UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

Título: Aplicación web para gestión documental de la empresa ATUK utilizando el Framework Mean Stack.

Trabajo de Titulación para optar al título de Ingeniero en Tecnologías de la Información

Autor:
Bravo Capuz Fredy Geovanni

Tutor:
MsC. Jorge Delgado.

Riobamba, Ecuador. ***2024***

DECLARATORIA DE AUTORÍA

Yo, Fredy Geovanni Bravo Capuz, con cédula de ciudadanía 1805430368, autor del trabajo de investigación titulado: Aplicación web para gestión documental de la empresa ATUK utilizando el Framework Mean Stack, certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor (a) de la obra referida, será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, a los 15 días del mes de octubre de 2024.

Fredy Geovanni Bravo Capuz

C.I:1805430368

ACTA FAVORABLE - INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN

En la Ciudad de Riobamba, a 31 de julio de 2024, luego de haber revisado el Informe Final del Trabajo de Investigación presentado por el estudiante **Bravo Capuz Fredy Geovanni** con CC: **1805430368,** de la carrera **Tecnologías de la Información** y dando cumplimiento a los criterios metodológicos exigidos, se emite el **ACTA FAVORABLE DEL INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN** titulado **"APLICACIÓN WEB PARA GESTIÓN DOCUMENTAL DE LA EMPRESA ATUK UTILIZANDO EL FRAMEWORK MEAN STACK"**, por lo tanto se autoriza la presentación del mismo para los trámites pertinentes.

Firmado electrónicamente por:
JORGE EDWIN DELGADO ALTAMIRANO

_____

Ing. Jorge Delgado Mg.
TUTOR

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación Aplicación web para gestión documental de la empresa ATUK utilizando el Framework Mean Stack, presentado por Fredy Geovanni Bravo Capuz, con cédula de identidad número 1805430368, bajo la tutoría de Mg. Jorge Edwin Delgado Altamirano; certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba a los 15 días del mes de octubre de 2024.

Lady Espinoza, PhD.
PRESIDENTE DEL TRIBUNAL DE GRADO

Firma

Ana Congacha,Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO

Firma

María Uvidia, Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO

Firma

C E R T I F I C A C I Ó N

Que, **BRAVO CAPUZ FREDY GEOVANNI con C.I. 1805430368,** estudiante de la Carrera **INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN**, Facultad de **INGENIERÍA** ; ha trabajado bajo mi tutoría el trabajo de investigación titulado " ***APLICACIÓN WEB PARA GESTION DOCUMENTAL DE LA EMPRESA ATUK UTILIZANDO EL FRAMEWORK MEAN STACK***", cumple con el 9 %, de acuerdo al reporte del sistema Anti plagio **TURNITIN**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 03 de octubre de 2024

Firmado electrónicamente por:
JORGE EDWIN DELGADO ALTAMIRANO

_____
Mgs. Jorge Delgado
TUTOR TRABAJO DE INVESTIGACIÓN

DEDICATORIA

Dedico este proyecto a mis padres, quienes han iluminado mi camino con su apoyo incondicional y sacrificios. Con constante aliento ha sido la fuerza que me impulsa a alcanzar mis metas.

A mi querido hermano, aunque la distancia física nos separe, tú presencia siempre está viva en mi corazón. A pesar de los kilómetros que nos separan, tu apoyo y ánimo han sido un puente que une nuestras vidas.

Fredy Bravo

AGRADECIMIENTO

Quiero expresar mis más sinceros agradecimientos a mis padres, cuyo amor incondicional y apoyo han sido la fuerza impulsadora detrás de cada logro en mi vida. Su Sacrificio y dedicación han sido mi inspiración y motivación para alcanzar mis metas académicas. Sin su aliento y orientación, este proyecto no habría sido posible.

A mi tutor, Ing. Jorge Delgado, quiero expresar mi profundo agradecimiento por su orientación experta, su paciencia y su compromiso durante todo el desarrollo de mi proyecto. Sus conocimientos y consejos han sido invaluables para mí, gracias por guiarme con sabiduría y por creer en mi capacidad para llevar a cabo este proyecto

A mis estimados docentes, quienes han compartido su sabiduría y experiencia a lo largo de mi trayectoria académica, les agradezco por su contribución significativa a mi formación. Cada lección y consejo ha sido fundamental en mi crecimiento intelectual.

A mis apreciados compañeros de estudio, agradezco la colaboración y apoyo mutuo que hemos compartido. Juntos, hemos enfrentado desafíos y celebrado éxito, creando una experiencia educativa rica en aprendizaje y amistad.

Fredy Bravo

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente proyecto de investigación tiene como objetivo el desarrollo de una aplicación web para la gestión documental de la empresa ATUK utilizando el framework Mean Stack, la aplicación permite llevar un control sistemático de los procesos, ingreso, creación, mantenimiento, uso y disponibilidad de la documentación empresarial. La metodología de desarrollo ágil aplicada a este proyecto fue Kanban, ya que muestra de una manera gráfica y comprensible las tareas a realizar, de igual manera, permite una mayor iteración de tareas debido a su filosofía de priorizar tareas en curso y terminarlas antes de iniciar con otras nuevas. La aplicación web se desarrolló en dos partes: para el frontend se utilizó el framework Angular versión 14 con el que se logró crear una aplicación SPA responsiva y enfocada a mejorar la experiencia del usuario; para el backend se utilizó Node apoyado del framework Express para el manejo de los diferentes verbos de HTTP y para el control de rutas. Además, se utilizó una base de datos no SQL con Mongo para una carga masiva de información, y que siempre se encuentre disponible para la empresas y usuarios. Cabe recalcar que esta aplicación web se desarrolló completamente con lenguaje JavaScript, utilizando el subconjunto de tecnologías Mean Stack las cuales son en la actualidad muy populares para el desarrollo web moderno.

Finalmente, el sistema web fue evaluado para medir su rendimiento mediante pruebas de carga y estrés utilizando las herramientas JMeter y BlazeMeter. Estas pruebas revelaron un margen mínimo de error en el procesamiento y almacenamiento de datos. Los resultados se presentan en forma de gráficas y árboles de resultados para facilitar su interpretación.

Palabras claves: Aplicación SPA, Gestión documental, Kanban, Mean Stack

ABSTRACT

This research project aims to develop a web application for the document management ofthe ATUK company using the Mean Stack framework; the application allows systematiccontrol of the entry, creation, maintenance, use, and availability of business documentation. The agile development methodology applied to this project was Kanbansince it shows graphically and understandably the tasks to be carried out; in the same way,it allows a more excellent iteration of tasks due to its philosophy of prioritizing tasks in progress and finishing them before starting with new ones. The web application was developed in two parts: for the front, the Angular version 14 framework was used, with which it was possible to create a responsive SPA application focused on improving the user experience. For the backend, Node, supported by the Express framework, was used to manage the different HTTP verbs and to control routes. In addition, a non-SQL database with Mongo was used for a massive load of information, which is always available to companies and users. It should be noted that this web application was developed entirely with JavaScript language, using the subset of Mean Stack technologies, which are currently very popular in modern web development.

Finally, the web system underwent a rigorous performance evaluation through load and stress tests using the JMeter and BlazeMeter tools. These tests revealed a minimal marginof error in data processing and storage, demonstrating the reliability of our system. The results are presented in the form of graphs and result trees for easy interpretation.

Keywords: Document management, Kanban, Mean Stak, SPA Application.

Firmado electrónicamente por:
ANA ELIZABETH MALDONADO LEON

Reviewed by:
Ms.C. Ana Maldonado León
ENGLISH PROFESSOR
C.I.0601975980

CAPÍTULO I. INTRODUCCIÓN

Según el autor (Castro, 2021), la tecnología paso de ser un lujo a una necesidad dentro de un mundo globalizado, la capacidad de gestión que se puede realizar a través de un software permitiendo a las empresas resolver actividades que tomarían semanas o meses en pocos minutos.

QUIPUX es un software de gestión documental desarrollado por personal de la Subsecretaria de Gobierno Electrónico y Registro Civil y que actualmente pertenece al Ministerio de Telecomunicaciones y de la Sociedad de la Información, predispuesto para el sector público. Su estructura se basó en el sistema de gestión documental ORFEO, que permite el control de la organización y trazabilidad de documentos digitales y físicos (Ecuador, 2022).

ATUK es una empresa privada con sede en Ambato que realiza actividades de mejoramiento y redistribución de territorios. Los principales procesos de la empresa ATUK son los siguientes: planimetría, planos arquitectónicos, permisos de construcción, escrituras, línea de fábrica, traspaso de dominio. En la actualidad se gestiona de manera tradicional la documentación de la empresa, sin un apoyo informático y sin la asignación de una persona responsable de controlar el estado de avance de los procesos.

Una aplicación web SPA (Single Page Application) es una aplicación donde todas las pantallas se muestran en la misma página web, no necesita recargar el navegador, técnicamente, con una SPA se generan vistas en lugar de páginas, no existen archivos HTML que se accedan de manera separada al módulo principal. Un claro ejemplo de aplicaciones SPA son Gmail, Outlook, Messenger, entre otras de mensajería instantánea (Desarrollo web, 2016).

Esta investigación tiene como objetivo desarrollar una aplicación web utilizando frameworks web modernos como: Angular y Node, ambos basados en JavaScript. Se pretende consolidar un respaldo de la información documental en una base de datos local de Mongo y distribuirla a través de un API Rest creado con Node y Express. Con este proyecto también se desea dar a conocer las ventajas de una aplicación web desarrollada en su totalidad con lenguaje JavaScript, tanto el lado del servidor como cliente, además, dar a conocer las ventajas de los servicios Rest y Restful para el manejo de información documental.

Finalmente, la aplicación web será evaluada con la herramienta JMeter para medir su rendimiento a través de pruebas de carga y estrés, también se realizará pruebas unitarias al servidor con la herramienta Postman, la cual permite evaluar de manera independiente las peticiones que se realizan al servidor, previniendo de esta manera errores en las actualizaciones y un código mejor estructurado.

1. Planteamiento del Problema y justificación

Según (Rubio, 2019), en muchas empresas se maneja diariamente una gran cantidad de información documental sobre: clientes, proyectos, presupuestos, entre otros. Se podría considerar que esos datos constituyen el activo más importante de la empresa y, según un informe de IDC publicado por DocuWare argumenta que, cualquier institución desaparecería en menos de tres semanas si esa información fuera extraviada o destruida.

