para la Actividad Comercial, la cual está conformada de la siguiente forma:

- Director de la Dirección de Asuntos Jurídicos.
- Jefe de la División de Servicios Generales.
- Jefe de la Sección de Cobros de la Dirección General Financiera.
- Jefe de la Unidad de Salud de la División de Bienestar Estudiantil.
- Jefe del Laboratorio Microbiológico de Alimentos de la Facultad de Ciencias Químicas y Farmacia.
- Representante Estudiantil del Consejo Superior Universitario.
- Representante designado por el señor Rector.

Cada persona y entidad que conforma la comisión tiene funciones asociadas para el correcto funcionamiento de esta. La Oficina de Actividad Comercial se encarga de intermediar con los arrendatarios que quieren ingresar un negocio a la Universidad o los existentes. Por esto, la aplicación se enfoca en mejorar el proceso que se desarrolla dentro de la misma, para que sea un punto de inicio y luego se incorporen todos los involucrados de la Comisión.

Actualmente, la actividad comercial dentro de la Universidad de San Carlos de Guatemala se maneja por medio de documentos de Excel, por personas dentro de la Oficina de Actividad Comercial. Distribuyen dichos documentos en carpetas según sea el mes y año que le corresponde. Cada parte del proceso se debe realizar dentro de la oficina.

Figura 2. **Distribución de archivos en la Oficina de Actividad Comercial**



Fuente: elaboración propia, empleando draw.io.

Partiendo de lo mencionado, estamos viviendo una revolución tecnológica desde hace algunos años, sin darnos cuenta y muchas veces sin entender cómo funciona la tecnología con la que convivimos a diario. Los cambios en la tecnología nos obligan a adaptarnos a ellos y a estar constantemente en un

aprendizaje, por lo que la USAC, no puede quedarse atrás en la vanguardia tecnológica en cada una de sus operaciones y procesos. Ya conocemos la tecnología y ahora entramos en la etapa de poner en práctica nuestros conocimientos.

2. FASE TÉCNICO PROFESIONAL

2.1. Ingeniería de software

En la actualidad, la ingeniería de software juega un papel importante en la sociedad, con un desarrollo cada vez más rápido. La economía incluso depende de esta, ya que cada día se utiliza más software para automatizar sistemas.

2.1.1. ¿Qué es software?

Software es un conjunto de programas de computadora que están hechos para crear alguna característica, funcionalidad o mejorar el desempeño de un sistema. Los productos de software pueden ser:

- Genérico: es desarrollado para un grupo de clientes utilizando las mismas funcionalidades. Utiliza un grado de abstracción un poco mayor.
- Específico: se ajusta a requerimientos solicitados por un cliente en particular.

Una vez dicho qué es el software entramos directamente a qué es la ingeniería de software.

2.1.2. ¿Qué es ingeniería de software?

Es una disciplina que utiliza teorías, métodos y herramientas para crear un producto de software profesional. Adopta un enfoque sistemático y ordenado,

dependiendo de la naturaleza del problema que se busca resolver y sus restricciones. Se puede utilizar diferentes métodos, teorías y herramientas para desarrollar un software de calidad.

Con la creciente popularidad del concepto de ciencias de la computación se puede confundir con ingeniería de software, por lo que se detallan sus diferencias a continuación.

Tabla II. **Diferencias entre ingeniería de software y ciencias de la computación**

Ingeniería de software	Ciencias de la computación
Se encarga de problemas prácticos del desarrollo de software.	Se encarga de las teorías y métodos usados por las computadoras y sistemas de software.
Algún conocimiento de las ciencias de la computación es utilizado por la ingeniería de software para resolver problemas.	Algunas teorías no pueden ser aplicadas completamente por la ingeniería del software cuando la solución propuesta a problemas complejos o de la vida tienen que ser desarrollados.

Fuente: PUNTAMBEKAR, Anuradha A. *Software Engineering and Quality Assurance*. p 2.

2.2. Crecimiento de la tecnología en los últimos años

En la actualidad, la tecnología está ganando cada vez más terreno. El crecimiento constante de Internet y la facilidad con la que se nos presentan las

herramientas tecnológicas son los principales factores del crecimiento exponencial experimenta.

Viendo un poco al pasado, tenemos que viajar cuatro décadas para encontrar la creación del primer computador portátil. Su precio para la época era muy elevado, comparado con lo fácil que es actualmente obtener una computadora en nuestras casas. Cada vez podemos acceder a más capacidad de almacenamiento y procesamiento por un precio más bajo. Cada día se crean nuevas herramientas para que las podamos usar. Ahora tenemos la información al alcance de un clic, lo que ha ayudado a muchas personas a crecer intelectualmente. Cada día sabemos más del uso de nuestras computadoras y teléfonos celulares. Son parte de nuestra rutina hace mucho y ahora es muy difícil imaginar una vida sin ellos.

2.2.1. Internet en la actualidad

Los usuarios en Internet cada día aumentan. Según la CIA (Agencia Central de Inteligencia de los Estados Unidos de América), Guatemala cuenta actualmente con 5 241 952 usuarios de Internet³. A continuación, se muestran los diez países con más usuarios de Internet según la CIA:

³ Country comparison. *Internet users*. <https://www.cia.gov/library/publications/the-world-factbook/fields/204rank.html>.

Tabla III. **Diez países con más usuarios de Internet**

#	País	Usuarios
1	China	730 723 960
2	India	374 328 160
3	Estados Unidos	246 809 221
4	Brasil	122 841 218
5	Japón	116 565 962
6	Rusia	108 772 470
7	México	73 334 032
8	Alemania	72 365 643
9	Irán	67 600 000
10	Indonesia	65 525 226

Fuente: Central Intelligence Agency. *The World Factbook*.

<https://www.cia.gov/library/publications/the-world-factbook/fields/204rank.html>. Consulta: 25 de marzo de 2020.

Es una realidad el hecho de que vivimos ligados a la tecnología, ya sea desde una computadora o un teléfono celular. Consumimos contenido de Internet y cada día aumenta, por lo que debemos saber adaptarlo a nuestras vidas para facilitar nuestras tareas.

Gracias al auge del Internet ahora ha habido una tendencia al desarrollo web, específicamente, del concepto de componentes web, que cada día se asienta con más fuerza gracias a sus características y beneficios.

2.3. Componentes web

La idea del desarrollo de componentes web no es actual. No precisamente el término “componente”, pero quizá *widget* o *plugin* fueron los primeros pioneros de lo que ahora conocemos como componentes web. En la actualidad tenemos *frameworks* y librerías como React, Angular, Vue o Polymer, los cuales toman el concepto de componente.

Entonces, ¿qué es un componente? Un componente es una pieza reusable de interfaz de usuario. Se puede pensar en seleccionadores de fecha y listas de selección, los cuales podrán utilizarse entre plataformas diferentes para facilitar a los programadores la creación de estas características.

Las especificaciones se empezaron a desarrollar aproximadamente en 2013, pero algunas tecnologías empezaron a usarse antes. Los componentes web supusieron una revolución en el mundo del desarrollo web en términos de reutilización y encapsulamiento, porque permiten al usuario encapsular código HTML, CSS y JavaScript sin que el código que lo alberga lo afecte, a menos que sea a propósito del usuario.

Los componentes web no son una tecnología como tal. Se basan en cuatro tecnologías que se pueden usar de forma independiente que son:

- Custom Elements: permite definir nuevos tipos de elementos.
- HTML Imports: permite importar archivos HTML dentro de archivos HTML para su reutilización.

- **Templates:** permiten crear porciones de código HTML que pueden ser replicados tantas veces como sea necesario.
- **Shadow DOM:** esta tecnología permite encapsular trozos de HTML, CSS y JavaScript para que no afecte el código de la máquina.

Entonces, un componente web está hecho de la siguiente forma: se introduce el Shadow DOM dentro del Template de un Custom Element, que luego puede ser importado desde otro componente, según sea la necesidad del usuario.

2.4. Shadow DOM

Es usado por las máquinas para poder entender el lenguaje HTML (Modelo de objeto del documento, por sus siglas). Cuando el navegador carga una página, transforma el HTML en un documento vivo. Para comprender la estructura de la página se analiza el HTML y se genera un modelo de datos. La jerarquía se preserva creando un árbol de nodos; es decir, el DOM. Los nodos que genera el navegador contienen métodos y propiedades, los cuales se pueden manipular con programas.

Al entender el DOM entramos al Shadow DOM, el cual se diferencia por dos cosas:

- **Cómo se crea/usa.**
- **Cómo se comporta en relación con el resto de la página.**

“Normalmente, creas nodos del DOM y los agregas como secundarios de otro elemento. Con shadow DOM, creas un árbol del DOM dentro del ámbito, adherido al elemento, pero separado de su secundario real. Este subárbol con ámbito se llama

shadow tree (árbol en las sombras). El elemento está adherido a su shadow host (host en las sombras). Todo lo que agregues en las sombras se hace local para el elemento de hosting, incluido el <style>. Así es como shadow DOM logra acotar el ámbito de estilo de CSS.⁴

2.5. Herramientas seleccionadas para el desarrollo

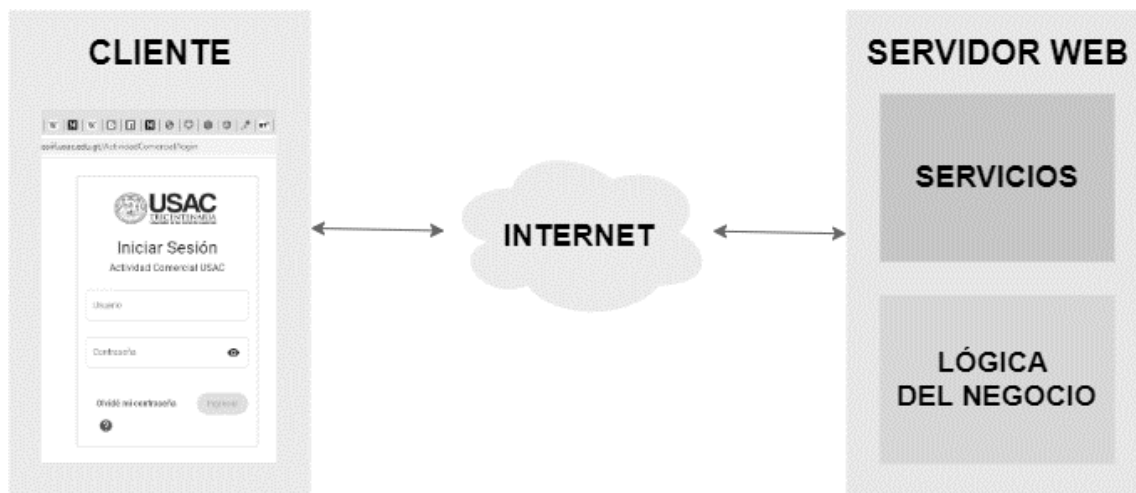
Debido al crecimiento en el desarrollo de aplicaciones web, se planteó crear una para atender los requerimientos de los usuarios de la Oficina de Actividad Comercial. ¿Por qué una aplicación web? Los motivos principales por los que se decidió desarrollar una aplicación web son:

- Porque necesita menos recursos que una aplicación instalada. Se puede utilizar cualquier equipo que tenga navegador web para usarla.
- Es fácil de usar. Con conocimientos básicos de informática se puede hacer uso de una aplicación web que tenga un diseño medianamente intuitivo.
- Gracias a su independencia del hardware causa menos problemas técnicos.
- Permite a varios usuarios visualizar la información al mismo tiempo y se puede acceder desde cualquier lugar con una conexión a Internet.
- Los datos se encuentran seguros, ya que no se debe preocupar por cuidar el disco duro de la computadora donde están almacenados. Estos permanecen en el servidor, que se encarga de distribuirlos sobre demanda de los usuarios.

⁴ BIDELMAN, Eric. *Shadow DOM v1: Componentes web independientes*. <https://developers.google.com/web/fundamentals/web-components/shadowdom>.

Una vez tomada la decisión sobre el tipo de aplicación a desarrollar queda por definir el *stack* de herramientas que se van a utilizar. Una aplicación web, por lo regular, sigue la siguiente arquitectura:

Figura 3. **Arquitectura de una aplicación web**



Fuente: elaboración propia, empleando draw.io.

Como se puede ver, la arquitectura consta de dos partes grandes, las cuales son el cliente y el servidor web.

Un aspecto que resalta es que los elementos no dependen estrechamente entre sí; el cliente puede ser utilizado, aunque el servidor web no esté disponible, pero no podrá obtener información para alimentar la aplicación. El servidor web puede funcionar sin el cliente, ya que únicamente se quedaría esperando peticiones.

2.5.1. Cliente

El cliente se encarga de ser una interfaz entre el usuario y el servidor web, el cual muestra al usuario la información organizada cuando se solicita. Debe ser intuitivo, ya que la mayoría de los usuarios no son expertos en informática.

Para el cliente del proyecto se utilizará una SPA (*Single Page Application*, por sus siglas en inglés), la cual cubre la interacción con el usuario y actualiza las funcionalidades del sistema a medida que el usuario lo solicite. De esta forma se actualiza la aplicación sin tener que recargar toda la página; es decir, se actualiza bajo demanda. Gracias a esto, el usuario puede seguir navegando en la página mientras el cliente recibe la información solicitada por el usuario. Es una aplicación mucho más fluida.

La herramienta elegida para desarrollar la aplicación para el cliente es Angular, la cual se detalla adelante en el documento.

2.5.2. Servidor web

El servidor web es el encargado de servir información al cliente. Contiene servicios y la lógica del negocio. De esta forma recibe, por medio de sus servicios, una petición del usuario; la procesa, dependiendo de lo que el usuario solicite, y envía una respuesta con la información.

El servidor web es el encargado de:

- Permitir el uso de los servicios solo a usuarios autenticados.

- Conectar con la base de datos de la aplicación. De esta forma es el servidor quién consulta y maneja la información.
- Velar por la seguridad de la información que se maneja en cada una de las peticiones que atiende.
- Permitir o denegar el acceso a ciertos host.

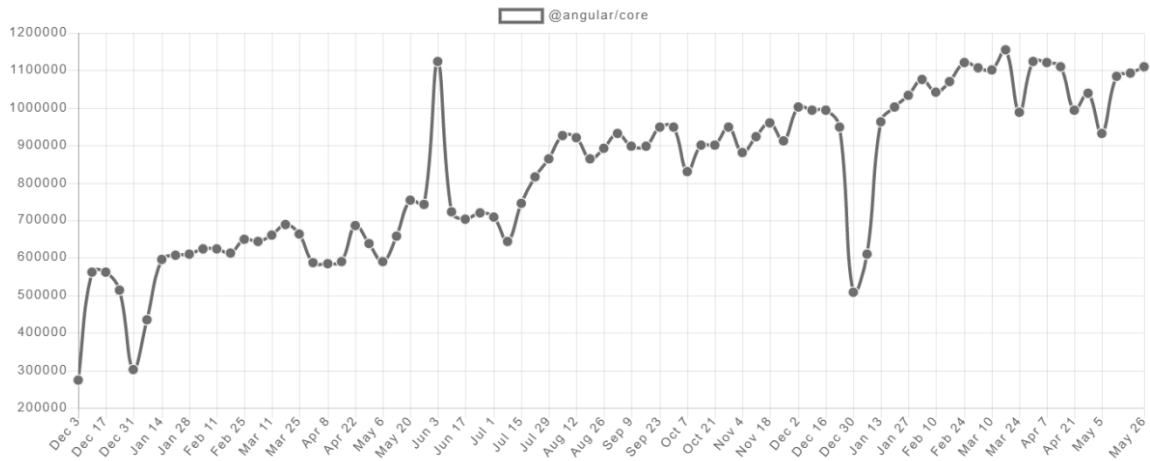
El servidor utilizado por el Departamento de Procesamiento de Datos es Weblogic, pero para el desarrollo del proyecto se decidió utilizar Node.js. Los motivos y características de este se presentan más adelante.

2.6. Angular

Es una plataforma y framework para construir aplicaciones de cliente en HTML y TypeScript que facilita el desarrollo de estas. Gracias a su reciente éxito y crecimiento provee una herramienta adecuada para el desarrollo de aplicaciones web, la cual se adapta a las necesidades del usuario de forma adecuada. Al ser Google quien da el soporte de Angular, cada versión que se libera ha sido probada en muchos proyectos de la misma empresa. Por tanto, al utilizar Angular optamos por una herramienta bastante robusta y suficientemente probada como para implementar la solución.

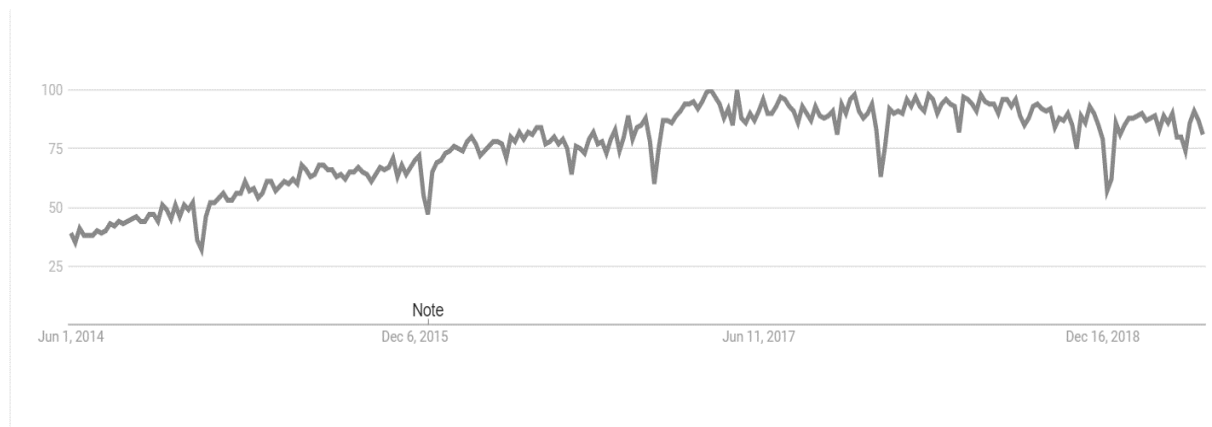
Se muestran dos figuras para ver el crecimiento que ha tenido Angular en los últimos años:

Figura 4. Descargas de Angular/core los últimos 2 años



Fuente: NPM trends. @angular/core. <https://www.npmtrends.com/@angular/core>. Consulta: 30 de mayo de 2019.

Figura 5. Interés a través del tiempo los últimos 5 años



Fuente: Google. *Angular Explore*.
<https://trends.google.com/trends/explore?cat=31&date=today%205-y&q=Angular>. Consulta: 30 de mayo de 2019.

2.6.1. Características de Angular

Angular nos ayuda como desarrolladores a entregar aplicaciones de calidad en menor tiempo, esto gracias a sus características *out-of-the-box*.

- Plataforma cruzada
 - Progressive Web Apps (PWA): se busca entregar experiencias parecidas a aplicaciones puras de dispositivo. Alto desempeño, uso fuera de línea y sin necesidad de instalación.
 - Nativo: se construyen aplicaciones móviles con estrategias de Apache Cordova, Ionic o NativeScript.
 - Desktop: para crear aplicaciones de escritorio entre Mac, Windows y Linux con los mismos métodos de Angular.

- Velocidad y desempeño
 - Generación de código: Angular genera las plantillas en código optimizado para las máquinas virtuales de JavaScript, dando los beneficios de código escrito a mano con la productividad de un framework.
 - Universal: se puede usar la aplicación en varios servidores como Node.js, .NET, PHP y otros.
 - Separación del código: carga rápido con un ruteo entre componentes que entrega una separación de código automática. De esta forma, los usuarios pueden cargar solo la vista que se necesita.

- **Productividad**
 - Plantillas (Templates): crear la interfaz de usuario con una plantilla de sintaxis simple y poderosa.
 - Angular CLI: de las mejores características, una línea de comandos para realizar acciones rápidamente como agregar componentes, crear un proyecto nuevo o desplegar la aplicación.
 - IDEs: obtiene predicción de sintaxis inteligente, errores instantáneos y feedback en editores populares e IDE's.
- **Desarrollo completo**
 - Pruebas: cuenta con herramientas específicas para ejecutar pruebas de forma rápida y estable.
 - Animación: crea coreografías y animaciones de alto desempeño y complejas con poco código con el API de Angular.

2.6.2. Angular Material

Es un set de componentes que ayudan al ágil desarrollo de aplicaciones web, creado y distribuido por Google como código abierto.

Sus principales características son:

- Crea la aplicación desde cero con componentes entendibles y con una UI moderna que funciona a través de la web, móviles y desktop.
- Rápido y consistente gracias a las múltiples pruebas que se han hecho a través de diferentes navegadores modernos.

- Versátil gracias a los temas que proveen; accesible e internacionalizado para dar la bienvenida a todos los usuarios nuevos.
- Optimizado por el equipo de Angular para Angular.
- Fácil de instalar gracias a los manejadores de paquetes para Node (NPM o Yarn) y fácil de usar.

2.6.3. ¿Por qué utilizar Angular para el proyecto?

Angular es uno de los frameworks más utilizado en los últimos años y los motivos por los que se decidió utilizarlo son los siguientes:

- Punto de vista del desarrollador
 - Es una herramienta con soporte por parte de Google, lo que permite utilizarla con total seguridad, ya que Google, antes de liberar una versión de su software, migra sus aplicaciones al mismo.
 - Tiene un ecosistema muy grande. Se pueden encontrar muchos componentes hechos por otras personas para implementar fácilmente.
 - Provee de una línea de comandos que facilita muchas de las acciones recurrentes como desarrollador.
 - Aunque no cuenta con un IDE específico, los editores de texto tienen complementos que son de utilidad para el desarrollo de aplicaciones de Angular.

- Permite desarrollar más rápido gracias a su generación de código, por lo que como desarrollador es más fácil hacer más con menos líneas de código.
- Punto de vista del usuario
 - Se tiene una aplicación atractiva y con diseños atractivos gracias a Angular Material.
 - Angular da una experiencia de usuario fluida, ya que muestra información de forma dinámica.
 - Se puede hacer uso de la aplicación sin tener equipo de última generación, ya que se ejecuta sobre el navegador.

Gracias a la característica universal de Angular se puede realizar un cambio dentro del servidor habitual de las aplicaciones en el Departamento de Procesamiento de Datos. Se coloca así al proyecto como un punto de inflexión para las futuras aplicaciones, con tecnología actualizada y a la vanguardia tecnológica.

2.7. Node.js

Node.js es un entorno de ejecución de JavaScript construido sobre el Chrome's V8 JavaScript engine. Es apreciado grandemente por su excelente desempeño y escalabilidad. De acuerdo con una encuesta realizada en el 2017 por la fundación de Node.js, actualmente es la principal herramienta utilizada en los servidores web, aplicaciones empresariales, soluciones de Big Data y sistemas embebidos.

Aunque Java es utilizado por millones de desarrolladores y es uno de los lenguajes más populares de los últimos años, JavaScript mantiene el título del lenguaje mayormente liberado en el mundo, junto con Node.js. De hecho, empresas como eBay, LinkedIn, Uber y Netflix han preferido usar Node.js sobre toda la gama de herramientas disponibles hoy para desarrollar sus aplicaciones web.

2.7.1. Java y Node.js

Java es propósito general, orientado a objetos, independiente de plataforma, portable y un lenguaje de programación interpretado robusto. Puede correr sobre cualquier máquina virtual de Java, lo cual les da flexibilidad a los desarrolladores. Podemos encontrarlo en aplicaciones web y móviles para empresas, pero también en juegos, sistemas embebidos e incluso en robótica.

A continuación, se describen las ventajas y desventajas de cada tecnología:

2.7.1.1. Ventajas Java

- Alta disponibilidad a los servicios que provee.
- Mantiene la integridad transaccional.
- Consultas paralelas de datos mediante el uso de filtros.
- Se integra fácilmente con base de datos de Oracle.
- Amplia gama de IDEs para su desarrollo, como NetBeans o Eclipse.
- Eficiencia computacional.
- Gran mantenibilidad de código.

2.7.1.2. Desventajas Java

- Su curva de aprendizaje es alta.
- Consumo de memoria RAM es excesivo.
- Si hay un bloqueo, el servidor se bloquea.
- Aplicación web de alto rendimiento y consumo de ancho de banda. Las aplicaciones resultantes necesitan de un buen ancho de banda para renderizarse con agilidad en los navegadores.
- Las aplicaciones web tardan un poco más en renderizar el contenido, por lo que su consumo de ancho de banda es alto en comparación con otros frameworks.
- Problemas para migrar si se quiere cambiar la lógica del negocio al navegador.

2.7.1.3. Ventajas Node.js

- Se puede ejecutar en una variedad de servidores, entre los que destacan Microsoft Windows, Mac OS X y Unix.
- Se ha utilizado en LinkedIn, eBay o PayPal.
- Facilidad para aprendizaje.

- Es perfecto para utilizarlo en aplicaciones con alto tráfico de usuarios y eventos.
- Utiliza el motor de JavaScript V8 de Google.
- Es útil para aplicaciones que necesitan mantener una conexión persistente entre el navegador y el servidor.
- Procesa grandes cantidades de datos de forma rápida y confiable.
- Análisis de errores, práctico y rápido despliegue de código.
- Constante mejora tecnológica debido a la comunidad de código abierto.
- Tiene una amplia comunidad que provee distintas librerías que permiten realizar un desarrollo web muy enriquecido.
- Está en constante mejora.
- Incluye una característica de reutilización de código, lo que aumenta grandemente la productividad de un desarrollador.
- Ofrece un estilo reactivo out-of-the-box, lo cual es muy complicado para Java.

2.7.1.4. Desventajas Node.js

- No hay muchos IDEs especializados, pero no es una desventaja considerable tomando en cuenta el avance que se tiene en editores de texto, por ejemplo, Visual Studio Code.
- El ecosistema de JavaScript no es tan robusto hasta el momento como el de Java, pero está en constante crecimiento.

Debido a que se necesita que la aplicación sea lo más rápida posible y no soporta acciones tan pesadas, se decidió utilizar Node.js sobre Java. Utilizará menos ancho de banda y memoria RAM; no bloquea los procesos y ayuda al desarrollador a reutilizar el código y aumentar su productividad. De esta forma se tendrá una aplicación rápida, liviana, de código abierto y con una comunidad en constante crecimiento.

2.7.2. Express.js para Node.js

Node.js es una combinación de librerías y herramientas que nos ayudan a hacer un servidor web.

Uno de los frameworks de Node.js más importantes es Express.js, el cual es un framework rápido y flexible. Provee una robusta cantidad de características para aplicaciones web y móviles. Crear un servidor web utilizando estas dos herramientas es muy fácil. Un servidor mínimo con 'Hola Mundo' tendría la siguiente estructura:

Figura 6. **Servidor web mínimo con Node.js y Express.js**

```
const express = require('express')
const app = express()
const port = 3000

app.get('/', (req, res) => res.send('Hola Mundo!!!'))

app.listen(port, () => console.log(`Servidor corriendo en el puerto ${port}
`))
```

Fuente: Express. *Hello World example*. <https://expressjs.com/en/starter/hello-world.html>.

Consulta: 25 de marzo de 2020.

2.7.3. Base de datos

La base de datos utilizada para el proyecto es Oracle, debido a que el Departamento de Procesamiento de Datos utiliza dicho motor de base de datos. Los requerimientos se pueden atender con una base de datos relacional, ya que no estamos hablando de cantidades considerables de datos como para una base no relacional.

Se antepone 'AC_' al nombre de las tablas para identificar que se trata de la aplicación de Actividad Comercial dentro del esquema del Departamento. Los procedimientos almacenados están en un paquete llamado ACTIVIDAD_COMERCIAL, para ayudar al orden dentro del esquema de base de datos.

2.8. Aplicación utilizada anteriormente

La aplicación desarrollada previamente fue implementada por el Departamento de Procesamiento de Datos. Estaba hecha en Oracle forms y atendía solo parte del proceso.

Figura 7. Pantalla de aplicación anterior

The screenshot displays two data tables from an Oracle Forms application. The first table, 'Posicionamiento', has columns for 'Id' and 'Descripción'. The second table, 'Forma de Distribución', has columns for 'Id', 'Tipo Destino', and 'Porcentaje %'. A 'Total %' field at the bottom right shows the value 100.

Posicionamiento	
Id	Descripción
1	EN EL AREA DE INFLUENCIA
2	FUERA DEL AREA DE INFLUENCIA

Forma de Distribución		
Id	Tipo Destino	Porcentaje %
2	BECAS	30
5	BIBLIOTECA CENTRAL	10
6	DBEU	5
8	LABORATORIO MICROBIOLOGICO DE ALIMENTOS	20
64	DIGA	35

Total % 100

Fuente: aplicación desarrollada por el Departamento de Procesamiento de Datos para el control de cuenta corriente y distribución de ingresos.

Esta aplicación tenía funcionalidades limitadas al registro de arrendatarios y contratos, lo cual dejaba de lado el proceso que se lleva a cabo y las variantes

de este, así como toda la configuración sobre los locales y el tipo de arrendatario. Se enfoca más en la parte de presupuesto, que se encarga de la distribución de los pagos efectuados por los arrendatarios. La nueva aplicación tomará un flujo más completo del proceso que incluye desde la configuración de los tipos de local y arrendatario, hasta la distribución de los pagos realizados a lo largo del tiempo.

2.8.1. Aplicación de nuevas herramientas

Las nuevas herramientas proveen de un aire fresco al diseño de la aplicación, velocidad y diseño agradable para el usuario, ya que se enfoca en la interfaz de usuario y su experiencia. Es como un set de servicios que el cliente va a consumir y que están desacoplados; es decir, la aplicación funciona si un servicio no está trabajando. Esto mantiene la aplicación disponible a pesar de que una parte de ella no funcione correctamente, para no detener por completo el uso de esta.

Con una aplicación web, los usuarios podrán ingresar sin tener que instalar algo en sus computadoras. Estará disponible para cualquier computadora que tenga conexión a Internet.

Como consideración, se tomaron los reportes generados por la aplicación hecha en Oracle forms para la nueva, ya que son muy útiles para los usuarios y los diseños son efectivos para la función que cumplen.

2.9. Manejo de pagos actual en la USAC

El proceso para manejo de pagos es el siguiente:

- El arrendatario que ya tienen un contrato se dirige a Caja Central de la Universidad de San Carlos de Guatemala a efectuar su pago.

- Una vez efectuado el pago, debe ir al Departamento de Presupuesto para que sea registrado si no lo está, y poder recolectar el pago.
- Luego, el arrendatario debe ir con su recibo de pago a la oficina de Actividad Comercial para que se actualice su estado en las hojas de cálculo.
- En caso el usuario no va a entregar el recibo a la oficina de Actividad Comercial, mensualmente se envía a dicha oficina una lista con los pagos de los arrendatarios para que se actualicen las hojas de cálculo.

Dicho proceso se debe hacer para todos los negocios de la Universidad y cada mes que se efectúa un pago. Además, los centros universitarios deben realizar el proceso en el campus central, lo cual dificulta mucho a las personas efectuar sus pagos.

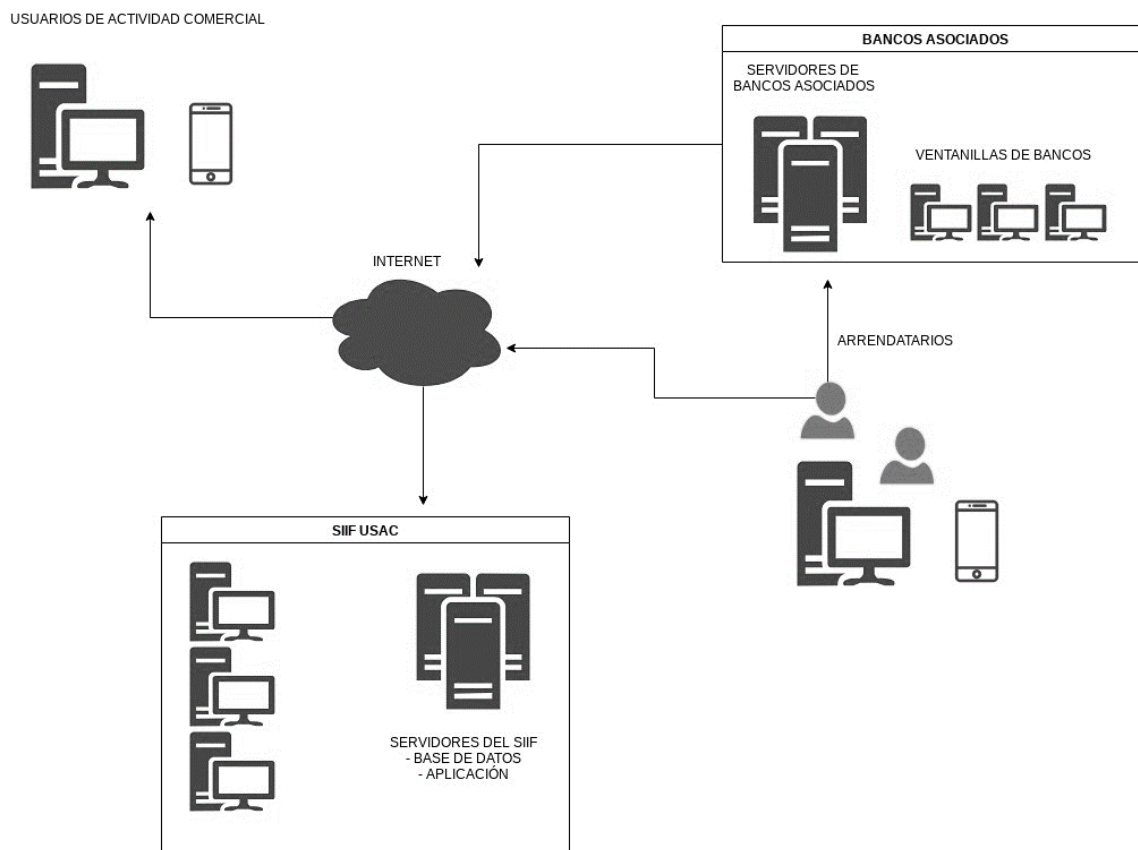
Con la aplicación se busca reducir el proceso de pago y tener un mejor manejo de estos. Los usuarios que quieran continuar su pago en la Caja Central de la Universidad pueden hacerlo y se registra dentro de la aplicación. Con la funcionalidad nueva se puede generar una boleta para pagar en cualquier banco o banca virtual afiliada. El sistema actualizará solo el estado del arrendatario. Optimiza el tiempo para las personas de la oficina de Actividad Comercial y los arrendatarios.