ATUK es una creciente empresa ubicada en la ciudad de Ambato que maneja una gran cantidad de información documental, no existe un respaldo electrónico de la información dificultando conocer el estado actual en el que se encuentra un trámite, por lo que, es común encontrar problemas como: falta de seguimiento de proyectos, extravío de información, desorganización, duplicidad, además del retardo en la entrega de resultados que han provocado inconvenientes tanto en la empresa como para sus clientes. Por lo que, se propone el desarrollo de una aplicación web, con el propósito de mejorar la gestión documental en los procesos: planimetría, planos arquitectónicos, permisos de construcción, escrituras, línea de fábrica, traspaso de dominio.

Basado en lo anterior, se desarrollará una aplicación web que gestione la documentación empresarial y mejore los tiempos de respuesta con respecto a la entrega de resultados. La interfaz gráfica de la aplicación se desarrolló con Angular, se optó por este framework debido a su arquitectura MVVM que permite tener en una sola página HTML toda la información precargada del servidor, minimizando los tiempos de respuesta. Por el lado del backend, se trabajó con una base de datos NoSQL con Mongo, ya que permite almacenar gran cantidad de información en formato BSON, también se desarrolló un servicio Rest con Node y Express para la comunicación entre el cliente y servidor a través de URLs. Toda la información obtenida estará en formato JSON, un formato autodescriptivo y fácil de entender y que actualmente es la mejor alternativa en proyectos que entregan y reciben grandes bytes de información.

1.1 Formulación del problema

¿Cómo el uso del Framework Mean Stack incidirá en el rendimiento de la aplicación web para la gestión documental de los procesos de la empresa ATUK?

1.2 Objetivos

1.2.1 Objetivo general

Implementar una aplicación web para gestión documental de la empresa ATUK utilizando el framework Mean Stack.

1.2.2 Objetivos específicos

- Investigar sobre el desarrollo de aplicaciones web para gestión documental utilizando el framework Mean Stack.
- Desarrollar una aplicación web para la gestión documental utilizando el framework Mean Stack.
- Evaluar el rendimiento de la aplicación web utilizando la herramienta Apache JMeter

CAPÍTULO II. MARCO TEÓRICO

2.1 Gestión documental

Ninguna organización es igual a su competencia, destacar de las demás es la esencia de toda empresa, por lo que, ningún sistema de gestión documental servirá para cumplir los mismos objetivos de una organización. El autor (Gallo, 2011) un sistema de gestión documental (SGD) ayudará en gran medida al aumento de la eficiencia y productividad, sin embargo, se debe conocer que el termino gestión documental también engloba otros conceptos relacionados entre sí.

- **Gestión documental. –** guía de procesos que permiten una mayor coordinación y control de aspectos que relacionan actividades como: creación, recepción, organización, acceso y emisión de documentos.
- **Gestión de la información. –** procesos que permiten la obtención de información y sus características.
- **Gestión del conocimiento.** – procesos por medio de los cuales se convierte a la información en un activo crítico y la predispone a los usuarios.
- **Gestión de archivos.** – preserva la información de un archivo de carácter administrativo o histórico.
- **Gestión de contenidos.** – mantiene actualizados los contenidos de una página web.
- **Gestión de calidad.** – conjunto de procedimientos que permiten mantener en orden y actualizados los procesos de calidad de los servicios y productos.

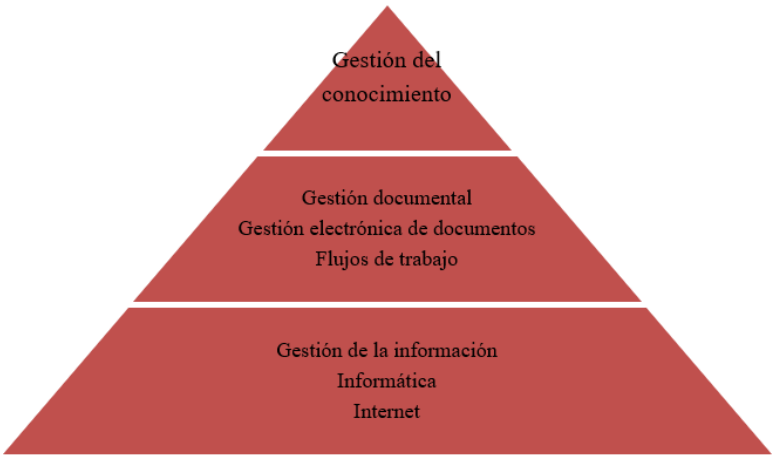


Figura 1: Pirámide esencial de la gestión de documentos

Elaborado por: Bravo Fredy, 2024

2.2 Aplicación web

Según el autor (Ollivier & Gury, 2016), en los 90s la web había nacido para consultar información del mundo y compartirla con usuarios que disponían una computadora, durante esa década las páginas web eran HTML estático y en pocas ocasiones se añadían gráficos de baja calidad. En la siguiente década, las páginas web se transformaban en medios participativos añadiendo formularios de ingreso de datos, estos eran enviados a servidores remotos, los mismos que remitían al usuario información. Cuesta imaginar el tiempo que duraban esas peticiones en aquella época, tendiendo en cuenta que si se ingresaba mal una petición esta debía ser validada desde un servidor ubicado a kilómetros de distancia del usuario. A medida que la web evolucionaba fue apareciendo nuevas tecnologías integradas en los navegadores, entre ellas el nacimiento de JavaScript, que permitían validar formularios antes de ser enviados, varios años más tarde la web revolucionó en lo que ahora conocemos como web 2.0, con más operabilidad y enfocada al usuario, es así como ahora se tiene las aplicaciones web SPA (Single Page Application) o por su traducción aplicaciones de una sola página; por ejemplo: Gmail, Hotmail y cualquier otro servicio de mensajería.

Figura 2: Páginas web estáticas de la web 1.0

Obtenido de: https://www.timetoast.com/timelines/evolucion-de-la-web-c62d2386-5ad0-4bbf-948d-f579fd1caf92

2.2.1 Single Page Application

En español se interpreta como Aplicación de Página Única y se diferencia de una página web común en la relación que existe entre el servidor y el navegador. Una página web convencional tiene una estructura convencional, formada por muchas páginas que comparten diseños similares, pero con diferente contenido, se accede a ella a través de enlaces que se abren en la misma pestaña del navegador o en diferentes, algunas páginas implementan AJAX como complemento para refrescar la visualización de información sin tener que recargar la página.

Por otro lado, una SPA solo se compone de una página y diferentes vistas, el usuario navega por estas vistas, pero se mantiene dentro de la misma estructura base de la página web, el despliegue de información es casi instantáneo a diferencia de un sitio web común, el objetivo de una SPA es mantener cargada la información en un template y exponerlos según el usuario lo requiera, igual que un API Rest.

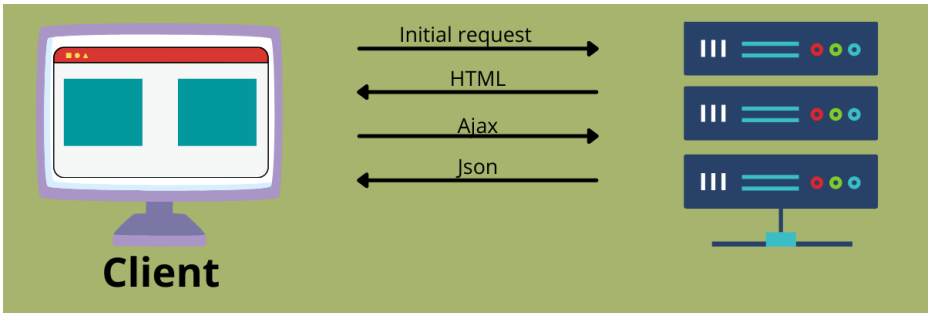


Figura 3: Single Page Application estructura

Obtenido de: https://geekflare.com/single-page-applications/

2.2.2 Angular

Según el autor (Puciarelli, 2020), Angular es un framework de código abierto que utiliza sintaxis de JavaScript, se utiliza frecuentemente para el desarrollo de páginas web SPA. Para poder utilizar Angular se debe tener conocimiento de TypeScript, que es básicamente JavaScript con mejoras, lo que lo vuelve un lenguaje de tipado fuerte para el frontend, y que utiliza parámetros de ECMAscript lo que lo vuelve ideal para el desarrollo de aplicaciones web escalables.

2.2.2.1 Arquitectura de Angular

Angular se compone de: módulos, componentes, servicios y directivas

Módulos. –

Permiten la reorganización del código para la gestión del desarrollo de aplicaciones complejas y que son reutilizables en cualquier contexto. Además, proporcionan módulos nativos como: HTTP, Browser, Forms, Reactive Forms, los cuales permiten la interacción directa con la aplicación a partir del concepto two way data-binding.

Componentes. –

Es básicamente la estructura de la aplicación web, en un componente se encuentran archivos que manipulan la lógica, estructura y estilos de las vistas, aquí se desarrolla la experiencia de usuario.

Servicios. –

Proporciona la funcionalidad a los componentes, también sirven para comunicación externa; por ejemplo: para consumir un servicio web o conectarse a un API REST.

Directivas. –

Manipulan directamente el DOM (Modelo de Objeto de Documento) del HTML, también se los conoce como atributos para las etiquetas HTML.

Figura 4: Arquitectura MVC de Angular

Obtenido de: https://naveenpete.wordpress.com/2016/09/07/mvc-mvvm-and-angular/

2.3 Representational State Transfer (REST)

API es una interfaz de programación de aplicaciones usada dentro del desarrollo móvil y web, funciona como un puente de comunicación entre el servidor y el usuario, de manera que comparten información a través de solicitudes, un API también se podría considerar como un mediador de recursos, ya que establece políticas de consumo e ingreso de datos a través de módulos de autenticación, mismos que previenen el uso inapropiado de información. Por otro lado, REST forma parte de la arquitectura de un API ya que a través de métodos y funciones procesa las peticiones que llegan desde el usuario.

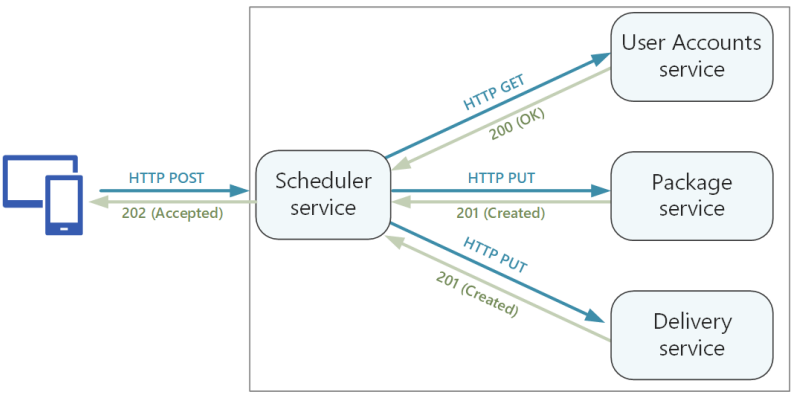


Figura 5: Arquitectura de un API Rest

Obtenido de: https://learn.microsoft.com/es-es/azure/architecture/microservices/images/api-design.png

La información resultante se comparte en formato de JSON y debe cumplir con algunos parámetros propios de la arquitectura de REST:

- Un API Rest funciona de manera independiente, sin importar quien consuma este servicio web, deberá hacerlo a través de peticiones HTTP.
- Los estados de sesión siempre se mantienen de lado del cliente.
- Las peticiones se almacenan en cache.
- Se puede realizar varias peticiones a la vez, gracias al uso de memoria cache.
- Permite el desacoplamiento entre cliente y servidor, permitiendo mejorar la escalabilidad.

Finalmente, la información del API Rest necesita ser enviada desde el servidor hacia el cliente a través de métodos, en SQL se lo denomina CRUD, 'abreviatura de Create, Read, Update, Delete', y que permiten manipular la información de una base de datos relacional, sin embargo, dentro de HTTP existen protocolos que realizan estas funciones a través de prefijos (Heredia, 2022).

En la tabla 1, se muestra una comparación entre métodos SQL y prefijos HTTP

Tabla 1: Prefijos de HTTP

Petición	Lenguaje SQL	Prefijo HTTP
Crear	Create	Post
Leer	Read	Get
Actualizar	Update	Put / Patch
Eliminar	Delete	Delete

Elaborado por: Bravo Fredy, 2024

2.4 Nodejs

Es un entorno multiplataforma para ejecución del lenguaje programación JavaScript por el lado del servidor, al ser de código abierto puede ser descargado e instalado por cualquier persona sin

pagar licencias de uso. La sintaxis de Node se basa en las especificaciones de ECMAScript y su estructura se orienta a eventos, también hace uso del motor de Google para ejecución de JavaScript en el lado navegador.

Empresas como Netflix, LinkedIn, eBay utilizan los recursos de Node para la ejecución de sus plataformas, debido a su velocidad y fácil implementación a través del manejador de paquetes NPM (Node Package Manager) se pueden añadir y organizar dependencias que a largo plazo permiten el crecimiento de una aplicación Node.

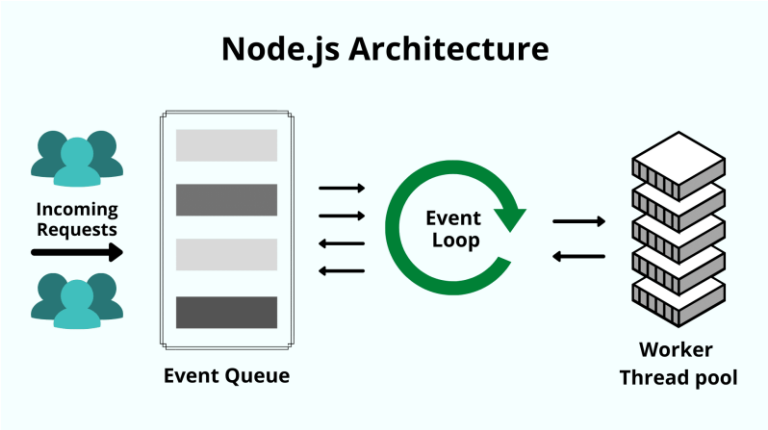


Figura 6: Arquitectura de Node

Obtenido de: https://kinsta.com/wp-content/uploads/2021/03/Nodejs-Architecture-1024x576.png

2.4.1 Node como servidor web

Node a más de funcionar como una capa de servicios de aplicaciones enfocadas a la web, también es utilizado en redes para el manejo de estándares como HTTP, TCP, DNS, TLS/SSL y UDP. En cuanto al manejo de la web, Node utiliza frameworks como complementos para el desarrollo de aplicaciones escalables de una sola página, Express es un framework que combina el uso de sintaxis de JavaScript para reforzar el desarrollo de APIs, mejorando la estructura de las peticiones HTTP y encaminándolas en rutas (URLs) (Pucciarelli, 2020).

2.5 Express

Express es un framework web utilizado en conjunto con Node para el desarrollo de aplicaciones web escalables y de fácil mantenimiento, tiene una buena reputación en la comunidad de desarrolladores por su alta confiabilidad a la hora de integrarlo como librería en proyectos que utilizan API Rest para la distribución de información.

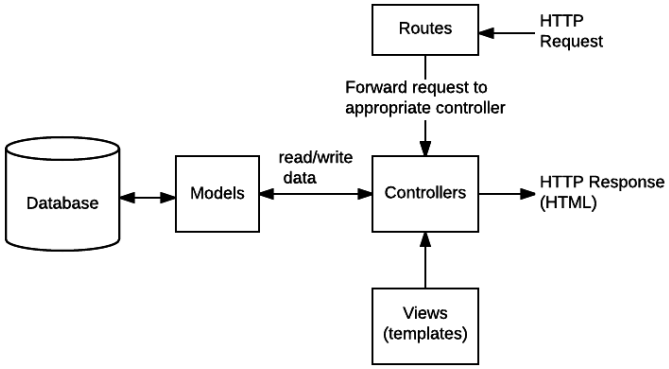


Figura 7: Rutas y Controladores con Express

Obtenido de: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/routes

Cuando Express está en ejecución, empieza a escuchar requerimientos del frontend a traves de rutas y las resuelve por medio de sus middlewares, para finalmente encaminar una respuesta a través de módulo de rutas. Para su instalación se requiere tener previamente instalado una versión de Node en la computadora, después desde una terminal se ejecuta el comando **$npm install -g express** para que se instale de manera global y pueda ser utilizado en cualquier poryecto futuro (Mardan, 2014).

2.6 Base de datos no relacionales

Según (Andrey & Rueda, 2015) la tendencia en el uso de bases de datos NoSQL surge a partir del inicio de la Web 2.0, con aplicaciones más sofisticadas que prefieren el uso dinámico de datos ante un estructurado modelo de tablas que ralentizaba las peticiones, esto significa que, en una aplicación web que utiliza base de datos nosql las peticiones se resuelven a nivel de aplicación y no a nivel de base de datos. Las bases de datos no relacionales tienen modelos que difieren de los modelos relacionales; estos son:

- Clave-valor
- Orientado a documentos
- Familia de columnas
- Grafos

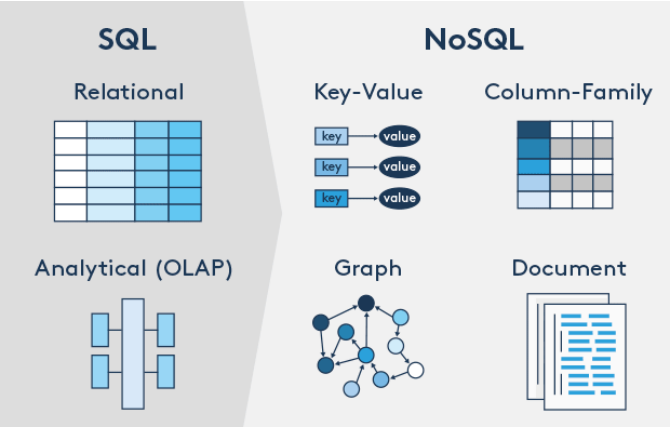


Figura 8: Diferencias entre SQL y NoSQL

Obtenido de: https://dbaexperts.tech/wp/wp-content/uploads/2022/02/NoSQL4.png

2.6.1 MongoDB

Es un gestor de base de datos no relacionales, lo que significa que almacena objetos y no filas como en una base de datos relacional, el formato de almacenamiento es JSON.

Un objeto de tipo JSON está compuesto por pares (clave y valor), encerradas dentro de llaves entre comillas y separados por comas; ejemplo {“nombre” : ”Freddy”, “apellido” : “Bravo”}. No obstante, dentro de una colección de datos masivos se pueden encontrar arrays delimitados por corchetes y se los conoce como objetos anidados.

En la tabla 2, se muestra los tipos de datos permitidos en un documento de Mongo.

Tabla 2: Datos permitidos en los objetos JSON en Mongo

Tipo de dato	**Ejemplo**
String	“nombre” : “Fredy”
int	“edad” : 25
real	“promedio” : 2.541
Array	“num_Telefono” : [“0985120214”, “0984521471”]

Objeto	“cliente” : {
	“_id” : 4d1s22c14f6331s45s,
	“nombre” : “Tomas”,
	“apellido” : “Edison”
	}

Boolean	“flag” : true
null	“usuario” : null

Elaborado por: Bravo Fredy, 2024

2.7 Postman

Post es una herramienta utilizada para realizar peticiones y testear APIs Rest propias o de terceros, en un principio Postman era una herramienta que se ejecutaba en el navegador de Google Chrome, sin embargo, gracias a los avances tecnológicos paso de ser una extensión a ser una aplicación que se ejecuta en diferentes sistemas operativos como: Windows, Linux o Mac (Muradas, 2019).

Postman tiene una serie de métodos que permiten testear un API Rest a través de métodos HTTP conocidos, entre los cuales se destacan:

- GET. – para obtener información
- POST. – para agregar información
- PUT. – para reemplazar datos
- PATCH. – para actualizar la información
- DELETE. – para eliminar información

2.7.1 Códigos de estado

Las interacciones que se producen entre el navegador y una página web se basan en el principio de solicitud-respuesta, cualquier petición que se realiza desde el navegador al servidor, este devolverá una respuesta HTTP.

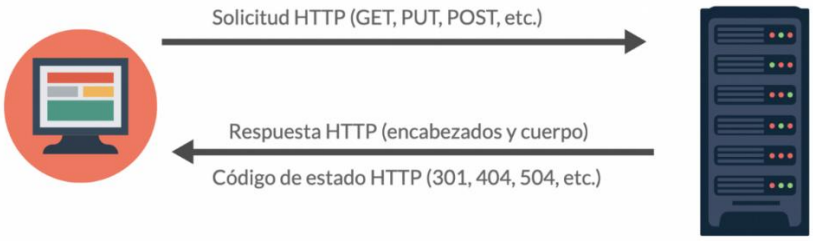


Figura 9: Estados HTTP del cliente al servidor

Obtenido de: https://www.siteground.es/kb/codigos-error-http-explicados/#%C2%BFQue_es_un_codigo_de_estado_HTTP

Las respuestas que llegan del servidor están separadas en dos partes: el cuerpo contiene la parte visual de la solicitud, y los encabezados que contienen metadatos sobre las respuestas del servidor. Los códigos de estado HTTP se dividen en 3 dígitos. El primer digito significa la

categoría de respuesta recibido, y los dos dígitos restantes definen una respuesta especifica por parte del servidor (SiteGround Web, s.f.)