2.10. Preliminares de la aplicación

Una vez definidas las herramientas a utilizar se detalla el desarrollo que se llevó a cabo. Se adjunta el diagrama de casos de uso de la aplicación en el apéndice 2 del presente documento.

La aplicación sigue la siguiente arquitectura:

Figura 8. **Arquitectura de la solución**



Fuente: elaboración propia, empleando draw.io.

A partir de una arquitectura basada en el estándar para aplicación web, se tiene como clientes a los usuarios de Actividad Comercial y arrendatarios, y el servidor web dentro de los servidores del Departamento de Procesamiento de Datos. La conexión con los bancos asociados existía previamente por parte del SIIF, por lo que la aplicación únicamente consume los servicios existentes que hacen referencia a pagos en los bancos.

2.11. Servidor del proyecto

Con el manejador de paquetes de Node.js (npm) se genera una estructura básica para el proyecto, que es una distribución de archivos, de la siguiente manera:

Figura 9. Estructura de archivos del servidor

```
> middlewares
> node_modules
> routes
JS admin_server.js
{} package-lock.json
{} package.json
① README.md
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

La explicación es la siguiente:

- `node_modules`: es la carpeta donde se guardan todos los módulos de Node.js que el servidor está utilizando, por ejemplo, `Express.js`.

- package.json es un archivo en el que se guarda la información del servidor:
 - Nombre
 - Versión
 - Descripción
 - Archivo que servirá como principal
 - Scripts que se ejecutan desde la línea de comando con npm:

Figura 10. **Scripts del servidor**

```

"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "server": "nodemon admin_server",
  "client": "npm start --prefix admin",
  "dev": "concurrently \"npm run server\" \"npm run client\""
}

```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Entonces, al ejecutar en la consola 'npm server' se ejecuta el equivalente en el script, el cual es 'nodemon admin_server'. Este inicia el servidor y lo reinicia cada vez que se detecta un cambio en los archivos.

- La configuración del repositorio, la cual es opcional
- Autor
- Licencia
- Página de inicio, que para el proyecto es el README del repositorio
- Lista de las dependencias con sus respectivas versiones

El archivo package.json es muy importante, ya que nos ayuda a construir la aplicación. Una situación en la que resulta útil es cuando tenemos el

servidor en un repositorio. Ya que la carpeta de `node_modules` ocupa un espacio considerable, no se sube; entonces, al clonar el proyecto no tiene los módulos necesarios para su funcionamiento. Para solucionar esto, gracias al `package.json`, solo necesitamos escribir en la consola el comando `'npm install'` y se instalan todas las dependencias especificadas dentro del archivo.

- `package-lock.json`: es generado automáticamente cada vez que hay cambios en `node_modules` o `package.json`. Describe los árboles de dependencias, lo que nos permite instalar exactamente las mismas dependencias utilizadas en el proyecto.

2.11.1. Archivo del servidor (`admin_server.js`)

Este es el archivo en donde se crea específicamente el servidor web. Como se mencionó, se hizo uso de `Node.js` y `Express.js` para crearlo, con la siguiente estructura:

- Encabezado del servidor: en el encabezado del servidor se definió la cantidad de `workers` con la ayuda de `throng`, que es un módulo que ayuda a ejecutar varios `workers` al mismo tiempo y maneja la concurrencia a la aplicación.

Figura 11. Encabezado del servidor

```
var throng = require('throng');

var WORKERS = process.env.WEB_CONCURRENCY || 1;
var PORT = process.env.PORT || 8081;

throng({
  workers: WORKERS,
  lifetime: Infinity,
  start: start
});
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

- Función de inicio: dentro de esta se crea el servidor como tal, que throng va a ejecutar y manejar. Con Express.js se configura el servidor de la siguiente forma:
 - Configura el puerto que se va a utilizar para el servidor y que reciba peticiones http.
 - Permite el control de acceso a todos los orígenes, ya que los arrendatarios podrán ingresar a la aplicación.
 - Envía mensajes en formato json, debido a su facilidad de uso y flexibilidad.
 - Para las peticiones con el formato `‘/api/actividadcomercial/’` se hace pasar por un middleware para validar la petición, cual será explicado en la siguiente sección.
 - Para las peticiones en general se debe utilizar los archivos que están dentro de la carpeta routes.

- Utilizar los archivos estáticos que se encuentran en la carpeta dist con ayuda de Express.js. Esto sirve para ejecutar Node.js y que el servidor sea quien sirva el cliente.

2.11.2. Funciones middleware

Aquí se ubican los archivos que tienen acceso a las peticiones que llegan al servidor; es decir, que pueden manipular la información antes que llegue a la función que va a atender la petición.

Dentro de esta carpeta se encuentra el archivo 'validate_request.js', que se encarga de verificar que las peticiones tengan un token de autorización para llegar al servicio como tal.

2.11.2.1. Autenticación por token

La función exportada por validate_request.js es la que se encarga de verificar que la petición tenga un token válido para atender la petición. Se utilizan tokens encriptados con JWT (JSON Web Tokens). La encriptación del token necesita de dos parámetros:

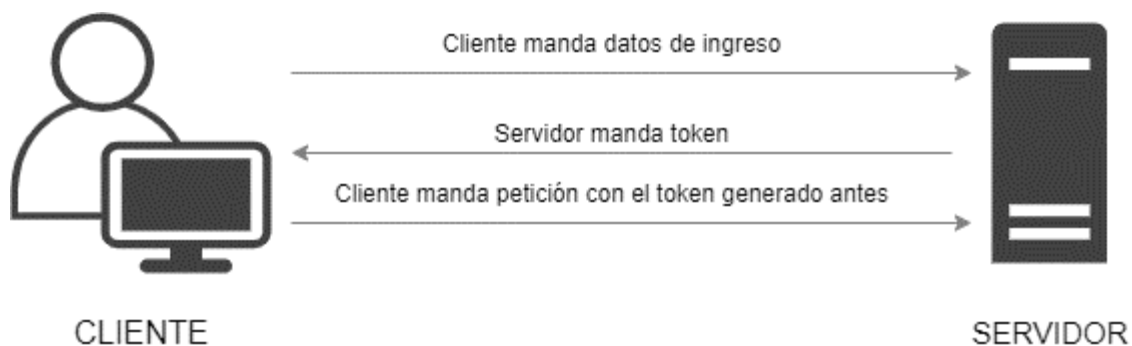
- Objeto para encriptar en formato JSON. Se puede encriptar cualquier objeto que se desee.
- Llave para encriptar y desencriptar, la cual se define para el servidor.

La autenticación por token es muy similar a la autenticación basada en cookies. No hay una diferencia notable en realidad, pero como desarrolladores es importante conocer las nuevas tecnologías que se están utilizando. Por tanto,

se decidió utilizar tokens sobre cookies, aunque no represente una ventaja técnica significativa. La forma en la que funciona la autenticación es la siguiente:

- El cliente intenta ingresar a la aplicación y envía una petición al servidor.
- Si el ingreso es correcto, el servidor genera un token para el cliente y lo envía de regreso. Si no es correcto el ingreso, se advierte al cliente que el usuario o contraseña son incorrectos.
- El cliente almacena el token para enviarlo adjunto con sus futuras peticiones.
- El servidor, cada vez que recibe una petición del cliente, ve si se envía un token válido en los encabezados de esta. De ser así, deja que se consuman los servicios. Si no se cuenta con un token válido no se permite hacer uso de los servicios.

Figura 12. **Autenticación por token**



Fuente: elaboración propia, empleando draw.io.

Cada vez que se hace una petición, esta pasa por la validación. Se descripta el token enviado y se toma la información necesaria para las peticiones; por ejemplo, el identificador del usuario. De esta forma evitamos al cliente tener que enviar dicho identificador en el cuerpo de la petición, ya que el token contiene dicha información.

2.11.3. Servicios

Dentro de la carpeta routes se encuentran varios archivos de JavaScript, los cuales tienen dentro los servicios y la lógica del negocio. El archivo raíz es 'index.js' y ahí se definen las rutas y funciones que atenderán las peticiones.

El archivo index, en su parte superior, define el router que va a dirigir las peticiones y variables que tienen el set de funciones para atender las mismas, de la siguiente forma:

Figura 13. Parte superior router>index.js

```
routes > JS index.js > ...
 1  var express = require('express');
 2  var router = express.Router();
 3
 4  var queryexec = require('./ejecutor_ora.js')();
 5  var auth = require('./auth.js')(queryexec);
 6  var requisito = require('./requisito.js')(queryexec);
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Como se puede observar, primero se crea una variable que requiere el módulo Express.js y otra para utilizar el router que provee Express.js

Luego se importa de 'ejecutor_ora.js', dentro del cual están las funciones para manipular la base de datos. Dado que los demás archivos exportan una función, se les manda como parámetro la variable queryexec, para que dichos archivos puedan hacer uso de las funciones de base de datos.

La estructura básica de los archivos donde se tienen los servicios es la siguiente:

Figura 14. **Estructura de servicios**

```
const oracledb = require('oracledb');

module.exports = function (queryexec) {
  var ENTIDAD = {

    CREAR: function(req, res){
      // Acciones a realizar
    },

    EDITAR: function(req, res){
      // Acciones a realizar
    }

  }

  return ENTIDAD;
}
```

Fuente: elaboración propia.

De esta forma se tiene modulada cada acción a ejecutar en la base de datos. De la misma forma se puede atender una petición con una función específica. Esta es la facilidad que da JavaScript al exportar una función que retorna un objeto JSON con funciones dentro.

Una vez hechas las funciones para atender las peticiones se debe configurar el router para que sepa qué función atenderá qué ruta, de la siguiente forma:

Figura 15. **Estructura del router**

```
// ruta para LOGEOS no necesita token como las demás
router.post('/api/auth', auth.LOGIN);
router.post('/api/autharrendatario', auth.ARRENDATARIO);
router.post('/api/out/infoarrendatario', arrendatario.INFO_ARRENDATARIO);
router.post('/api/out/pagosdeudor', arrendatario.PAGOS_DEUDOR);

// ruta para BITACORA
router.post('/api/actividadcomercial/bitacora/logout', bitacora.LOGOUT);
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

El router necesita dos parámetros:

- Endpoint al que se hace la petición. Los endpoint con formato '/api/actividadcomercial/*' pasaron por la validación de petición.
- Función que atenderá la petición. Las funciones reciben por defecto los parámetros req y res, que son request y response, respectivamente. Request es la petición y response la respuesta que el servidor está enviando.

2.11.4. Autenticación

La autenticación de la aplicación hace una distinción entre arrendatarios y todos los demás roles que son trabajadores de la Universidad. Para los

trabajadores se autentica mediante un servidor LDAP del Departamento de Procesamiento de Datos y a los arrendatarios, por medio de base de datos.

2.11.4.1. Autenticación de trabajadores

La autenticación para los trabajadores es diferente a los arrendatarios, porque estos se registran en un servidor LDAP propio de la Universidad. Todo el manejo de dicho servidor es responsabilidad del Departamento de Procesamiento de Datos.

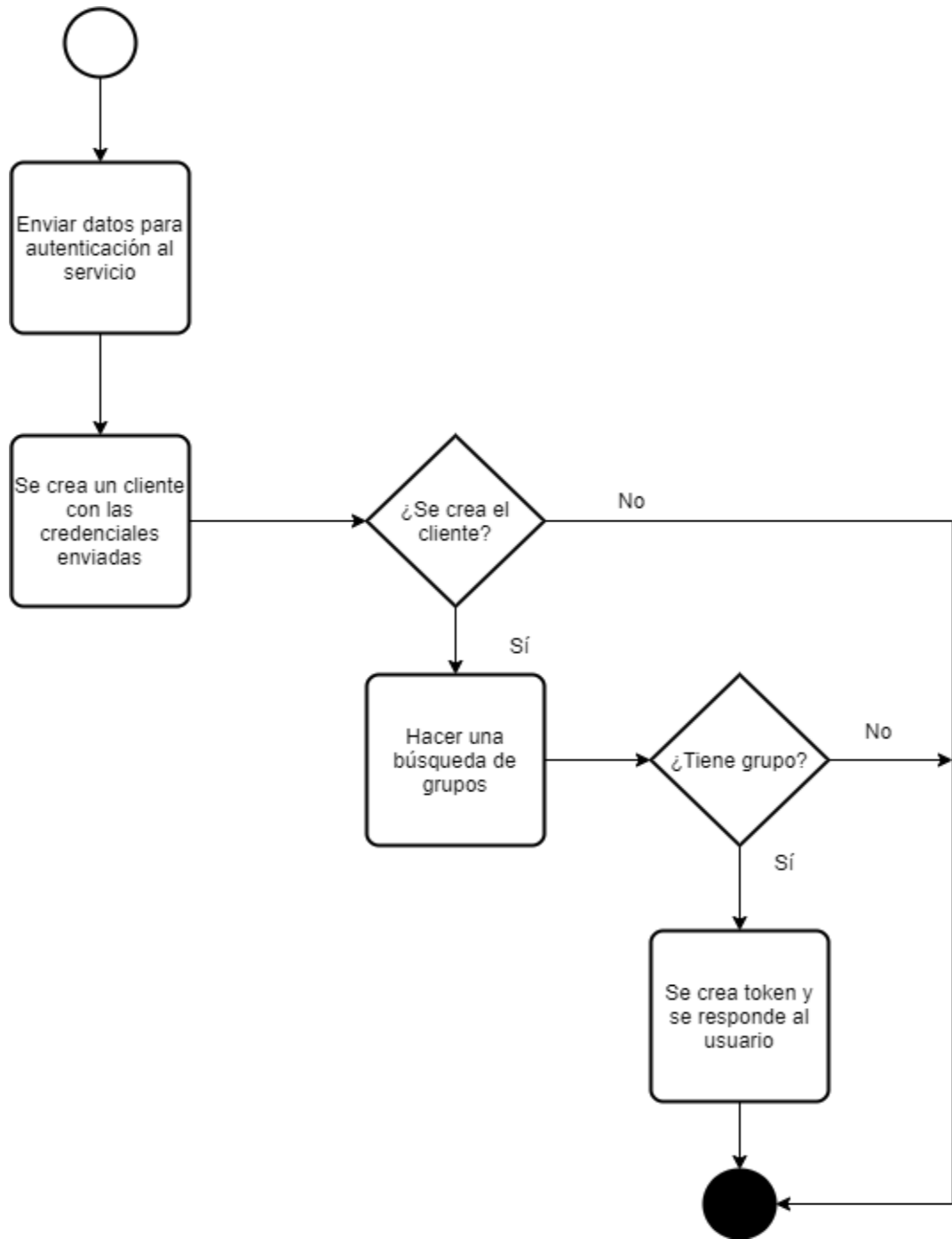
Los usuarios ingresan al LDAP con su registro de trabajador y una contraseña. Si desean ingresar nuevos usuarios o se tiene algún problema se debe contactar con el Departamento de Procesamiento de Datos.

Los roles para la aplicación son los siguientes:

- actividad_comercial_admin para usuarios administradores de Actividad Comercial.
- actividad_comercial_operador para usuarios regulares de Actividad Comercial.
- actividad_comercial_distrib para usuarios del Departamento de Presupuesto.

El procedimiento para autenticar con el servidor LDAP es el siguiente:

Figura 16. Autenticación usando LDAP



Fuente: elaboración propia, empleando draw.io.

2.11.4.2. Autenticación de arrendatarios

Los arrendatarios se autentican por medio de la base de datos, como se dijo. Debido a la falta de conocimientos tecnológicos de algunos usuarios se definió un usuario y contraseña para los arrendatarios, cuyos valores son:

- Usuario: número de identificación (ej. DPI).
- Contraseña: número de teléfono celular registrado en la base de datos.

De esta forma, los usuarios no deben preocuparse por su usuario y contraseña, ya que estos no van a cambiar a menos que el mismo usuario solicite el cambio en la oficina de Actividad Comercial.

Para almacenar las contraseñas de forma segura se utilizó un encriptado de cadenas de texto llamado bcrypt.

2.11.4.2.1. Bcrypt

Fue diseñado por Niels Provos y David Mazieres basado en Blowfish; de ahí proviene su nombre.