Existen cinco categorías para agrupar las respuestas según su propósito:

- Código 1XX. – respuesta de solicitud en curso.
- Código 2XX. – respuesta afirmativa, o se está gestionando.
- Código 3XX. – respuesta de redirección.
- Código 4XX. – respuesta negativa, expresan errores o problemas para completar una solicitud desde el navegador (cliente).
- Código 5XX. – respuesta negativa, expresan errores o problemas desde el servidor de la web.

Figura 10: Código de estado HTTP 404

Obtenido de: https://blog.sinapsis.agency/definiciones-error-404/

2.8 Apache Jmeter

Es una herramienta utilizada para testing de funcionalidades de una aplicación web, permite ejecutar un archivo .jmx a través de líneas de comando o de la propia interfaz gráfica de Jmeter. Su funcionalidad se basa en peticiones de tipo HTTP, FTP, LDAP, para cada petición se extrae datos y los representa a través de listeners o escuchas (Marco de Desarrollo de la Junta de Andalucía, s.f.).

2.8.1 Prueba de carga básica con Jmeter

JMeter tiene un componente denominado Test Plan, aquí se definen parámetros relacionados con las pruebas de carga tales como: requisición, tipo de reportes a generar, entre otros parámetros. Jmeter posee una GUI para una visualización más amigable con el usuario, aunque parezca más fácil de utilizar tiene algunas desventajas, siendo la más notable el uso de memoria de la PC, por lo que, es recomendable utilizar el Shell de Jmeter y realizar las pruebas de rendimiento mediante líneas de comando.

Se debe conocer los siguientes conceptos para poder entender como diseñar un correcto Test Plan con Jmeter:

- **Test Plan:** registra los datos devueltos por el servidor en cada muestra, representa la raíz del árbol.
- **Thread group:** controlan la cantidad de subprocesos utilizados para la prueba de JMeter.
- **Samplers**: envían solicitudes continuas al servidor esperando respuestas.
- **Logic Controllers:** personaliza la lógica de Jmeter para saber cuántas solicitudes se requieren enviar.
- **Listeners:** permite el acceso a la información que JMeter obtiene de las pruebas mientras se ejecuta el sistema.
- **Timers:** definen el tiempo que transcurre entre cada evaluación.

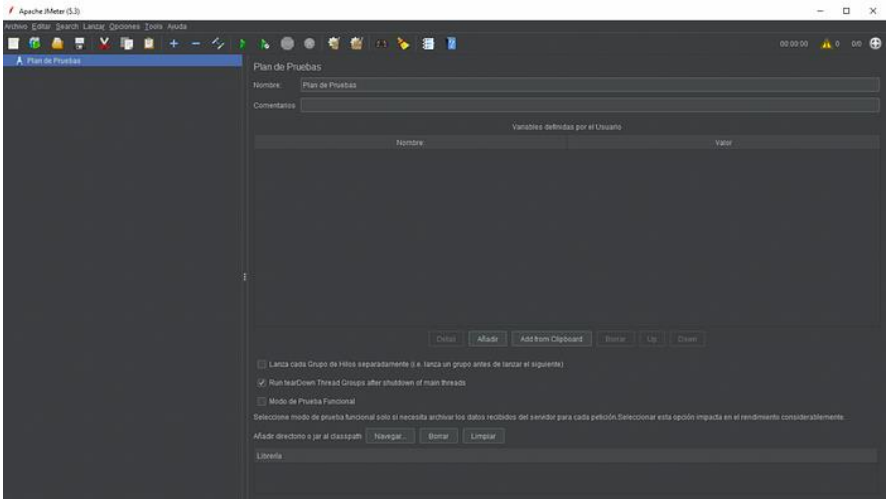


Figura 11: GUI de Apache Jmeter

Obtenido de: https://medium.com/crux-learnings/apache-jmeter-una-opci%C3%B3n-para-pruebas-automatizadas-8a625d45828

2.8.2 Ng-Zorro

Según el autor, (NG-ZORRO, s.f.) Una biblioteca de componentes de interfaz de usuario angular de clase empresarial basada en Ant Design, todos los componentes son de código abierto y de uso gratuito bajo la licencia MIT.

Características:

- Un lenguaje de diseño de UI de clase empresarial para aplicaciones Angular.
- Más de 60 componentes Angular de alta calidad listos para usar.
- Escrito en TypeScript con tipos completamente definidos.
- Admite modo OnPush, alto rendimiento.
- Potente personalización del tema en cada detalle.
- Soporte de internacionalización para docenas de idiomas.

Compatibilidad

- Navegadores modernos, compatibilidad con navegadores
- Representación del lado del servidor
- Electrón

Soporte Angular

ng-zorro-antd mantiene la misma versión principal con @angular/core, ahora es compatible con Angular 18.0.0.

Especificación de diseño

ng-zorro-antdsincroniza las especificaciones de diseño con Ant Design de forma regular, puedes consultar el registro en línea.

CAPÍTULO III. METODOLOGIA

3.1 Metodología

En el desarrollo de la aplicación web se consideró aplicar una investigación tipo bibliográfica extrayendo información sobre la metodología ágil Kanban enfocada al desarrollo de software. También se evaluará el rendimiento a través de pruebas de carga y estrés con la herramienta Apache Jmeter, cumpliendo con la problemática expuesta a lo largo de la investigación.

3.2 Tipo y diseño de la investigación

Esta investigación tiene un enfoque cuantitativo, basado en el hecho de que se evaluará el rendimiento de la aplicación web con Apache Jmeter, esta herramienta donde nos permite extrae datos de diferentes métricas como: márgenes de error, rendimiento, desviación estándar, entre otras que aportan a la escalabilidad de la aplicación.

3.2.1 Según la fuente de investigación

Investigación bibliográfica: Para desarrollo del proyecto se recopilará información de libros, revistas científicas, foros, conferencias, tesis de grado y páginas web.

3.2.2 Según el objeto de estudio

Investigación aplicada: De tipo aplicada ya que se planea brindar una solución tecnológica al problema de gestión documental de los procesos: planimetría, planos arquitectónicos, permisos de construcción, escrituras, línea de fábrica y traspaso de dominio de la empresa ATUK, de

igual manera se ofrece una solución basada en los nuevos parámetros de desarrollo web, utilizando herramientas tecnológicas modernas.

3.2.3 Según el tipo de variable

Es una investigación cuantitativa ya que se evaluará el rendimiento de la aplicación web con las métricas que la herramienta Apache Jmeter proporciona, como son: márgenes de error, rendimiento, desviación, cantidad de bytes ingresados y enviados.

3.3 Unidad de análisis

Se establecerá los criterios en base a las cargas simuladas durante la evaluación del rendimiento, para las pruebas de carga y estrés se considerarán hasta 30 peticiones enviadas simultáneamente desde el cliente hasta el servidor, con el fin de encontrar el punto en el que la aplicación web pierde rapidez y aumenta la latencia, todas estas pruebas se las lleva a cabo con Apache Jmeter.

3.4 Población y muestra

Debido a que se evaluará el rendimiento de la aplicación con la herramienta Apache Jmeter, se consideró una población finita, para las simulaciones se utilizara una simulación con peticiones realizadas al servidor.

3.5 Técnicas e instrumentos de recolección de datos

Para llevar a cabo la recolección de datos, se va a utilizar técnicas e instrumentos que se detallan en la Tabla 3.

Tabla 3: Técnicas e instrumentos de recolección de datos

TECNICAS	INSTRUMENTO
Entrevista	Guía de entrevista
Focus Group	Perfil de usuario

Elaborado por: Bravo Fredy, 2024

3.5.1 Entrevista

Con el objetivo de recopilar información detallada sobre el estado actual del proceso de desarrollo y los requisitos para la aplicación web, se llevó a cabo una entrevista con el gerente de la empresa ATUK. Durante esta sesión, se utilizó un enfoque visual y ágil, incorporando el uso de un tablero Kanban para visualizar y gestionar los requisitos del proyecto de manera más eficaz.

3.5.2 Focus Group

Utilizando la técnica del focus group, se obtuvo información valiosa en una primera reunión con el grupo asignado para el desarrollo de la aplicación web de la empresa Atuk. Posteriormente, se llevaron a cabo reuniones periódicas a lo largo del desarrollo del proyecto para seguir recabando opiniones y mejorar la investigación.

3.6 Técnicas de análisis e interpretación de la información

Una vez recopilada la información de la entrevista, se procedió a realizar un análisis para definir los métodos de desarrollo de la aplicación web, tomando en cuenta varios criterios para asegurar su correcto funcionamiento. Se utilizó el tablero Kanban como herramienta principal para la recolección de información, aprovechando su capacidad para definir los requisitos del sistema conforme a la metodología Kanban. Posteriormente, se evaluó el rendimiento de la aplicación web utilizando Apache JMeter.

3.7 Identificación de variables

3.7.1 Variable independiente

Aplicación web.

3.7.2 Variable dependiente

Rendimiento de la aplicación web para la gestión documental de la empresa ATUK.

3.8 Operacionalización con variables

Tabla 4: Operacionalización con variables

PROBLEMA	TEMA	OBJETIVOS	VARIABLES	INDICADORES

¿Cómo el uso del framework Mean Stack incidirá en el rendimiento de la aplicación web para la gestión documental de los procesos de la empresa ATUK?	Aplicación web para gestión documental de la empresa ATUK utilizando el framework Mean Stack.	**GENERAL**
		Implementar una aplicación web para gestión documental de la empresa ATUK utilizando el framework Mean Stack.
			INDEPENDIENTE
			Aplicación web
				- Número de módulos de la aplicación
				- Número de Servicios
				  - Planimetría
				  - Planos Arquitectónicos
				  - Escritura
				  - Permisos de construcción
				  - Línea de fábrica
				  - Traspaso de dominio

		ESPECIFICOS • Investigar sobre el desarrollo de aplicaciones web utilizando el framework Mean Stack. • Desarrollar una aplicación web para la gestión documental utilizando el framework Mean Stack. • Evaluar el rendimiento de la aplicación web utilizando la herramienta Apache JMeter.	**DEPENDIENTE** Rendimiento de la aplicación web para la gestión documental de la empresa ATUK.	Subcriterios de rendimiento según la herramienta Jmeter: - Número de peticiones atendidas - Tiempo de respuesta del servidor - Cantidad de bytes enviados y recibidos - Margen de error

Elaborado por: Bravo Fredy, 2024

3.9 Desarrollo de la aplicación web

Para el desarrollo de la aplicación web para la gestión documental se utilizó la metodología ágil Kanban dividida en seis etapas. Cada etapa está representada en una tabla de tres columnas, cada columna representa el estado de la tarea en que se encuentran las fases de desarrollo.

3.9.1 Primera etapa del desarrollo web

En la tabla 5 se muestra la primera etapa del desarrollo de la aplicación web, se añadieron las actividades planificadas en la columna tareas pendientes, y se añadió como tarea terminada la revisión teórica del desarrollo web con el framework MEAN Stack.

Tabla 5: Estado de las actividades planificadas en la primera etapa

PENDIENTES	EN PROCESO	TERMINADAS
Realizar los wireframes de la aplicación web	Realizar la toma de requerimientos funcionales y no funcionales	Revisión teórica de la documentación para el desarrollo web con Mean Stack
Diseño de la arquitectura del frontend	Elaborar los diagramas de caso de uso	
Modelado de datos		

Elaborado por: Bravo Fredy, 2024

3.9.1.1 Análisis de requerimientos funcionales y no funcionales

Como se ilustra en la Figura 12, se elaboró un diagrama de casos de uso para la gestión de usuarios de la aplicación web. En este diagrama, se asignaron diferentes roles: el gerente tiene control total sobre todas las operaciones de la aplicación, el tramitador dispone de permisos limitados sin capacidad para eliminar datos, y la secretaria tiene acceso únicamente para realizar consultas.

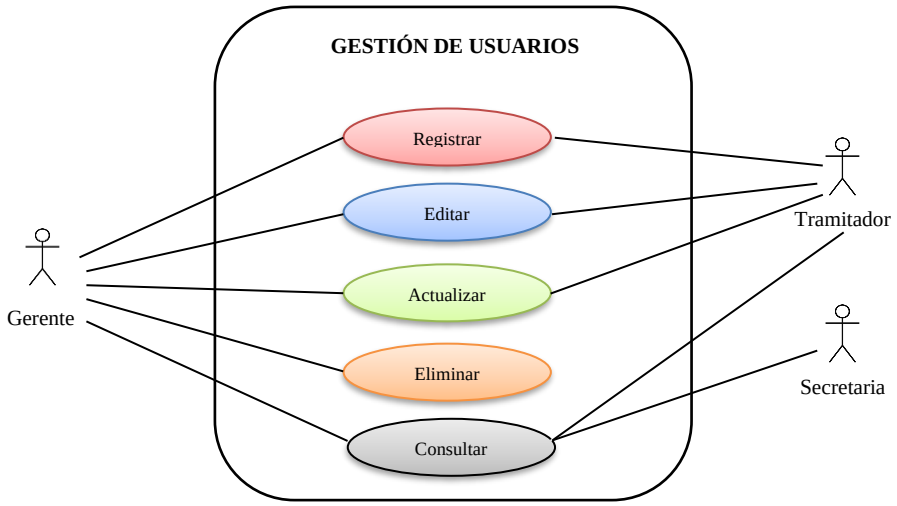


Figura 12: Diagrama de casos de uso para la gestión de usuarios

Elaborado por: Bravo Fredy, 2024

Así mismo, se realizó un diagrama de casos de uso para la gestión documental de la aplicación web. Basado en los roles asignados anteriormente, se muestra en la Figura 13 la distribución de las acciones que cada usuario tiene permitido realizar, siendo el gerente quien tiene todos los permisos de la aplicación; mientras que el tramitador está restringido en cuanto al registro de pagos por servicios y no tiene acceso a la caja. Finalmente, la secretaria no tiene autorización para crear clientes o servicios, ni para llevar a cabo las funciones del tramitador.

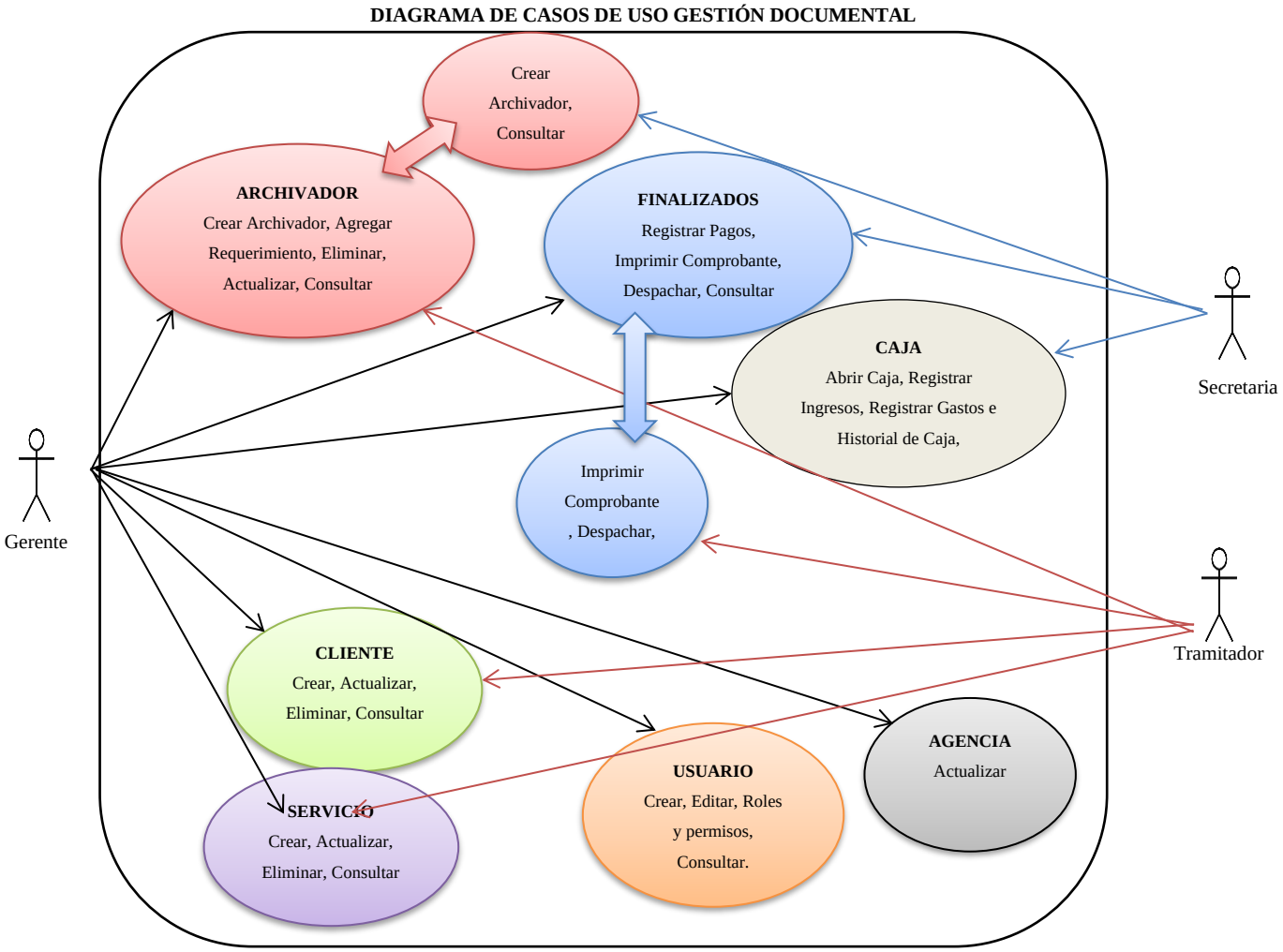


Figura 13: Diagrama de casos de uso para la gestión documental

Elaborado por: Bravo Fredy, 2024

3.9.1.1.1 Requerimientos funcionales

En la tabla 6 se muestra los requisitos funcionales obtenidos directamente de la empresa ATUK en base a sus funciones.

Tabla 6: Requerimientos funcionales de la aplicación web

Id	Requerimiento	Descripción	Prioridad
RF1	Identificación y autenticación	La aplicación permite el ingreso a usuarios registrados y genera un token de acceso que caduca en 8 horas.	Alta

RF2	CRUD Archivadores	La aplicación realizará las operaciones CRUD para Archivadores.	Alta
RF3	CRUD Servicios	La aplicación realizará las operaciones CRUD para Servicios.	Alta
RF4	CRUD Clientes	La aplicación realizará las operaciones CRUD para Clientes.	Alta
RF5	CRUD Usuarios	La aplicación realizará las operaciones CRUD para Usuarios.	Alta
RF6	CRUD Finalizados	La aplicación realizará las operaciones CRUD de listados de edición y exportación de archivos.	Alta
RF7	CRUD Caja	La aplicación realizará las operaciones CRUD para la Caja.	Alta
RF8	Asignación de roles	La aplicación debe asignar roles y mostrar una vista diferente para cada usuario con rol asignado	Alta
RF9	Importación y exportación de datos externos	La aplicación debe permitir cargar datos y también exportarlos	Alta

Elaborado por: Bravo Fredy, 2024

3.9.1.1.2 Requerimientos no funcionales

En la tabla 7 se muestran los requerimientos no funcionales en base a la apariencia y criterios de funcionalidad de la aplicación web

Tabla 7: Requerimientos no funcionales de la aplicación web

Id	Requerimientos	Descripción	Prioridad
RNF1	Versatilidad	La aplicación web se ejecuta en cualquier navegador	Alta
RNF2	Apariencia	El sistema web es responsivo	Alta

RNF3	Documentación	El sistema proporciona ayuda al usuario	Alta

Elaborado por: Bravo Fredy, 2024

3.9.2 Segunda etapa del desarrollo web

La segunda etapa del desarrollo web se enfoca en el diseño de la apariencia de la aplicación web, para este propósito se diseñó con la herramienta Pencil los wireframes de las vistas principales de la aplicación web. Cabe resaltar que el diseño puede variar con respecto al resultado final de la aplicación, ya que se busca ocupar menos recursos visuales y enfocarse más en la funcionalidad.

En la tabla 8 se muestran las actividades planificadas en la segunda etapa de desarrollo, se ubican en la columna de tareas terminadas las que se encontraban en proceso y se añaden nuevas a la columna de tareas pendientes.