Bcrypt es un algoritmo que aún no ha sido roto, por lo que actualmente es de las mejores opciones para encriptar. Aunque el costo computacional es mayor a otros algoritmos, es un intercambio justo para mantener la seguridad de la aplicación.

Para utilizar bcrypt en Node.js existe un módulo llamado bcrypt y se puede instalar usando npm. Una vez instalado se puede utilizar de la siguiente forma:

Figura 17. **Uso de bcrypt**

```
const bcrypt = require('bcrypt');
const saltRounds = 10;
const hash = bcrypt.hashSync(myPlaintextPassword, saltRounds);
bcrypt.compareSync(myPlaintextPassword, hash);
```

Fuente: NPMJS. *Bcrypt*. <https://www.npmjs.com/package/bcrypt>. Consulta: 25 de marzo de 2020.

HashSync y compareSync son métodos síncronos, pero el módulo tiene métodos asíncronos equivalentes por si el usuario necesita hacer algo mientras se está encriptando; estos son hash y compare, y reciben como tercer parámetro un callback.

Como observación: no se puede hacer comparaciones normales entre cadenas de texto encriptadas; es decir, no es efectivo realizar un `cadena1 = cadena2`. Se debe utilizar los métodos del módulo para comparar.

2.11.5. Bitácora

El Departamento de Procesamiento de Datos solicitó el ingreso de actividad en una bitácora que ellos crearon previamente, dentro de la cual se registra el inicio de sesión de una persona, las acciones que realiza sobre las tablas (INSERT, UPDATE, DELETE) y cerrar sesión; todo esto para un mejor control de la información que se manipula en la aplicación.

La bitácora consta de tres tablas:

- BITACORA_ACCESO: es donde se registra todo el acceso a la aplicación. Se debe identificar a la persona, el tipo de usuario, la aplicación que usa y la dirección IP. Se genera un registro con la información enviada y la hora del inicio de sesión.
- BITACORA_TABLA: aquí se registran las acciones que se llevan a cabo dentro de una tabla. Se envía el identificador del acceso junto con el nombre de la tabla que se utiliza y la acción.
- BITACORA_COLUMNNA: en conjunto con la tabla anterior registra los campos nuevos, modificaciones y eliminaciones para las columnas de una tabla. Es un detalle de los registros que forman parte de la tabla.

Se debe registrar 5 acciones dentro de la bitácora: acceso, insert, update, delete y salida, para lo cual hay algunas funciones hechas por el Departamento de Procesamiento de Datos. Se encuentran en el paquete SIIF.BITACORA_PK y son las siguientes:

- registrarAcceso(pUsuario in varchar2, pTipo_Usuario in varchar2, pAplicacion in varchar2, plp_Cliente in varchar2,pld_Bitacora out number). Registra que un usuario ha iniciado sesión en la aplicación. Toda la actividad que haga dentro de las tablas del esquema será registrada para control del Departamento, donde:
 - pUsuario es el registro del usuario que está ingresando.
 - pTipo_Usuario es el tipo de usuario que está ingresando.
 - pAplicacion es la aplicación que está haciendo el registro; en este caso es 'AC', por las siglas de Actividad Comercial.
 - plp_Cliente es la dirección ip del usuario que está ingresando.

- pld_Bitacora es el identificador de la sesión que se acaba de registrar. Posteriormente se guarda en el token para que, cada vez que se realice una operación en la bitácora, la petición tenga en su cuerpo este identificador.
- registrarLogout(pld_Bitacora in number). Registra que el usuario ha cerrado sesión de la aplicación, donde:
 - pld_Bitacora es el identificador obtenido con registrarAcceso y que está en el token del usuario.
- registrarEventoTabla(pTabla in varchar2,pOperacion in varchar2, pld_Bitacora in number, pld_Bitacora_Tabla out number). Registra la actividad de una tabla en específico, realizada por un usuario donde:
 - pTabla es el nombre de la tabla que se está utilizando.
 - pOperacion es la acción que se está realizando (INSERT, UPDATE, DELETE).
 - pld_Bitacora es el identificador obtenido con registrarAcceso y que está en el token del usuario.
 - pld_Bitacora_Tabla es el identificador del registro que se acaba de ingresar a la tabla.
- registrarEventoColumna(pld_Bitacora_Tabla IN number, pColumna in varchar2, pValor_Anterior in varchar2, pValor_Nuevo in varchar2). Que registra la actividad para la columna de una tabla en específico donde:
 - pld_Bitacora_Tabla es el identificador obtenido con registrarEventoTabla.

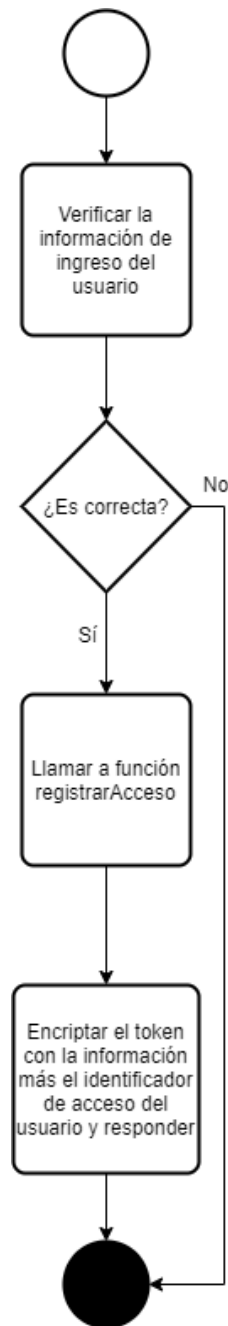
- pColumna es el nombre de la columna que se está utilizando.
- pValor_Anterior es el valor anterior que tenía la columna.
- pValor_Nuevo es el valor que acaba de tomar esta columna.

Para registrar la actividad en la bitácora, en la carpeta routes está el archivo 'bitácora.js', dentro del cual se centralizan las operaciones que se ejecutan en la bitácora.

2.11.5.1. Registrar acceso del usuario

Para registrar el acceso se envía la información necesaria a la función. Al terminar, regresa en una variable el identificador del registro de acceso para que el servidor pueda encriptarlo en el token.

Figura 18. **Registrar acceso del usuario**

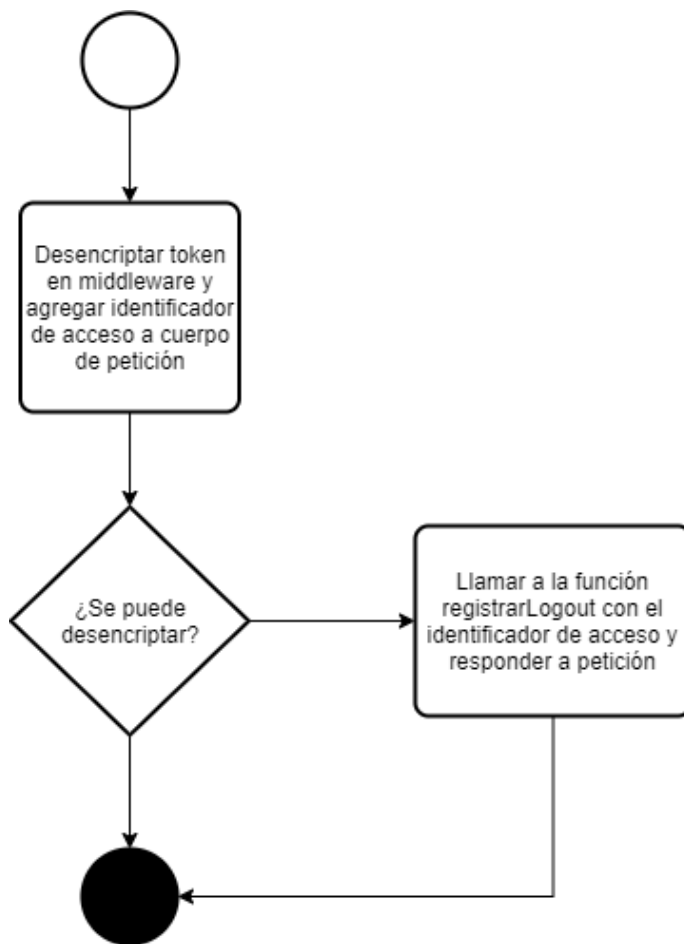


Fuente: elaboración propia, empleando draw.io.

2.11.5.2. Registrar salida del usuario

Se creó un servicio específico para registrar la salida del usuario que recibe la petición con el identificador de acceso generado. Se utiliza la función registrarLogout especificada anteriormente.

Figura 19. Registrar salida del usuario

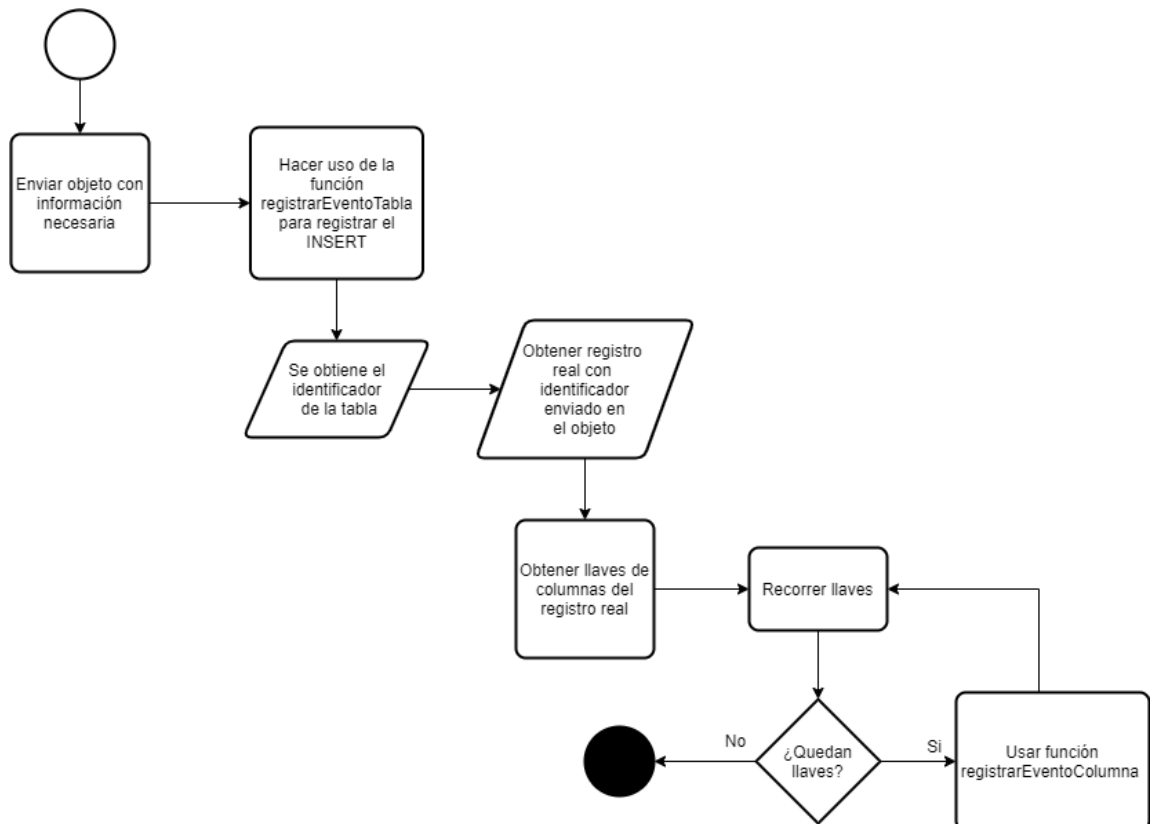


Fuente: elaboración propia, empleando draw.io.

2.11.5.3. Registrar inserción de registro

Para registrar actividad en una tabla se debe esperar a que se realice la operación. Luego, la bitácora obtiene la información que se acaba de registrar con una consulta por medio del identificador del registro. Una vez se tiene la información, se recorre e inserta en la tabla para registrar columnas.

Figura 20. Registrar inserción de registro

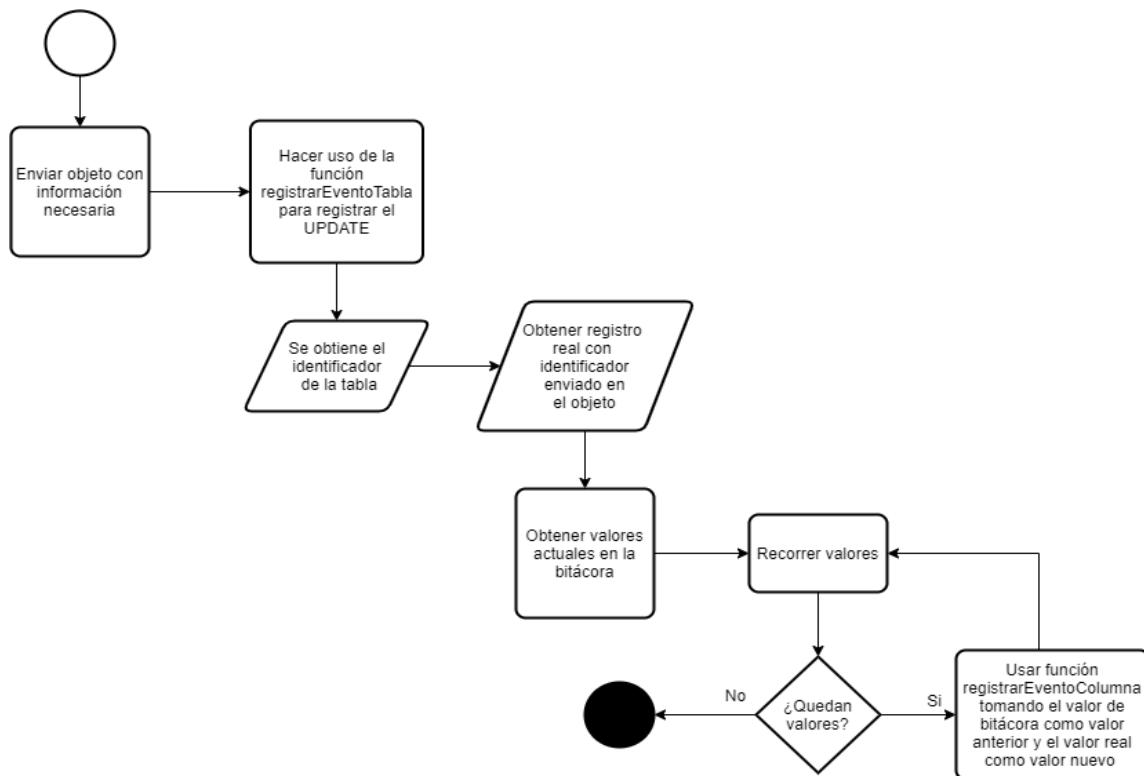


Fuente: elaboración propia, empleando draw.io.

2.11.5.4. Registrar modificación de registro

La modificación tiene un poco más de trabajo. Se debe esperar a que se realice la operación. Luego, la bitácora obtiene la información que se acaba de modificar con una consulta por identificador del registro. Una vez que se posee la información se debe tener los últimos valores que han sido insertados en la tabla para columnas y verificar si hay algún cambio con el registro real obtenido. Si el valor de la columna del registro es distinto, se hace una inserción del identificador del registro seguido de las columnas con valor nuevo.

Figura 21. Registrar modificación de registro

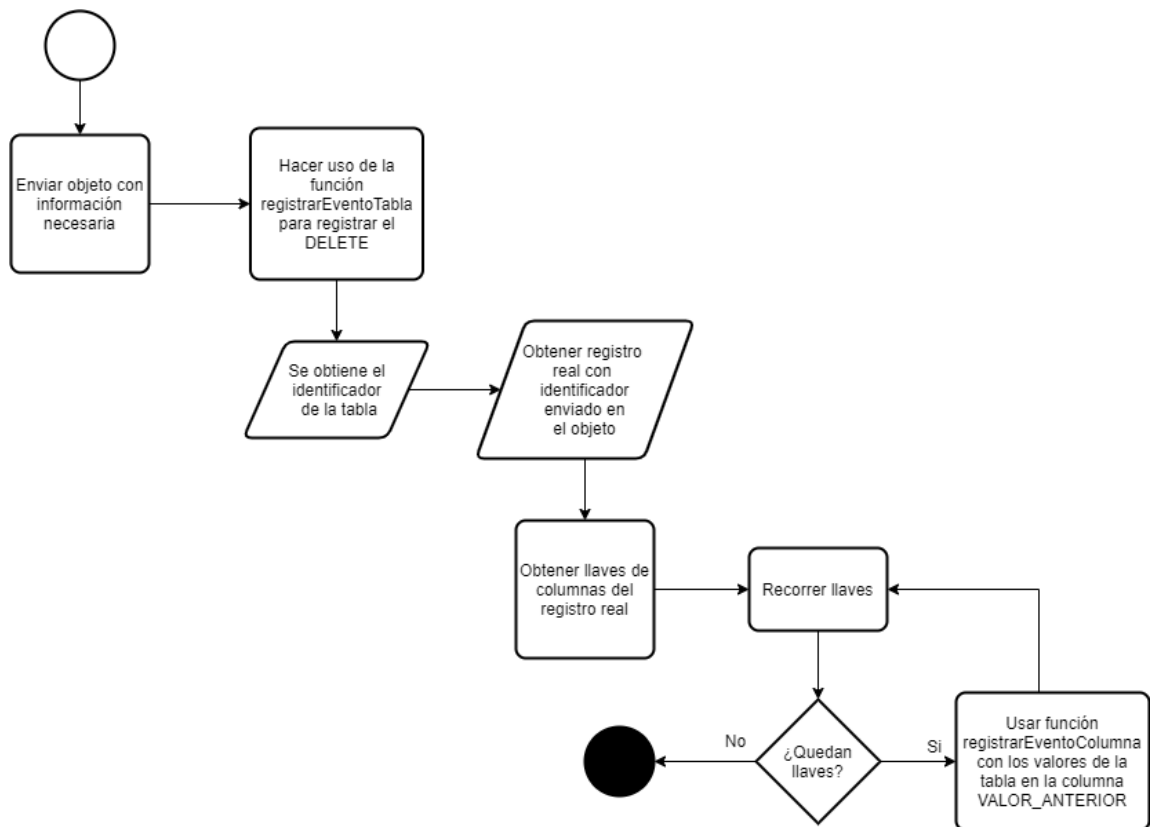


Fuente: elaboración propia, empleando draw.io.