Tabla 8: Estado de las actividades planificadas en la segunda etapa

PENDIENTES	EN PROCESO	TERMINADAS
Instalación de librerías y dependencias para el desarrollo del backend	Realizar los wireframes de la aplicación web	Realizar la toma de requerimientos funcionales y no funcionales
Desarrollo del backend	Diseño de la arquitectura del frontend	Elaborar los diagramas de caso de uso
Pruebas unitarias al backend con Postman	Modelado de datos	

Elaborado por: Bravo Fredy, 2024

3.9.2.1 Desarrollo de los wireframes de la aplicación web

- Login

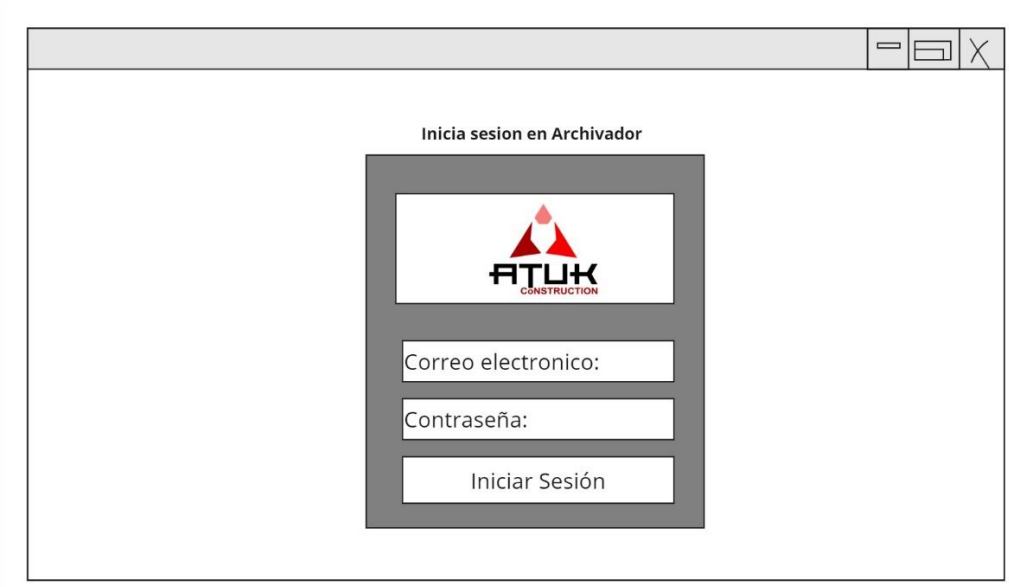


Figura 14: Diseño del Login

Elaborado por: Bravo Fredy, 2024

- Dashboard

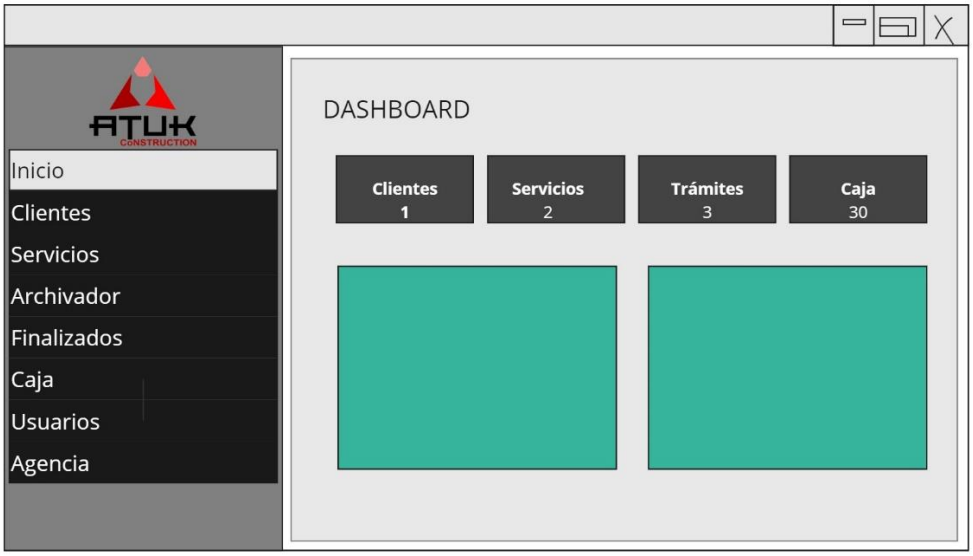


Figura 15: Diseño del Dashboard

Elaborado por: Bravo Fredy, 2024

- Clientes

Figura 16: Diseño de la vista crear cliente

Elaborado por: Bravo Fredy, 2024

- Servicios

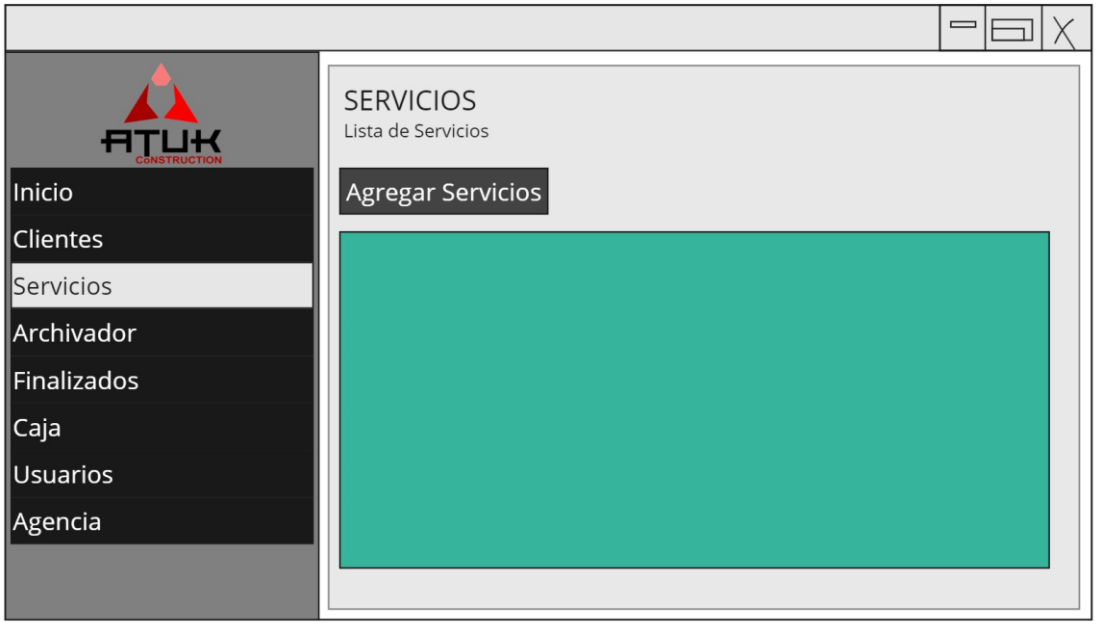


Figura 17: Diseño de la vista crear servicios

Elaborado por: Bravo Fredy, 2024

- Archivadores

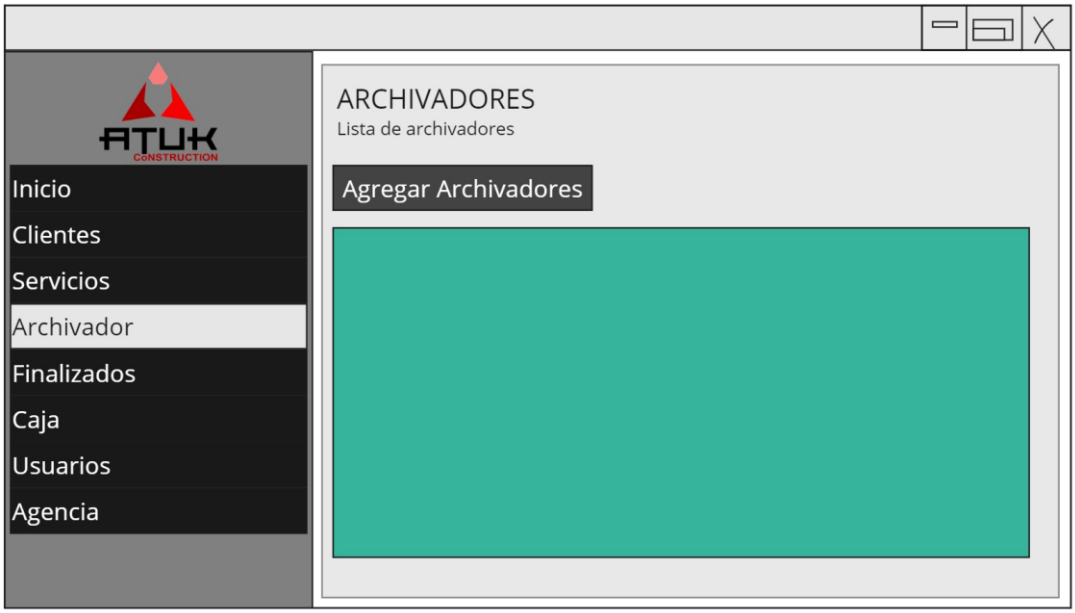


Figura 18: Diseño de la vista crear archivadores

Elaborado por: Bravo Fredy, 2024

- Finalizados

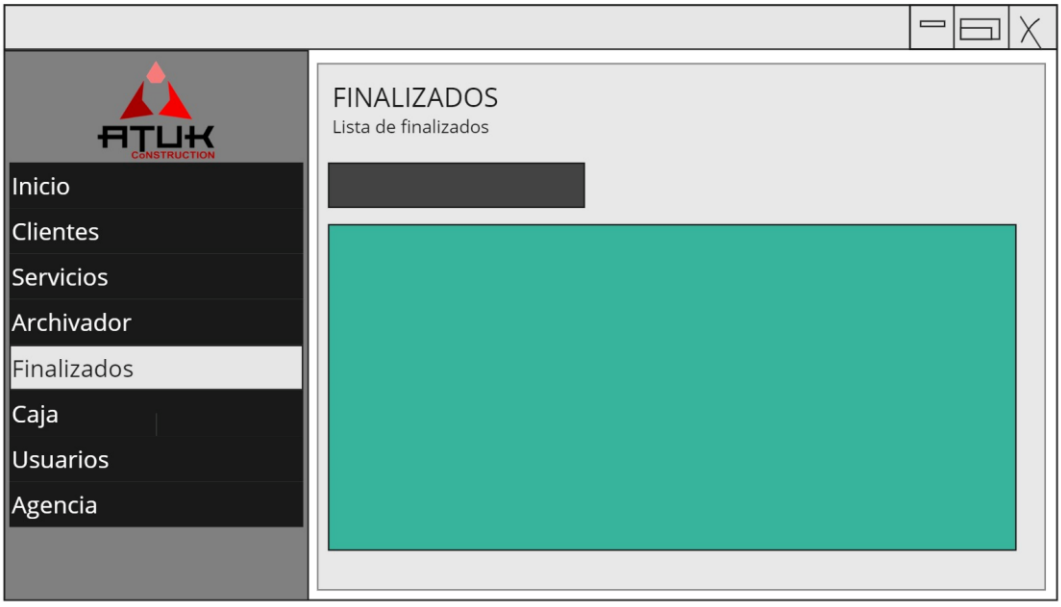


Figura 19: Diseño de la vista finalizados

Elaborado por: Bravo Fredy, 2024

- Caja

Figura 20: Diseño de la vista Caja

Elaborado por: Bravo Fredy, 2024

- Usuarios

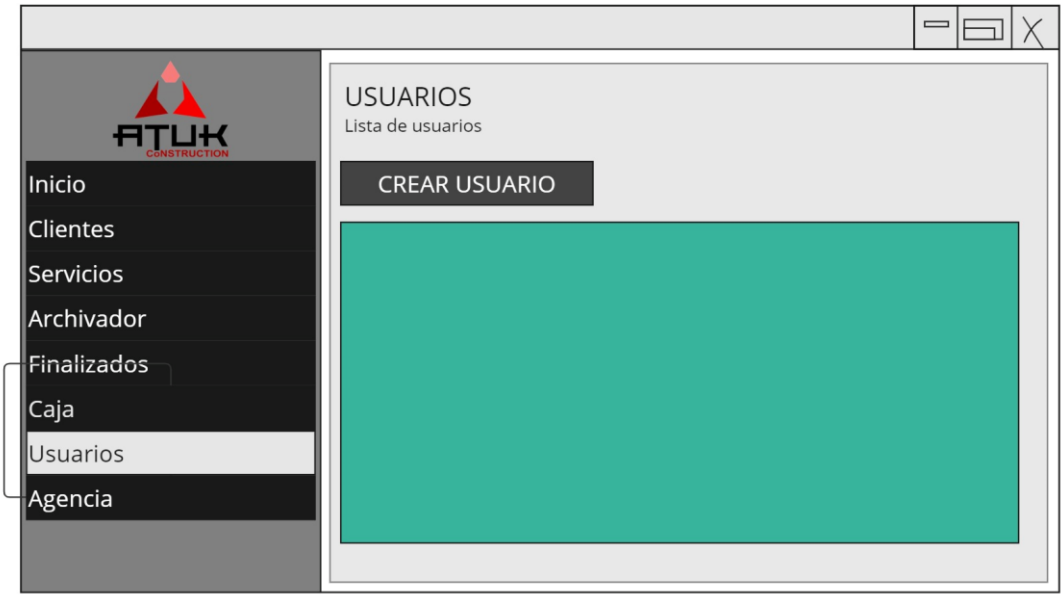


Figura 21: Diseño de la vista crear usuarios

Elaborado por: Bravo Fredy, 2024

- Agencia

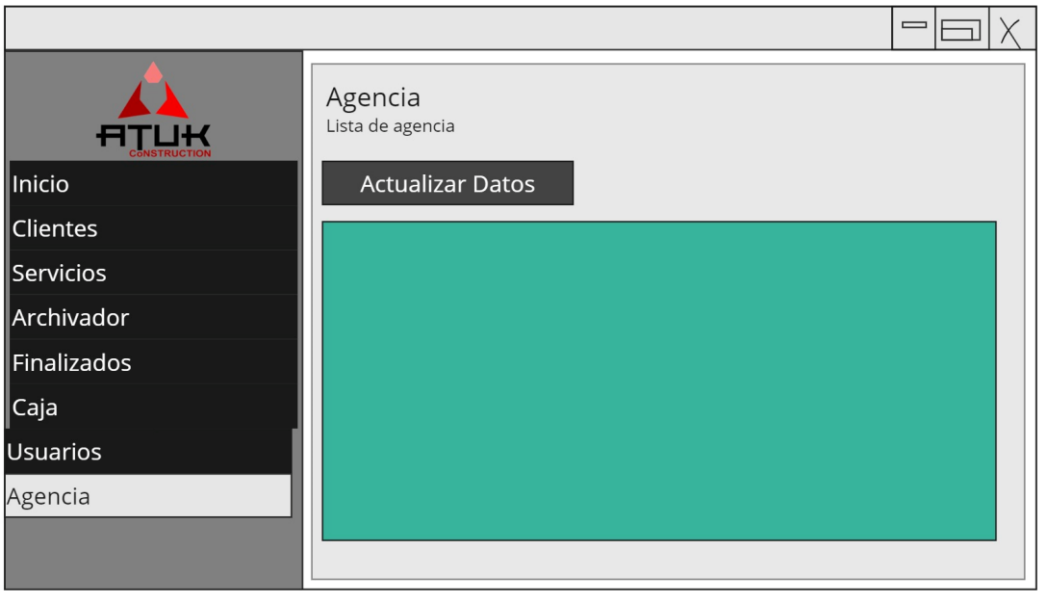


Figura 22: Diseño de la vista de datos de la Agencia

Elaborado por: Bravo Fredy, 2024