2.11.5.5. Registrar eliminación de registro

Para eliminar un registro, luego que se eliminó en las tablas reales se registra el identificador de este en la tabla de columnas.

Figura 22. Registrar eliminación de registro



Fuente: elaboración propia, empleando draw.io.

2.12. Cliente del proyecto

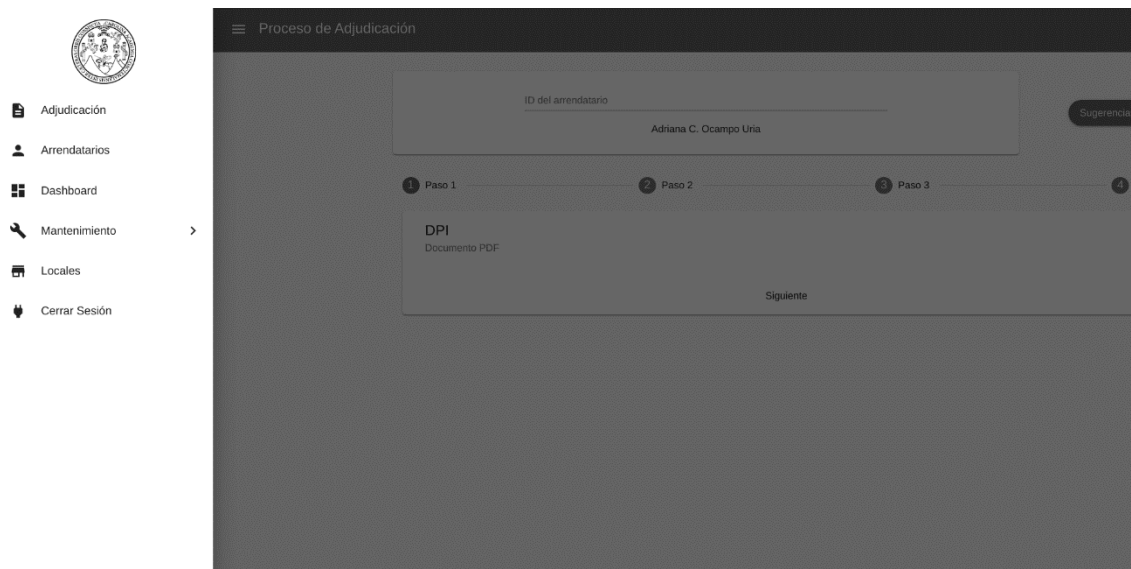
Para crear el cliente base se utilizó Angular CLI y se incorporó el módulo para ruteo con la bandera –routing con Angular 6, ya que era la versión estable cuando se inició el proyecto.

2.12.1. Diseño de la aplicación

Con ayuda de Angular Material y conceptos de diseño minimalista se mejoró la interfaz de usuario en comparación a los prototipos propuestos. Con base en Gmail, se logró un diseño con fondos blancos y detalles azules.

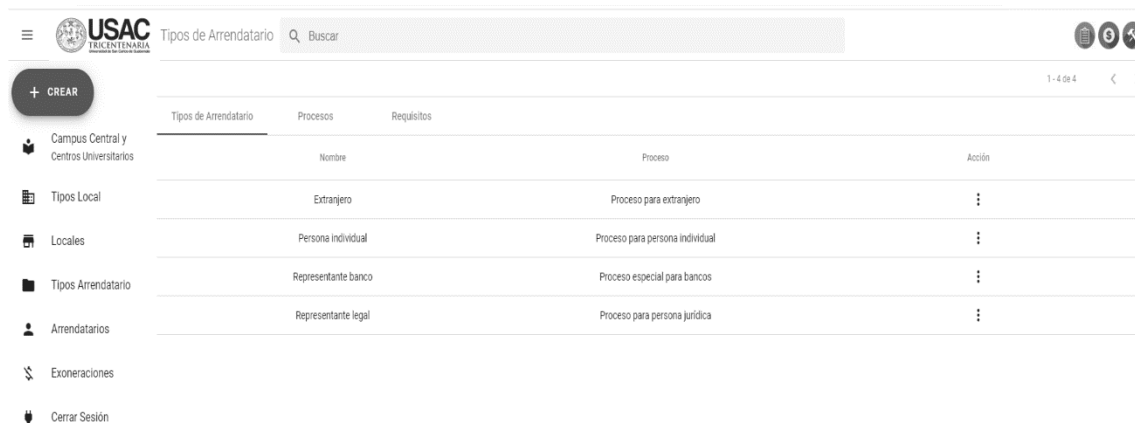
En las siguientes dos figuras se muestra el diseño de los prototipos y el diseño que se definió para la aplicación final:

Figura 23. Diseño para menú del prototipo



Fuente: prototipo desarrollado para el módulo de Actividad Comercial.

Figura 24. **Diseño para menú de la aplicación**



Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Como se puede observar en las figuras anteriores, el prototipo fue desarrollado con los componentes de Angular Material sin cambio alguno. El menú tenía la opción de ser mostrado cuando el cursor estaba cerca del icono, lo cual podría resultar molesto, ya que se bloqueaba el resto de la ventana. Caso contrario con el diseño final, que ofrece dos opciones para mostrar el menú: al acercarse a los iconos de este, se muestra el menú sobre la ventana que se encuentra, sin bloquearla. La otra opción es para anclar el menú completo; es decir, el menú se queda estático con sus opciones, tal y como se muestra en la figura 24.

De las ventajas del diseño minimalista es que se muestra la información de forma directa y simple, con menos énfasis en los efectos de la aplicación, los cuales ocupan tiempo y procesamiento.

Figura 26. **Diseño de la aplicación para áreas de influencia**

EN EL AREA DE INFLUENCIA

Editar área de influencia

Nombre
EN EL AREA DE INFLUENCIA

Destinos ▼ Porcentaje

ASOCIACIÓN ESTUDIANTIL ASOCIACIÓN ESTUDIANTIL UNIDAD ACADÉMICA
BIBLIOTECA CENTRAL LABORATORIO MICRO DIGA

*Para borrar una distribución se debe hacer click en ella dentro del mapa.
*La distribución "No asignada" no se puede borrar.

Cancelar Guardar

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Con un cambio evidente, se sigue atendiendo el requerimiento del usuario con una interfaz mucho más amigable e intuitiva. Muestra la totalidad de porcentajes y da la opción de modificarlo fácilmente al hacer clic sobre la parte del gráfico que se quiere borrar. Se puede agregar un nuevo destino en la parte superior, seleccionarlo e ingresar el porcentaje que le corresponde; además, ajusta el porcentaje por si se pasa del 100 % en la asignación. De esta forma, el usuario tiene una mejor y más fluida experiencia con la aplicación.

La aplicación fue diseñada de tal forma que el usuario tenga una menor curva de aprendizaje y una mejor experiencia, con una interfaz mucho más amigable, directa y eficiente. Gracias a la característica asíncrona que ofrece Node.js, la página puede estar disponible mientras la información es solicitada; así disminuyen los tiempos de espera de carga de información. Ya que es una SPA, la página no es recargada en ningún momento, solo se cambia el componente mostrado bajo demanda del usuario. Esto gracias al router de Angular, que se encarga de hacer el cambio de componente dependiendo de la ruta a la que se quiere dirigir el usuario. Se explica a profundidad en el apartado de Lazy Loading, desarrollado adelante en el documento.

2.12.1.1. Filtro de búsqueda y botón para crear

Los cambios en el diseño de la aplicación redujeron la cantidad de filtros de búsqueda y botones de crear, debido a que en la página principal se muestran ambos. Los componentes se suscriben a un servicio que les permite utilizar el mismo botón para crear y también el mismo filtro de búsqueda. De esta forma se utiliza el mismo componente para realizar tareas desde cualquier componente que necesite hacer uso de ellas.

En los prototipos se diseñó que cada tabla tuviera un filtro de búsqueda en la parte superior y un botón para crear en la parte inferior.

Figura 27. Diseño de tabla del prototipo

ID	Nombre	Descripción	Tipo	Estado	Acción
1	AEI	Asociación de Estudiantes de Ingeniería	Asociación	✓	⋮
2	AED	Asociación de Estudiantes de Derecho	Asociación	✓	⋮
3	AEU	Asociación de Estudiantes de Ingeniería	Asociación	✓	⋮
4	DIGA	Dirección General de Administración	Asociación	✓	⋮
5	División de Servicios Generales	Contratación de servicio de mantenimiento y limpieza	División	✓	⋮

Fuente: prototipo desarrollado para el módulo de Actividad Comercial.

A diferencia del diseño final de la aplicación que aprovecha todo el espacio de la ventana para mostrar la información, utiliza el menú y la barra superior para almacenar el botón para crear y el filtro de búsqueda respectivamente. Esto, además de brindar un espacio mayor para mostrar datos, permite una mejor distribución de los componentes. Para hacer uso del filtro general y el botón se debe crear servicios que indiquen lo que se quiere filtrar y cómo se debe filtrar.

El servicio de búsqueda tiene el nombre 'BusquedaService', dentro del cual se cuenta con lo siguiente:

Figura 28. **Servicio para búsqueda**

```
import { Subject } from 'rxjs';

export class BusquedaService {

    private busqueda = new Subject<any>();

    notifyObservable$ = this.busqueda.asObservable();

    constructor() { }

    public notificarBusqueda(data: any){
        if(data){
            this.busqueda.next(data);
        }
    }
}
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Se crea un observable para notificar la búsqueda desde otro componente; en este caso, el componente de 'home', el cual debe crear un objeto de 'BusquedaService', usar el método notificarBusqueda y enviar como parámetro el valor del filtro. Luego, en el componente que está cargando la información, de igual forma se crea un objeto de 'BusquedaService' para usar notifyObservable\$; se suscribe y ejecuta lo que deba ejecutar con la información recolectada en el servicio.

Figura 29. **Uso del servicio para búsqueda desde componente**

```
suscribirFiltrar() {  
  this.filtrarSub = this._busqueda.notifyObservable$.subscribe(res => {  
    this.filtrarTabla(res);  
  });  
}
```

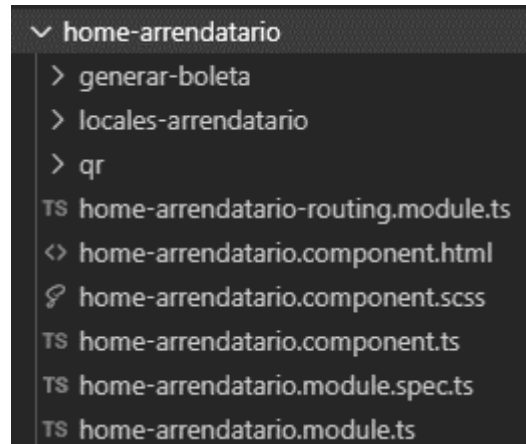
Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Es de la misma forma con el botón crear, pero las acciones, en lugar de filtrar, serán abrir diálogos que permitan crear nuevos registros.

2.12.2. Distribución de los componentes

Los componentes se agruparon por roles dentro de la aplicación; de esta forma se tiene un 'home' para el administrador, operador de actividad comercial, usuario de presupuesto y arrendatario. Cada uno de estos componentes tiene dentro de su carpeta los componentes de los que hacen uso; por ejemplo, para el home del arrendatario, tiene dentro las funciones que este realiza:

Figura 30. **Estructura de componente home-arrendatario**



Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Entonces, ya que un arrendatario tiene las opciones de generar boleta, ver sus locales y generar un código QR, esos componentes se crean dentro de su home. Hay componentes que son utilizados por varios roles; en ese caso, el componente existe solo una vez en el componente que se usó primero. Esta es una de las ventajas de utilizar componentes, ya que se pueden utilizar los mismos sin necesidad de crear otro para un rol distinto.

Como se observa en la figura 30, home-arrendatario cuenta con varios archivos. Normalmente, un componente tiene entre 3 y 4 archivos:

- Archivo HTML.
- Archivo de estilo (CSS, SCSS, SASS, LESS, STYL), es a elección del usuario y su preferencia.

- TS archivo TypeScript para la lógica del componente.
- Un archivo de especificación del componente, cuya generación se puede ignorar configurando el proyecto.

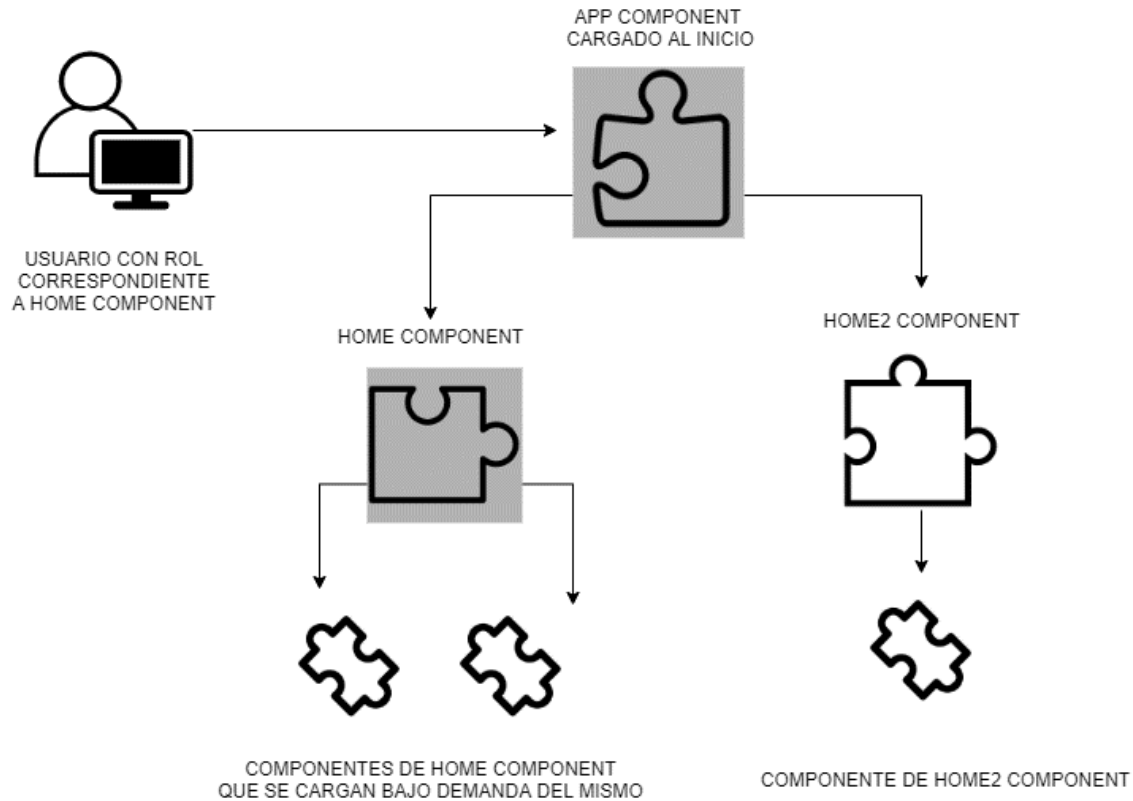
Los otros archivos son generados por la implementación de Lazy Loading, que se detalla a continuación.

2.12.3. Lazy Loading

Por defecto, los módulos del proyecto son cargados todos al inicio, aunque no vayan a ser utilizados por el usuario. Para aplicaciones de tamaño considerable, la implementación de Lazy Loading puede ayudar a optimizar los tiempos de carga.

Lazy Loading segmenta la aplicación dependiendo de la ruta que el usuario está accediendo; de esta forma se descarga solo el componente que el usuario solicita para utilizar.

Figura 31. **Lazy Loading**



Fuente: elaboración propia, empleando draw.io.

La implementación de Lazy Loading en Angular 6 se hace dentro de las rutas del router. Utiliza importación dinámica; es decir, que cada vez que se activa una ruta el router va a solicitar el módulo correspondiente. De esta forma, cada componente cuenta con un módulo dentro del cual se especifica todo lo que va a importar, declarar y utilizar como proveedor, y con un módulo de router independiente. De esta forma, el módulo y router principal mantienen una estructura más limpia y liviana.

2.12.3.1. Implementación de Lazy Loading

Para implementar Lazy Loading en el router principal se debe registrar rutas de la siguiente forma:

Figura 32. Lazy Loading router principal

```
import { ModuleWithProviders } from "@angular/core";
import { Routes, RouterModule } from "@angular/router";

const router: Routes = [
  { path: "", redirectTo: "login", pathMatch: "full" },
  {
    path: "login",
    loadChildren: "./login/login.module#LoginModule"
  },
  {
    path: "arrendatario",
    loadChildren: "./home-arrendatario/home-
arrendatario.module#HomeArrendatarioModule"
  }
]

export const routes: ModuleWithProviders = RouterModule.forRoot(router);
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

De esta forma, cuando se active la ruta 'arrendatario', el router irá a cargar el módulo 'HomeArrendatarioModule', dentro del cual se debe especificar todos los componentes que usará únicamente dentro del mismo.

Figura 33. Lazy Loading HomeArrendatarioModule

```
import { LocalSaveService } from "../services/local-save-service.service";
import { ComponentPageTitle } from "../services/page-title.component";
import { Handle } from "../services/handle";
import { ControlComponent } from "../services/control";
import { BusquedaService } from "../services/busqueda.service";
import { QrComponent } from '../qr/qr.component';

@NgModule({
  imports: [CommonModule, HomeArrendatarioRoutingModule, ImportsModule],
  declarations: [HomeArrendatarioComponent, QrComponent ],
  providers: [
    { provide: MatPaginatorIntl, useClass: CustomMatPaginatorIntl },
    LocalSaveService,
    ComponentPageTitle,
    Handle,
    ControlComponent,
    BusquedaService
  ],
  entryComponents: [QrComponent]
})
export class HomeArrendatarioModule {}
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Para los componentes que funcionan dentro de diálogos, su definición debe ser dentro del módulo que abrirá en la parte de 'entryComponents', como se puede observar en la figura 33 con el componente 'QrComponent'.

Si el componente cargado tiene hijos, entonces dentro del módulo de ruteo interno se deben definir, de la siguiente forma:

Figura 34. Lazy Loading cargar hijos

```
import { NgModule } from "@angular/core";
import { Routes, RouterModule } from "@angular/router";
import { HomeArrendatarioComponent } from "../home-arrendatario.component";

const routes: Routes = [
  {
    path: "",
    component: HomeArrendatarioComponent,
    canActivate: [RoleGuard, AuthGuard],
    data: { ROL: 6 },
    children: [
      {
        path: "",
        redirectTo: "fichaArrendatario",
        pathMatch: "full"
      },
      {
        path: "fichaArrendatario",
        loadChildren:
          "../home/arrendatarios/ficha-arrendatario/ficha-
arrendatario.module#FichaArrendatarioModule"
      },
      {
        path: "locales",
        loadChildren:
          "../locales-arrendatario/locales-
arrendatario.module#LocalesArrendatarioModule"
      },
    ]
  }
];

@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class HomeArrendatarioRoutingModule {}
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Como se puede observar, dentro de este router interno no se declara una ruta para 'QrComponent', ya que este está declarado en el módulo como un componente de entrada. Además, se puede observar que dentro del router

interno se llaman componentes ubicados en otros home, lo cual nos ayuda a reutilizar el código de la aplicación gracias a los componentes.

Como consideración, el import dentro de los router internos debe ser 'forChild', ya que no se permite tener dos 'forRoot' dentro de la aplicación. Una vez definidos los módulos para los hijos ya está implementado Lazy Loading en el proyecto, disminuyen los tiempos de carga y también el tamaño de los archivos cuando se genera un archivo de distribución. De esta forma, en lugar de tener un archivo grande, hay muchos archivos pequeños.

Dentro de la figura anterior hay dos partes importantes, RoleGuard y AuthGuard, los cuales se encargan de proteger las rutas.

2.12.4. AuthGuard

Es un guardián para proteger el acceso a las rutas. Se encarga de ver que la persona que intenta cargar un componente esté autenticada, verifica si tiene un token válido y, de no tenerlo, redirecciona al login de la aplicación.

Figura 35. AuthGuard

```
import { Injectable } from "@angular/core";
import {
  CanActivate,
  ActivatedRouteSnapshot,
  RouterStateSnapshot,
  Router
} from "@angular/router";
import { Observable } from "rxjs";
import { LoginService } from "../services/login.service";

@Injectable({
  providedIn: "root"
})
export class AuthGuard implements CanActivate {
  constructor(private _router: Router, private _loginService: LoginService) {}

  canActivate(
    next: ActivatedRouteSnapshot,
    state: RouterStateSnapshot
  ): Observable<boolean> | Promise<boolean> | boolean {

    if (this._loginService.isAuthenticated()) {
      return true;
    }

    this._router.navigate(["/login"]);
    return false;
  }
}
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

2.12.5. RoleGuard

Cumple una función similar a AuthGuard, pero ya que el usuario tiene un token válido verifica si tiene los permisos para acceder a la ruta que está intentando; es decir, RoleGuard limita a los usuarios a navegar solo en las páginas que deben hacerlo y no utilizar componentes hechos para otros roles.

Figura 36. RoleGuard

```
import { Injectable } from "@angular/core";
import { CanActivate, ActivatedRouteSnapshot, Router, Route
} from "@angular/router";
import { Observable } from "rxjs";
import { LocalSaveService } from "../services/local-save-service.service";
import { Handle } from "../services/handle";

@Injectable({
  providedIn: "root"
})
export class RoleGuard implements CanActivate {
  constructor(
    private _router: Router,
    private Handle: Handle,
    private localSave: LocalSaveService
  ) {}

  canActivate(
    next: ActivatedRouteSnapshot
  ): Observable<boolean> | Promise<boolean> | boolean {
    const user = this.localSave.get("user");

    if (user == undefined) {
      this.Handle.message("Debe iniciar sesión");
      this._router.navigate(["/login"]);
      return false;
    }
    if (user.ROL === next.data.ROL) {
      return true;
    }

    this._router.navigate(["/login"]);
    return false;
  }
}
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

De esta forma valida si el rol del usuario es igual al rol que tiene asignado la ruta y, de no ser así, lo envía al login lo redirige al home que le corresponde.

2.12.6. Gateway

En el proyecto hay un servicio llamado 'Gateway', dentro del cual se almacenan valores para consumir los servicios. En este se centralizan todas las peticiones http que hace el cliente; si se desea hacer una modificación no se busca dentro de los componentes, sino en el servicio.

Dentro del servicio se definen las ip a las que se irá a consumir los servicios; en este caso, será la ip donde se encuentra el servidor. Cuenta con un único método llamado 'consumir', que se encarga de mandar las peticiones y recibe una llave. Dentro del servicio se ubica un objeto que almacena llave, valor de los endpoints, de la siguiente forma:

Figura 37. Estructura de llaves para Gateway

```
const endpoints = new Map();

// PARA LOGEO
endpoints.set("LOGIN", "/api/auth");
endpoints.set("ARR_LOGIN", "/api/autharrendatario");

// PARA REQUISITO
endpoints.set("LISTA_REQUISITO", "/api/actividadcomercial/requisito/lista");
endpoints.set("CREAR_REQUISITO", "/api/actividadcomercial/requisito/crear");
endpoints.set("EDITAR_REQUISITO", "/api/actividadcomercial/requisito/editar");
endpoints.set("BORRAR_REQUISITO", "/api/actividadcomercial/requisito/borrar");
```

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

El servicio se encarga de enviar también el token de autorización para los servicios que lo necesitan y valida las llaves antes de enviar la petición.

2.13. Ambientes de la aplicación

La aplicación fue desarrollada en diferentes ambientes dependiendo del estado en el que se encontraba, cada ambiente con características distintas.

2.13.1. Desarrollo

Fue hecho de forma local con los siguientes recursos:

- Lenovo ThinkPad. 16 GB de RAM, Intel core i7, con Debian 9
- Oracle 11g como base de datos
- Node.js 12.16.1
- Angular 6

2.13.2. Pruebas y producción

Para las pruebas y producción se utilizaron los siguientes recursos:

- Servidor Dell Power Edge R710, Intel Xeon 2.93 Ghz, 4 núcleos, 8 hilos, 128 GB de RAM y 3 TB de disco con sistema operativo Oracle Enterprise Linux.
- Oracle 11g como base de datos en el esquema del SIIF
- Node.js 12.16.1

- Angular 6

2.14. Atención a los requerimientos

Los requerimientos iniciales fueron el punto de partida para el desarrollo de la aplicación, pero algunos variaron luego de conversaciones con el Departamento de Procesamiento de Datos y los usuarios finales. El backlog del proyecto estuvo en constante cambio y gracias a la metodología de desarrollo fue posible atender los cambios en los requerimientos.

Los requerimientos tuvieron un proceso de cambio y desarrollo, que finalizaron con la aprobación por parte de los usuarios finales. Luego se realizaron pruebas exploratorias para reducir el riesgo de fallo en la aplicación, en conjunto con el Departamento de Procesamiento de Datos, y buscar errores en la funcionalidad. Se dedicaron aproximadamente 30 horas en el período de pruebas interno y la corrección de los errores. Se aplicó el siguiente formato para los casos de prueba:

- ID: identificador para el caso de prueba con el formato CP (Caso de Prueba) y un número.
- Caso de prueba: nombre del caso de prueba (ej. Crear Campus).
- Descripción: una breve descripción de la funcionalidad.
- Fecha: la fecha en la que se registra el caso de prueba.

- Funcionalidad: descripción de la secuencia que se realiza para cumplir el caso de prueba.
- Acciones de entrada: información de entrada que recibe el caso de prueba para su realización.
- Resultado esperado: los valores de salida o comportamiento que se espera luego de llevar a cabo el caso de prueba.
- Procedimientos especiales requeridos: si para ejecutar el caso de prueba se debe hacer algo fuera de la aplicación, por ejemplo, contactar con el Departamento de Procesamiento de Datos para solicitar alguna información.
- Dependencias con otros casos de prueba: si se debe ejecutar un caso de prueba antes se especifica el identificador.
- Resultado obtenido: resultado o comportamiento recibido luego de ejecutar el caso de prueba.
- Estado: si se pasó el caso de prueba, puede ser satisfactorio, no satisfactorio o parcialmente satisfactorio, si se debe arreglar algo que no afecta el resultado obtenido.
- Última fecha de estado: la fecha en la que se obtuvo el resultado expresado anteriormente.
- Observaciones: por si se quiere hacer una aclaración o anotar algún mensaje de consola para ayudar a la corrección de la funcionalidad.

Luego de ejecutar las pruebas en un día se genera una lista de casos de prueba no satisfactorios para su corrección.

Tabla IV. **Resumen de casos de prueba no satisfactorios**

Lista de casos de prueba no satisfactorios	Comentarios
CP-17	
CP-19	
CP-23	El index estaba unusable
CP-30	
CP-35	Se sigue registrando una nueva asociación para historial e igual se puede borrar si se necesita

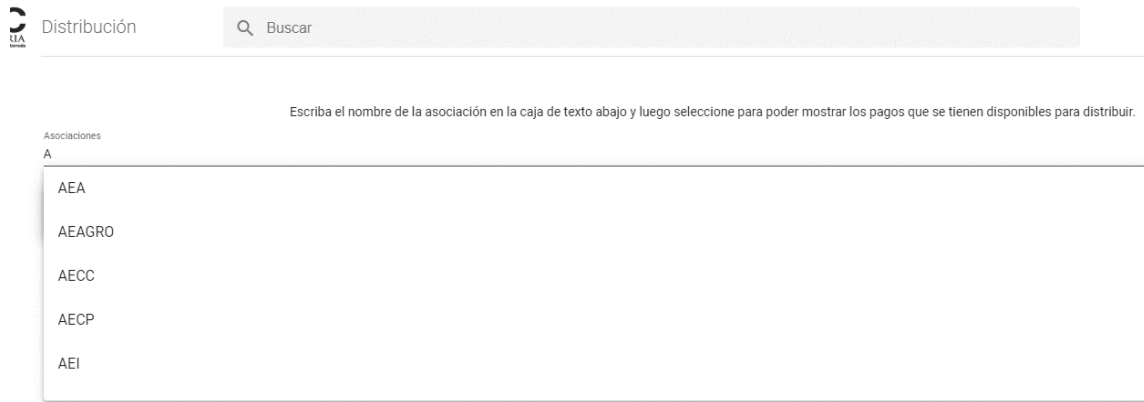
Fuente: elaboración propia, con datos del reporte de pruebas con fecha 18 de marzo de 2020.

2.14.1. Evaluación de objetivos

Para evaluar los objetivos se atiende de uno en uno con evidencia de su implementación.

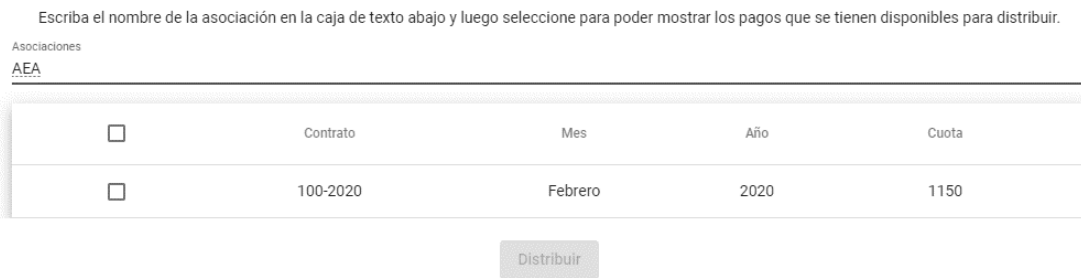
Para consultar las asociaciones que pertenecen a alguna área de influencia y visualizar el presupuesto disponible, se llegó a un acuerdo que permite tener una mejor experiencia. Ya que las asociaciones se acercan al Departamento de Presupuesto de la Universidad, se cambió la visualización de asociaciones por área de influencia por un campo de autocompletado. Se ingresa el nombre de la asociación y luego de seleccionar se muestran los pagos que tiene disponible para reclamar su porcentaje.

Figura 38. **Pantalla para visualización de asociaciones**



Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Figura 39. **Pagos disponibles para una asociación**



Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Al seleccionar como mínimo un pago disponible se activa el botón 'Distribuir', lo cual genera un archivo en formato PDF con el total que la asociación va a recibir.

Sobre registrar, editar, borrar y consultar las entidades arrendatario, local y área de influencia, se muestra el mecanismo utilizado para la entidad área de influencia, el cual se utiliza para las demás.

Figura 40. **Visualización de área de influencia**



USAC TRICENTENARIA

Áreas de influencia

Q Buscar

1 - 2 d

Nombre	Acción
EN EL AREA DE INFLUENCIA	⋮
FUERA DEL AREA DE INFLUENCIA	⋮

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Para crear un registro en las entidades mencionadas se utiliza el botón crear, el cual muestra un diálogo que solicita la información necesaria.

Figura 41. **Crear una nueva área de influencia**

EN EL AREA DE INFLUENCIA

Crear área de influencia

Nombre

Destinos ▼ Porcentaje

No asignado

*Para borrar una distribución se debe hacer click en ella dentro del mapa.
*La distribución "No asignada" no se puede borrar.

Cancelar Guardar

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Todos los mantenimientos de entidades cuentan con un apartado de acciones en la tabla, que muestra una lista con, al menos, las opciones de editar y borrar. Al hacer clic en Editar se muestra un diálogo como en la figura 41, pero con información del registro que se está editando.

Figura 42. **Editar un área de influencia**

Editar área de influencia

Nombre
EN EL AREA DE INFLUENCIA

Destinos ▼ Porcentaje

ASOCIACIÓN ESTUDIANTIL ASOCIACIÓN ESTUDIANTIL UNIDAD ACADEMICA
BIBLIOTECA CENTRAL LABORATORIO MICRO DIGA

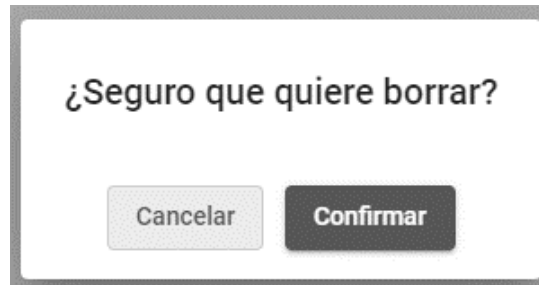
*Para borrar una distribución se debe hacer click en ella dentro del mapa.
*La distribución "No asignada" no se puede borrar.