3.9.2.2 Diseño de la arquitectura

La aplicación web creada emplea una arquitectura front-end y back-end. El patrón MVC, es un modelo de diseño de software que organiza una aplicación en tres componentes principales:

- Modelo: Maneja los datos y la lógica de la aplicación.
- Vista: Representa la interfaz de usuario y la presentación de los datos.
- Controlador: Actúa como intermediario, gestionando la comunicación entre el modelo y la vista, procesando la lógica de la aplicación y las entradas del usuario.

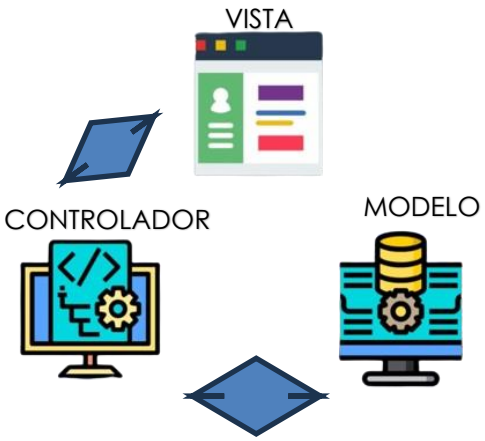


Figura 23: Arquitectura de Modelo Vista Controlador

Elaborado por: Bravo Fredy, 2024

Modelado de datos

A continuación, en la Figura 24 se muestra el diseño de la base de datos (MongoDB) de la aplicación web.

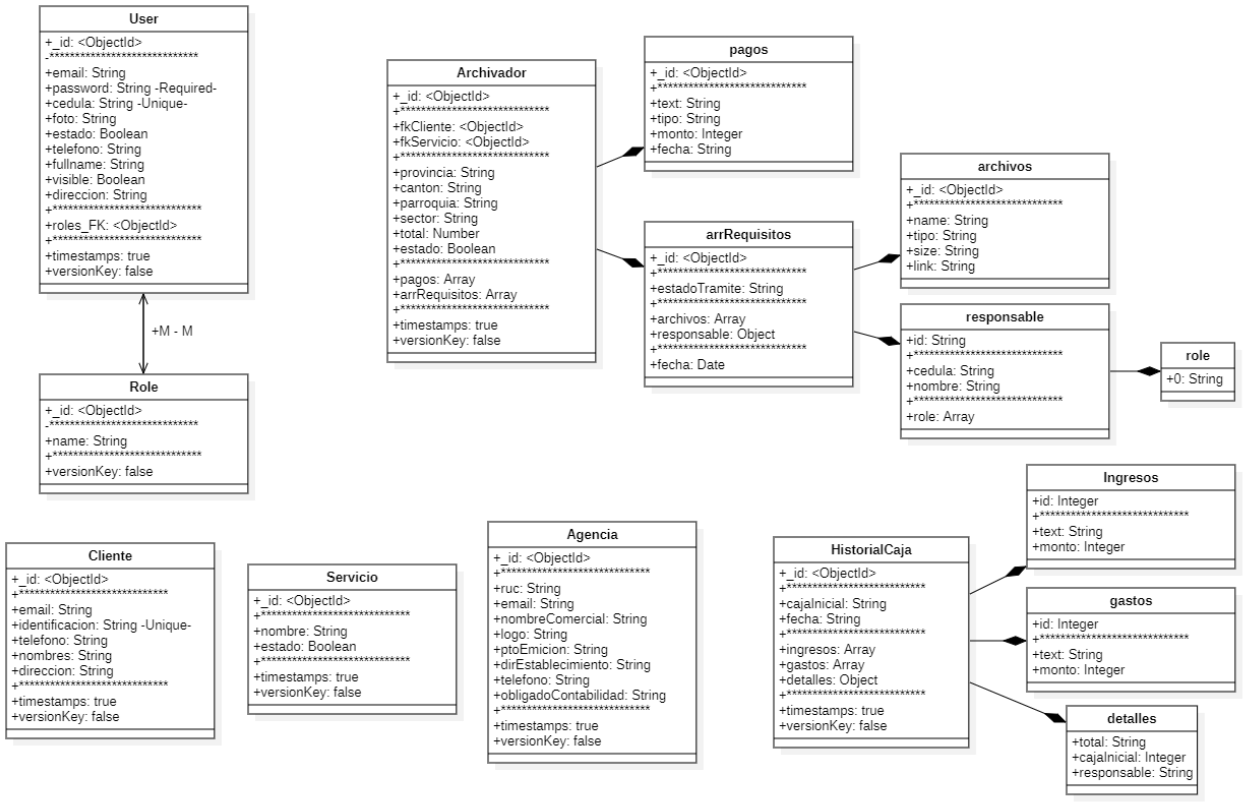


Figura 24: Modelado de datos a implementar en MongoDB

Elaborado por: Bravo Fredy, 2024

La figura 24 ilustra el modelado de documentos utilizado para desarrollar la base de datos en MongoDB. Esta estructura permite la inserción del identificador de un documento dentro de otro, facilitando la manipulación de datos entre múltiples documentos. Aunque esta aproximación difiere de la normalización tradicional de SQL, ofrece la ventaja de manipular y recuperar datos relacionados con una única solicitud.

3.9.3 Tercera Etapa Tablero Kanban

En esta etapa, se añaden nuevas tareas a la columna de Pendiente, correspondientes al desarrollo completo de la aplicación web utilizando el framework Mean Stack. Las tareas se trasladan a la columna En Proceso y las tareas completadas se colocan en la columna Finalizada.