Cancelar Guardar

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Finalmente, para eliminar un registro se hace clic en la opción Borrar, lo cual muestra un diálogo de confirmación para evitar equivocaciones por parte de los usuarios. Al aceptar se elimina el registro de la entidad.

Figura 43. **Borrar un área de influencia**



Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Los pagos no cuentan con una tabla en la cual se muestren todos. Se pueden observar en la ficha de cada arrendatario, desde un perfil de Actividad Comercial o el usuario del arrendatario.

Figura 44. **Visualización de pagos por arrendatario**

de León Mejía de Puac, Xeny
Maribel

CORREO: xmejia@gmail.com

DIRECCIÓN: San Carlos Sijá

FECHA DE NACIMIENTO: 9/10/1985

FECHA DE REGISTRO: 20/3/2020

NÚMERO DE CASA: 55667710

NÚMERO DE CELULAR: 55667710

PROFESIÓN: Ama de casa

↻ ▾ Cuenta corriente

Mes	Año	Cuota	Estado
Enero	2020	1150	PAGADO
Febrero	2020	1150	PAGADO
Marzo	2020	1150	PENDIENTE

El arrendatario tiene una deuda de Q1150

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

La generación de los pagos se hace automáticamente al finalizar un proceso de arrendamiento o al realizar un convenio de pago. El usuario no tiene

posibilidad de crear pagos o eliminarlos por su cuenta. Para quitar el pago a un arrendatario se hace uso de la función de exoneración que se encuentra en el menú.

Figura 45. Exoneración de pagos

Año	Mes	Descripción	Negocio
2020	Enero	EXONERACION DE PRUEBA.	Tienda

Crear exoneración

Mes: Enero

Año: 2020

Arrendantes: TODOS

Negocios: Tienda

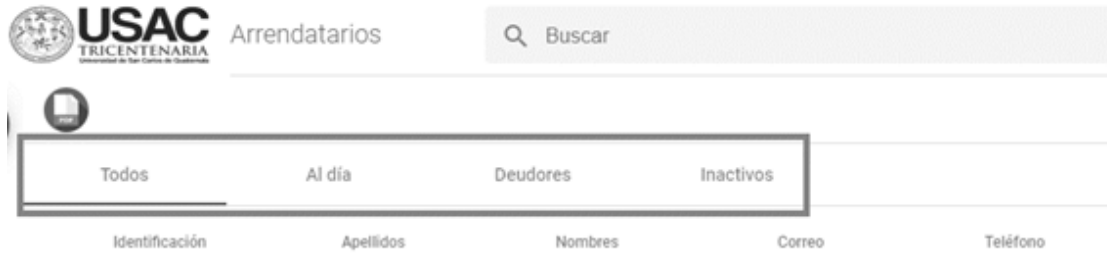
Descripción: Exoneración

Cancelar Guardar

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

La aplicación es capaz de manejar los pagos internamente y actualizar los estados de arrendatarios en tiempo real. Para consultar los arrendatarios se cuenta con varios filtros en el apartado Arrendatarios.

Figura 46. **Filtros para consultar arrendatarios**



Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Como se puede observar se puede filtrar a los arrendatarios mostrando todos, los que están al día, los que deben y finalmente, los que están inactivos, es decir los que no tienen local arrendado en el momento de la consulta o que fueron dadas de baja por los usuarios. Los usuarios tanto de Actividad Comercial como los arrendatarios tienen acceso a ver la cuenta corriente del mismo, resumida o un archivo formato PDF con el detalle de esta.

Figura 47. **Resumen de cuenta corriente**

The screenshot shows the 'Cuenta corriente' section of the application. It features a title 'Cuenta corriente' with a refresh icon and a dropdown arrow icon to its left. Below the title is a table with the following data:

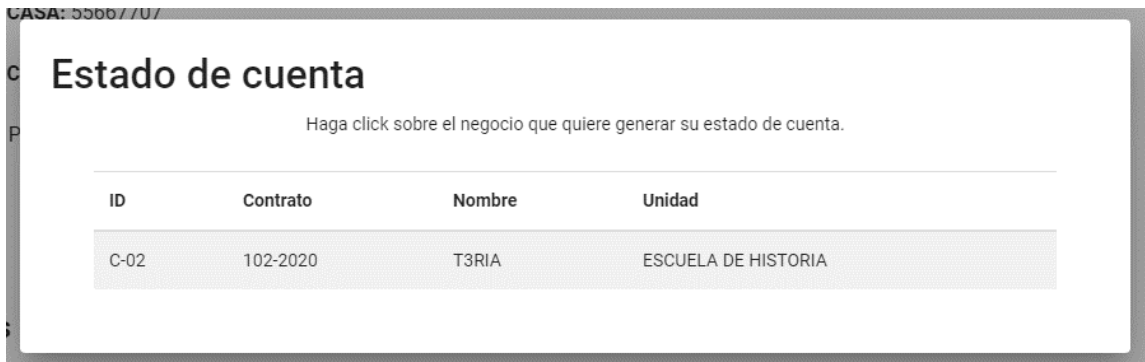
Mes	Año	Cuota	Estado
Marzo	2020	10000	PENDIENTE

El arrendatario tiene una deuda de Q10000

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

El detalle de cuenta corriente muestra todos los pagos que tiene pendiente el arrendatario en el contrato que se seleccione.

Figura 48. **Selección de contrato para detalle de cuenta corriente**



CASA: 55667/07

Estado de cuenta

Haga click sobre el negocio que quiere generar su estado de cuenta.

ID	Contrato	Nombre	Unidad
C-02	102-2020	T3RIA	ESCUELA DE HISTORIA

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Luego de seleccionar, se genera automáticamente el archivo que muestra todos los cargos que está pagando el arrendatario por mes.

Figura 49. **Fragmento de detalle de cuenta corriente**

Universidad de San Carlos de Guatemala
Dirección General Financiera
Sistema para distribución de ingresos
Reporte emitido el: 15-3-2020

Estado de cuenta

Arrendamiento de negocios

Contrato: 102-2020

Arrendante:

Ubicación: En la escuela de historia.

Año	Mes	Descripción	Cargos Q.	Abonos Q.	Saldo Q.
2020	Marzo	BASURA	50	0	50
2020	Marzo	RENTA	600	0	600
2020	Abril	BASURA	50	0	50
2020	Abril	RENTA	600	0	600
2020	Mayo	BASURA	50	0	50
2020	Mayo	RENTA	600	0	600
2020	Junio	BASURA	50	0	50
2020	Junio	RENTA	600	0	600
2020	Julio	BASURA	50	0	50
2020	Julio	RENTA	600	0	600

Fuente: archivo generado por aplicación desarrollada para el módulo de Actividad Comercial.

Además de ofrecer control para los arrendatarios, se presentan opciones similares para los locales. De esta forma, el usuario puede consultar el estado en el que se encuentra su proceso de arrendamiento en tiempo real.

Figura 50. **Filtros para consultar locales**

The screenshot shows the 'Locales' section of the USAC application. At the top left is the USAC logo (Universidad del Sagrado Corazón). To its right is the word 'Locales' and a search bar with a magnifying glass icon and the text 'Buscar'. Below the search bar is a navigation bar with five tabs: 'Todos', 'Ocupados', 'Disponibles', 'En proceso', and 'En jurídico'. The 'Todos' tab is currently selected. Below the navigation bar is a table with the following columns: 'ID Local', 'Campus', 'Nombre', 'Unidad', and 'Área de influencia'. The table contains one row of data.

ID Local	Campus	Nombre	Unidad	Área de influencia
CAFE-01	Campus Central	Cafetería 01	FACULTAD DE ARQUITECTURA	EN EL AREA DE INFLUENCIA

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Con ayuda de los filtros se puede consultar el estado de los locales, ya sea todos, los ocupados, los disponibles, aquellos que están actualmente en proceso y los que están en jurídico esperando un contrato para finalizar su proceso. El estado se actualiza según las acciones que realice el usuario. Se inicia un proceso de arrendamiento para un local disponible, ya sea primer contrato o renovación. Al terminar de ingresar la información solicitada se mueve el local a la pestaña 'En proceso' y con esto se puede solicitar los requisitos. Al terminar el proceso de solicitud de requisitos se genera un archivo con el que el local pasa a jurídico. Cuando el arrendatario tiene el contrato se puede dar por finalizado el proceso, lo cual genera que el estado del local cambie a ocupado.

Una vez los objetivos se muestran cumplidos con su correcta implementación y posteriores pruebas, se puede ampliar el SIIF con un módulo específico para la actividad comercial, como la aplicación desarrollada.

2.14.2. Ventajas del proyecto

Una vez finalizado el proyecto se lograron concretar las siguientes ventajas:

- División de tareas gracias a los roles asignados a cada usuario.
- Información estandarizada; es decir, se reducen los múltiples nombres para un tipo de local, por ejemplo.
- Información disponible desde cualquier lugar y sin necesidad de tener una computadora con muchos recursos.
- Información actualizada sin necesidad de realizar alguna tarea adicional para ello.
- Procesos automatizados como la generación de notas automáticas.
- Visualización de los procesos de arrendamiento más dinámicos.
- Copia digital de la información. Ya no se tendrá toda la información solo en un disco duro, porque el servidor almacena los archivos y los sirve cuando un usuario quiere consultarlos.
- Alternativa al proceso para pagar, como la generación de boleta. Un arrendatario se puede ahorrar mucho tiempo con la función de generar boleta.
- Actualización de estados en tiempo real: los arrendatarios y locales cambian de estado sin necesidad de una acción adicional.

- Visualización de la cuenta corriente de los arrendatarios en tiempo real.
- Reportes de los pagos realizados por los arrendatarios, lo que ahorra mucho tiempo al usuario.
- Tener una aplicación apropiada para los estándares web actuales que sirva como proyecto semilla para una expansión en el futuro.
- Mejor manejo de los arrendatarios, su información personal y la de sus pagos.
- Con el código QR generado, los monitores tendrán información de los pagos del arrendatario al escanearlo desde su teléfono móvil.
- Automatización en la distribución de pagos para una asociación vigente.
- Visualización de los pagos que corresponden a una asociación vigente.
- Distribución de pagos anual automatizada según configuración de la aplicación.
- Contar con un módulo de configuración que permite a los usuarios cambiar los parámetros en las notas automáticas y reportes.
- Mejor control con los locales que se encuentran dentro de la Universidad.
- Organización del campus central y centros universitarios.

- Obtener información filtrada y generar reportes de esta en entidades arrendatario y local.
- Contar con una aplicación altamente configurable para afrontar distintos escenarios sin necesidad de un cambio en el código fuente.

3. FASE SE ENSEÑANZA – APRENDIZAJE

Se elaboró un plan para la enseñanza- aprendizaje de los usuarios finales. Consta de las siguientes partes:

3.1. Estrategia de enseñanza

La estrategia de enseñanza planteada para los usuarios finales consiste básicamente en un tiempo de adaptación de 2 semanas para se familiaricen con la aplicación. Podrán usarla ellos mismos con asistencia. Harán ciclos completos de los distintos procesos. Habrá 3 semanas de soporte para las dudas que puedan surgir luego del acompañamiento. Además, como primera línea de atención, se entregará manuales de usuario detallados que ayuden a responder a situaciones de flujo normal sin necesidad de hacer contacto con alguna persona.

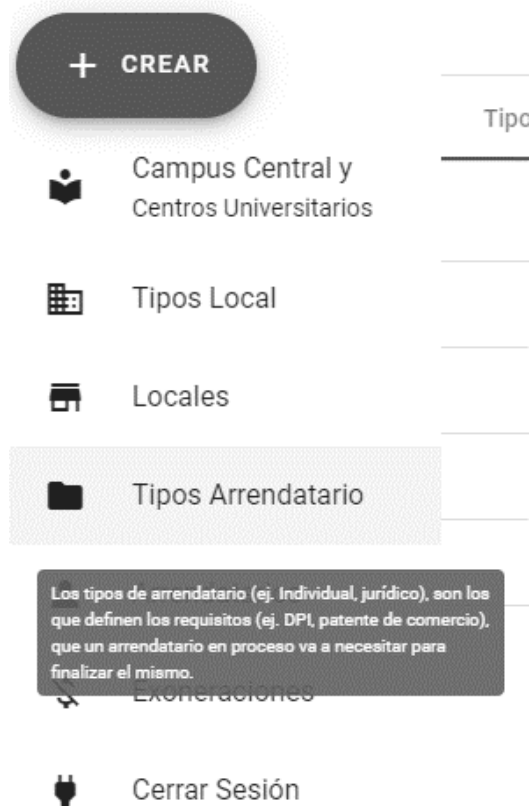
3.2. Aplicación altamente intuitiva

Se diseñó una aplicación altamente intuitiva para ayudar a los usuarios en su curva de aprendizaje, para que puedan tener una mejor experiencia. A pesar de que la aplicación sigue un diseño minimalista, también incorpora algunos aspectos que la hacen altamente intuitiva.

3.2.1. Iconos asociados a acciones

En toda la aplicación se pueden encontrar iconos acompañados de texto descriptivo o de una anotación que complementa su significado y funcionalidad, para ayudar al usuario a entender la función de cada uno.

Figura 51. **Icono acompañado de anotación en menú**



Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

De esta forma, el usuario recibe asistencia de la aplicación cuando navega entre funciones de esta.

3.2.2. Elementos de ayuda en formularios

En todos los formularios se presentan al usuario distintos elementos para un uso sencillo. Los elementos de ayuda son los siguientes:

3.2.2.1. Anotaciones de error en formularios

Para el usuario, muchas veces es difícil entender las limitantes que puede tener un campo de ingreso de información; por ejemplo, el tamaño máximo y el mínimo que puede tener, si debe ingresarlo o no. Por tales razones, la aplicación incluye en el diseño una serie de anotaciones de error debajo de los campos solicitados que se muestran de la siguiente forma:

Figura 52. **Anotación de error en campo de formulario**

The image shows a web form titled "Crear arrendatario" with several input fields. Two fields are highlighted with red boxes to indicate errors: "Tipo de arrendatario" and "Estado civil". Both fields have a small downward arrow icon and the text "Campo requerido" below them. The other fields include "Nombres", "Apellidos", "Fecha nacimiento" (with a calendar icon), "Profesión/Oficio", "DPI", "Extendida en", "Dirección", "Teléfono celular", "Teléfono de casa", and "Correo". At the bottom, there are "Cancelar" and "Crear" buttons.

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

3.2.2.2. Listas de selección en formularios

Para la información que tiene relación con algo se muestra al usuario listas de selección y no un campo de ingreso, para brindarle una serie de opciones válidas y ayudar a la base de datos a estandarizar la información.

Figura 53. Lista de selección en formulario

The image shows a web form titled "Crear arrendatario". A dropdown menu is open over the "Nombres" field, displaying the following options: "Extranjero", "Persona individual", "Representante banco", and "Representante legal". The "Representante banco" option is highlighted. Below the dropdown, the form contains several input fields: "Apellidos", "Fecha nacimiento" (with a calendar icon), "Soltero/a" (with a dropdown arrow), "Profesión/Oficio", "DPI", "Extendida en", "Dirección", "Teléfono celular", "Teléfono de casa", and "Correo". At the bottom of the form are two buttons: "Cancelar" and "Crear".

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

3.2.3. Activación de botones para realizar acción

Una de las formas de ayudar al usuario a usar la aplicación es la activación de botones con el flujo de información; es decir, si la información no está completa, el botón para realizar la acción no estará disponible hasta que el usuario ingrese lo solicitado.

Figura 54. **Activación de botón para guardar**

The figure consists of two vertically stacked screenshots of a web form titled "Crear campus".

The top screenshot shows the form with an empty text input field labeled "Nombre". Below the input field are two buttons: "Cancelar" and "Guardar". The "Guardar" button is dimmed and has a light gray background, indicating it is inactive.

The bottom screenshot shows the same form, but the text input field now contains the text "Campus Mixcd". The "Guardar" button is now dark gray with white text and a thick black border, indicating it is active.

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

3.2.4. Anotaciones en procesos

La aplicación cuenta con anotaciones en los procesos que pueden requerir asistencia. Da instrucciones al usuario mientras él está realizando alguna acción.

Figura 55. **Anotación en proceso de crear área de influencia**

INFLUENCIA

DE

Crear área de influencia

Nombre

Destinos ▼ Porcentaje

No asignado

No asignado: 100

*Para borrar una distribución se debe hacer click en ella dentro del mapa.
*La distribución "No asignada" no se puede borrar.

Cancelar Guardar

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

Como se observa en la figura 39, al momento del usuario crear una nueva área de influencia se le hacen dos anotaciones, las cuales lo ayudan a entender el funcionamiento de esta. Por otro lado, en diálogo para iniciar un proceso de arrendamiento se muestra una anotación mucho más larga que indica la finalidad del proceso de arrendamiento y la información del local que está por iniciar el proceso. Además, informa al usuario cómo utilizar el autocompletado del campo para seleccionar arrendatario.

Figura 56. **Anotación en iniciar proceso de arrendamiento**

Iniciar proceso de arrendamiento | I-02

Aquí se inicia el proceso de arrendamiento entre un local y un arrendatario, luego de ser asignado se deben completar ciertos pasos y cumplir con algunos requisitos para poder consolidar el arrendamiento. Al iniciar el arrendamiento se mostrarán los primeros requisitos que el arrendatario debe cumplir.

Iniciando proceso para local Independiente 02 dentro de ESCUELA DE HISTORIA de tipo Puesto independiente. Este local está ubicado en: En el límite de influencia., y pertenece al área de influencia: FUERA DEL AREA DE INFLUENCIA

Para seleccionar al arrendatario escriba el nombre y luego haga click en la lista que se mostrará luego abajo.

Tipo de proceso

Arrendatarios

Nombre

Negocio

Fuente: aplicación desarrollada para el módulo de Actividad Comercial.

3.3. Fase de pruebas con usuarios

Una vez fue aprobada, la aplicación enfrentó una serie de pruebas de uso con los usuarios finales. En dos semanas se dio a conocer al usuario las funcionalidades y se le dio apoyo con las dudas. Se simuló un proceso completo con las funciones de la aplicación, para que los usuarios se pudieran sentir familiarizados con la misma. Durante la fase de pruebas se hicieron correcciones de forma y diseño con los usuarios, así como el formato de los reportes.

Durante las fases de pruebas se hicieron algunas mediciones sobre los tiempos que tomaba a los usuarios finales realizar tareas comunes antes de la aplicación, con los siguientes resultados:

Tabla V. **Tiempos para generar acciones**

Actividad	Tiempo previo	Tiempo actual
Buscar arrendatario con deuda	3 - 5 minutos	< 1 segundo
Generar reporte de pagos y distribuirlos por cargo	1 semana	~ 3 segundos
Generar las notas para el proceso de arrendamiento	5 - 10 minutos por nota	~ 3 segundos
Registrar un pago del usuario	3 - 5 minutos	< 1 segundo

Fuente: elaboración propia con base en mediciones con usuarios finales.

CONCLUSIONES

1. Gracias al desarrollo de una aplicación web que atiende los requerimientos de los usuarios de Actividad Comercial y cumple con los estándares del Departamento de Procesamiento de Datos se hizo una expansión en el SIIF al agregar un módulo para la actividad comercial dentro del mismo. Esta ayuda a reducir tiempos y a automatizar procesos para los usuarios que harán uso de esta.
2. Se mejoró la organización en la distribución de pagos para las asociaciones estudiantiles, a partir de cambios que se aceptaron durante el desarrollo del requerimiento "Mostrar las asociaciones". Además, se mejoró el funcionamiento y la experiencia de usuario al permitir a los representantes de la asociación consultar los pagos que tienen disponibles para cobrar.
3. Una vez desarrollados los requerimientos solicitados hay un mantenimiento completo de las entidades arrendatario, local y área de influencia. El usuario puede ver, crear, editar y borrar los registros que no alteran el comportamiento de la aplicación. Se desarrolló una estrategia para generar los pagos automáticamente, lo cual ahorra mucho tiempo a los usuarios.
4. Se provee a los usuarios de funciones que permiten visualizar el estado del arrendatario en cualquier momento y en tiempo real. Deja las tareas de verificación y cambio de estados al sistema, que con ayuda del SIIF

puede verificar los pagos realizados por los arrendatarios en cualquier momento, para tener información actualizada todo el tiempo.

5. Los usuarios de Actividad Comercial pueden visualizar el estado de un local en cualquier momento, así como llevar a cabo un proceso de arrendamiento completo. Pueden cambiar de estado al local correspondiente mientras la aplicación provee las notas automáticas necesarias para avanzar en el proceso.

RECOMENDACIONES

1. Hacer énfasis en un plan de comunicación eficiente que ayude a la continuidad del proyecto en caso de que se cierre la Universidad, ya que es algo que puede suceder con facilidad sin previo aviso y afecta mucho los tiempos.
2. Establecer una mejor comunicación con los usuarios finales, ya que durante el proyecto cambió la jefatura de la Oficina de Actividad Comercial, lo que cortó la comunicación durante un tiempo. Como resultado, el proyecto tuvo algunos cambios de diseño.
3. Contar con más de una vía de comunicación con los interesados, para no tener complicaciones en caso se pierda el contacto en una de estas.
4. Invertir más tiempo en la estimación de tiempos y agregar la holgura necesaria para cumplir con los mismos. De esta forma se tendrá un mejor desarrollo de proyecto y se podrá cumplir con la mayoría de los tiempos estipulados desde el inicio.
5. Implementar una metodología de desarrollo que sea flexible con los requerimientos del cliente. De esta forma, se está listo para cualquier cambio o solicitud nueva, ya que los procesos en la Universidad están propensos a sufrir cambios.

BIBLIOGRAFÍA

1. Angular. *Angular* [en línea]. <<https://angular.io>>. [Consulta: mayo de 2019].
2. _____. *Angular Material UI component library*. [en línea]. <<https://material.angular.io>>. [Consulta: mayo de 2019].
3. ARIAS, Dan. *Hashing in Action: Understanding bcrypt*. [en línea]. <<https://auth0.com/blog/ hashing-in-action-understanding-bcrypt>>. [Consulta: marzo de 2020].
4. BIDELMAN, Eric. *Shadow DOM v1: Componentes web independientes*. [en línea]. <<https://developers.google.com/web/fundamentals/web-components/shadowdom>>. [Consulta: mayo de 2019].
5. Central Intelligence Agency. *The World Factbook*. [en línea]. <<https://www.cia.gov/library/publications/the-world-factbook/fields/204rank.html>>. [Consulta: marzo de 2020].
6. CORY, Rylan. *Web Component Essentials: An introduction to creating reusable user interfaces with Web Components*. Leanpub, 2017. 105 p.
7. Express. *Node.js web application framework*. [en línea]. <<https://expressjs.com>>. [Consulta: marzo de 2020].

8. Google. *Google Trends*. [en línea]. <<https://trends.google.com/trends/explore?cat=31&date=today%205-y&q=Angular>>. [Consulta: mayo de 2019].
9. LOZANO, Edson. *¿Cómo se desarrolla la actividad comercial en la USAC?* [en línea]. <<http://soy.usac.edu.gt/?p=3426>>. [Consulta: abril de 2019].
10. OpenJS Foundation. *Node.js*. [en línea]. <<https://nodejs.org/en>>. [Consulta: febrero de 2020].
11. PUNTAMBEKAR, Anuradha A. *Software Engineering and Quality Assurance*. Pune, India: Technical Publications. 282 p.
12. SAQUETE, Ramón. *Qué son y en qué consisten los Web Components*. [en línea]. <<https://www.humanlevel.com/articulos/desarrollo-web/que-son-y-en-que-consisten-los-web-components.html>>. [Consulta: mayo de 2019].
13. USAC. *Reglamento general para el desarrollo de la actividad comercial en las instalaciones de la universidad de San Carlos de Guatemala*. [en línea]. <http://sitios.usac.edu.gt/wp_auditoria/wpcontent/uploads/2014/10/REGLAMENTO-GENERAL-PARA-EL-DESARROLLO-DE-LA-ACTIVIDAD-COMERCIAL-EN-LAS-INSTALACIONES-DE-LA1.pdf>. [Consulta: mayo de 2019].

APÉNDICES

Apéndice 1. **Lista de requerimientos iniciales**

- Interacción con actividad comercial
 - Registro de asociaciones oficiales: se debe tener un registro de las asociaciones que de verdad están verificadas por medio de documentos.
 - Registro de áreas de influencia: las áreas de influencia corresponden a la dirección que toma el pago del arrendatario dentro del área en la que se encuentra el local. Es útil ingresar nuevas áreas de influencia con la siguiente información:
 - Nombre del área
 - Localización
 - Asociaciones dentro de la misma
 - Porcentaje que corresponde a cada asociación del área.
 - Registro de tipos de local
 - Nombre del tipo de local.
 - Cuota establecida para el tipo de local por metro cúbico.
 - Registro de tipos de gastos
 - Nombre del tipo de gasto.

Continuación apéndice 1.

- Descripción del gasto.
- Registro de locales:
 - Nombre del local.
 - Área de influencia en la que se encuentra el local.
 - Tipo de local.
 - Gastos asociados al local con su respectivo monto dependiendo del tipo de local.
 - Fotografía del local.
- Registro de arrendatario
 - Datos para registrar al arrendatario nuevo.
 - Considerar empresas.
 - Considerar extranjeros.
- Configuración de requerimientos para la solicitud de un local por cada tipo de arrendatario.
- Mapa con las ubicaciones de los locales.
- Control del depósito: cada arrendatario paga como depósito una cantidad equivalente a una mensualidad, la cual debe ser registrada. Si el local es devuelto en condiciones normales, se devuelve el dinero al arrendatario.
- Control de deudores: se debe llevar el control de todos los arrendatarios que tienen dos meses sin pagar y enviar notificaciones para recordar que debe hacerlo.

Continuación apéndice 1.

- Proceso de arrendamiento: para arrendar hay un proceso definido, el cual debe ser visualizado cada vez que el arrendatario se adelanta en el proceso y, de ser descartado, se debe almacenar el motivo.
 - Arrendatario registrado y activo.
 - Local asignado.
 - Contrato asignado. El contrato lo efectúa Caja y procede a enviarlo a Actividad Comercial. En esta versión del sistema no se realizará el contrato por cuestiones de gestión. Se cargará el pdf del contrato al sistema para que sea persistente y consultable rápidamente.

- Manejo de roles de usuario:
 - Usuario administrador
 - ✓ ABC asociaciones.
 - ✓ ABC áreas de influencia.
 - ✓ ABC tipos de local.
 - ✓ ABC tipos de gastos.
 - ✓ ABC locales.
 - ✓ ABC arrendatarios.
 - ✓ ABC tipo de arrendatario.
 - ✓ ABC requisitos.

 - Usuario regular + administrador:

Continuación apéndice 1.

✓ Catálogos:

❖ Locales:

- Disponibles.
- No disponibles.
- En proceso.

❖ Arrendatarios:

- Activos.
- Inactivos.
- Morosos.
- En proceso.

❖ Filtro de búsqueda:

- Por locales.
- Por arrendatarios.
- Por áreas de influencia.
- Por asociaciones.
- Por tipos de gastos.
- Por tipos de local.

- Visualización de historial de pagos por contrato: se debe ver el historial de pago según contrato asignado, con la información del arrendatario que tiene asociado el contrato.

• Cobros

- Control de la cuenta corriente: el sistema tendrá control sobre la cuenta corriente por arrendatario, para que el proceso de pago sea

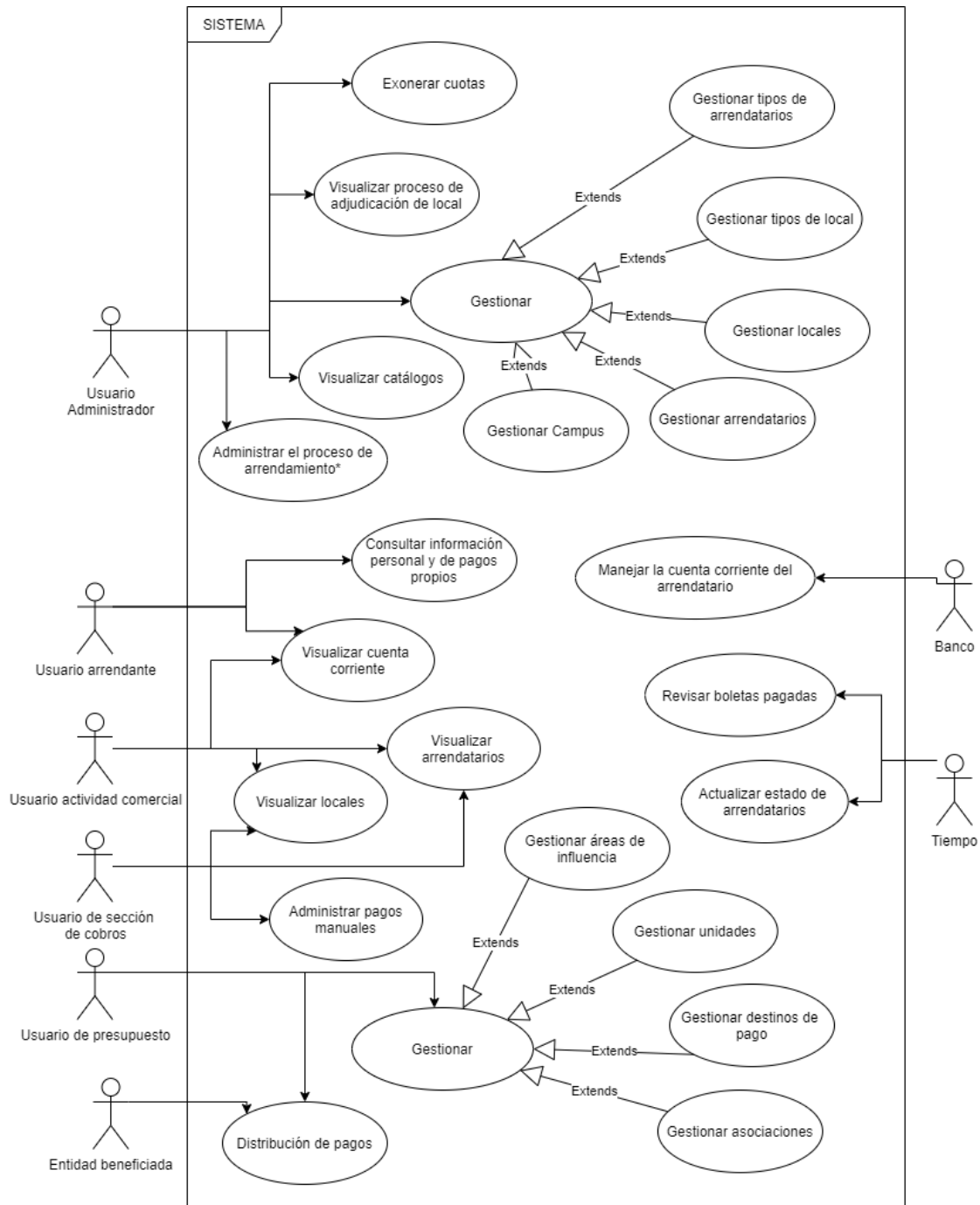
Continuación apéndice 1.

mucho más rápido. Cuando pague el usuario se podrá actualizar su estado dentro del módulo anterior y registrar todos los datos necesarios sobre el pago:

- Fecha
 - Número de orden de pago
 - Contrato pagado
 - Observaciones
- Referencia de sistema existente: existe un sistema que no se usa actualmente, el cual se puede tomar como referencia para llevar a cabo el módulo de agilización de pagos de contratos.
- Presupuesto
 - Distribución de dinero por área de influencia: cada local tiene un área de influencia, la cual tiene el dinero que paga el arrendatario. El dinero se debe distribuir por asociaciones dentro del área.
 - Agrupación de dinero por asociación: se debe agrupar el dinero por asociación para que, cuando alguna llegue a solicitar un cheque, se pueda emitir en base a lo disponible. La agrupación debe desglosar el presupuesto de cada asociación por arrendatario y actualizar su capital en tiempo real.
- Pruebas de cada módulo para verificar su funcionamiento.
 - Manuales para el uso de cada módulo.
 - Capacitación para las personas de actividad comercial.
 - Asistencia para migración de los datos.

Fuente: elaboración propia.

Apéndice 2. Diagrama de casos de uso



Fuente: elaboración propia, empleando draw.io.

ANEXOS

Anexo 1. **Diferencias ingeniería de software y ciencias de la computación**

Software engineering	Computer science
The software engineering is concerned with the practical problems of producing software.	The computer science deals with the theories and methods used by the computers and software systems.
Some knowledge of computer science is necessary for the software engineers to develop the software.	Elegant theories cannot be completely applicable to the software engineering when software solution to complex or real world problem has to be developed.

Fuente: PUNTAMBEKAR, Anuradha A. *Software Engineering and Quality Assurance*. p 2.

Anexo 2. Diez países con más usuarios de internet

RANK	COUNTRY	INTERNET USERS	DATE OF INFORMATION
1	<u>CHINA</u>	730,723,960	est.
2	<u>INDIA</u>	374,328,160	est.
3	<u>UNITED STATES</u>	246,809,221	est.
4	<u>BRAZIL</u>	122,841,218	est.
5	<u>JAPAN</u>	116,565,962	est.
6	<u>RUSSIA</u>	108,772,470	est.
7	<u>MEXICO</u>	73,334,032	est.
8	<u>GERMANY</u>	72,365,643	est.
9	<u>IRAN</u>	67,600,000	est.
10	<u>INDONESIA</u>	65,525,226	est.

Fuente: Central Intelligence Agency. *The World Factbook*.

<https://www.cia.gov/library/publications/the-world-factbook/fields/204rank.html>. Consulta: 25 de marzo de 2020.