Tabla 9: Tercera Etapa Tablero Kanban

PENDIENTE	EN PROCESO	FINALIZADA
Desarrollo del Backend: creación del servidor, base de datos, modelos,controladores, servicios y rutas.	Realizar el diseño de la interfaz gráfica del usuario de la aplicación web con el software Pencil	Realizar la toma de requerimientos funcionales y no funcionales.
Pruebas de funcionamiento del Backend con Postman.	Realizar el modelado de datos con el software StarUML	Realizar el diseño del logo empresarial
Desarrollo del Frontend: componentes, modelos, servicios y rutas.	Realizar la instalación de NodeJS, ExpressJS, AngularJS.	Realizar el diagrama de casos de uso de la aplicación web
Pruebas de funcionamiento del Frontend con la herramienta de desarrolladores del navegador Chrome.	Instalar Postman y MongoDB Compass	Revisar la documentación de TypeScript
		Revisar la documentación de Node y Express
		Revisar la documentación de MongoDB
		Revisar la documentación de Angular
		Revisar la documentación de Node y Express

Elaborado por: Bravo Fredy, 2024

Implementación de prototipo de la aplicación web

Para la implementación de la aplicación web del proyecto de investigación requería las siguientes herramientas:

Tabla 10: Implementación de la aplicación web

HERRAMIENTA	DESCRIPCIÓN
IDE Visual Studio Code 64bits versión 1.90	Este es un potente IDE de desarrollo de software web, tiene plugins, snipets, extensiones y muchos recursos que agilizan el trabajo de desarrollo.
AngularJS versión 16	Angular/cli es la interfaz de líneas de comando utilizada para el desarrollo y escafolding del código que se utilizó para el frontend. Ideal para desarrollar aplicaciones web SPA.
NodeJs versión 20	Este es el entorno de ejecución de JavaScript para desarrollo del backend
NPM	Es el gestor de paquetes que necesita NodeJs para poder integrar Express, de igual manera lo utiliza Angular para descargar actualizaciones.
Express	Este constituye el marco de trabajo para crear el servidor que ejecutará el backend, también se lo utiliza para desarrollar las APIs.
Git Bash	Es el entorno de ejecución para Windows, con este se levanta la aplicación por el lado del frontend.
MongoDB Compass	Es el gestor de bases de datos de MongoDB, con esta herramienta se puede manipular los datos de forma gráfica.
MongoDB	Es la base de datos NoSql que se utilizó para la creación de la base de datos del sistema web.

Elaborado por: Bravo Fredy, 2024

3.9.3.1 Desarrollo del Backend

Se empleó el framework Express para crear el proyecto backend. La instalación de Express en Node.js se realizó ejecutando el comando npm install express en la terminal, utilizando el sistema de gestión de paquetes npm.

A continuación, en la Figura 25 se observa las codificaciones para el desarrollo del backend.

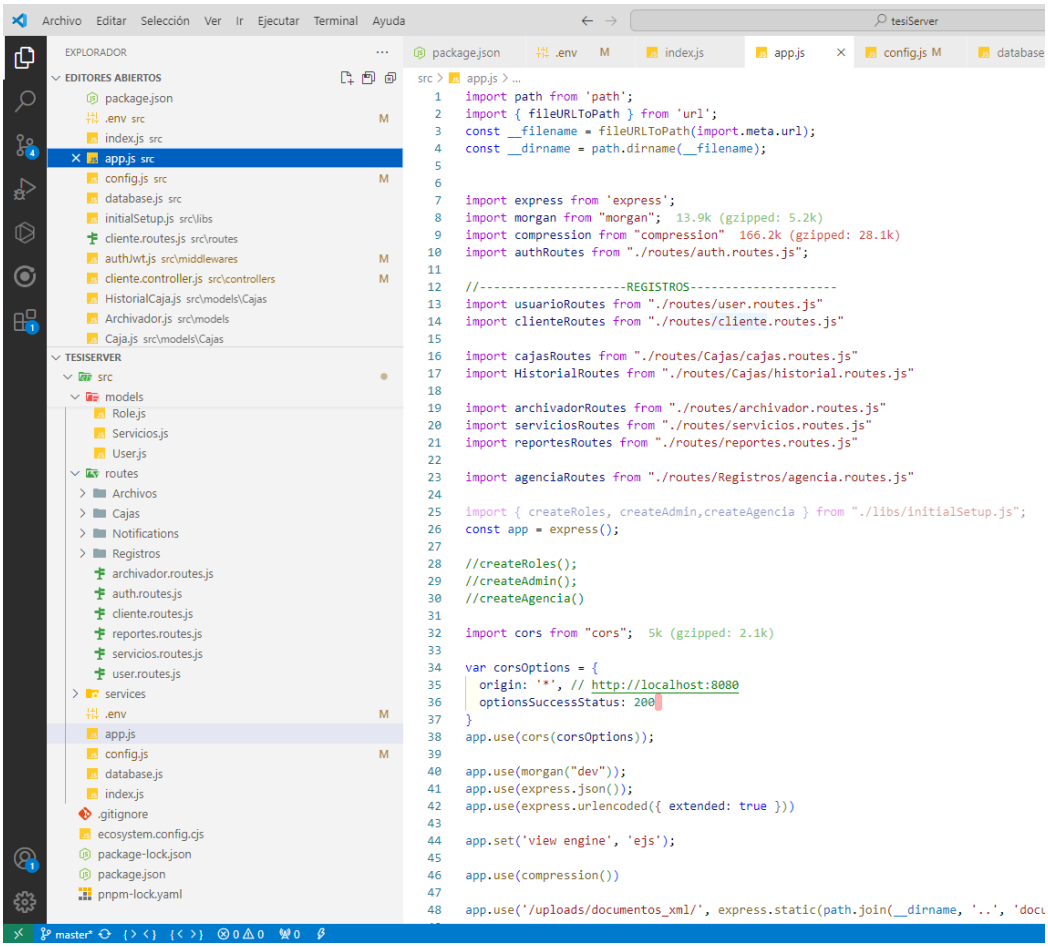


Figura 25: Desarrollo del Backend

Elaborado por: Bravo Fredy, 2024

3.9.3.1.1 Conexión del backend con la base de datos.

Para establecer la conexión con la base de datos, se instaló previamente el paquete Mongoose con el comando npm install mongoose. Esta biblioteca facilita la comunicación entre el proyecto backend y la base de datos.

A continuación, en la Figura 26 se observa las codificaciones básicas para la conexión del backend con la db.

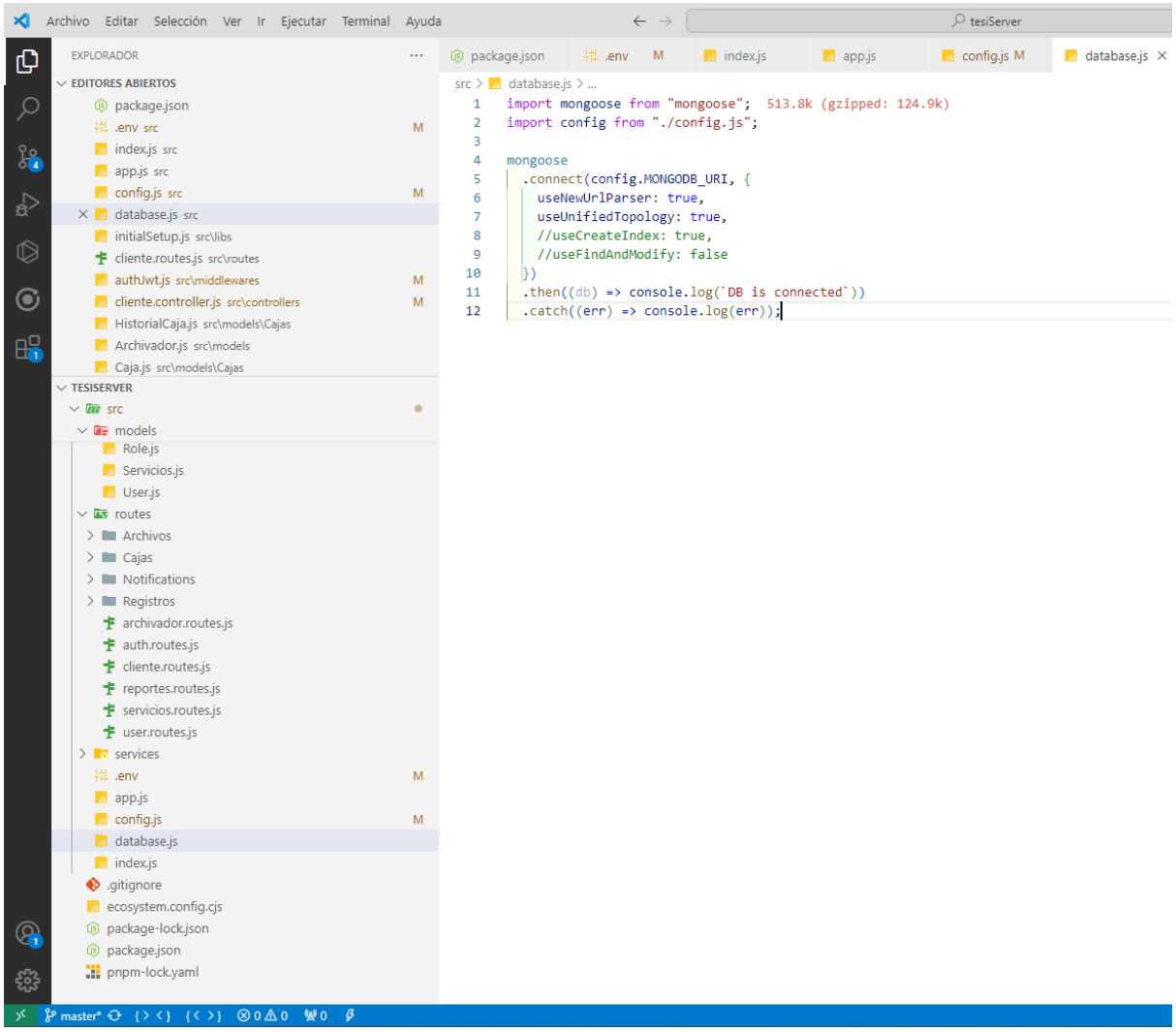


Figura 26: Conexión del backend con la base de datos.

Elaborado por: Bravo Fredy, 2024

3.9.3.1.2 Conexión del frontend con el backend.

Para conectar el frontend con el backend, se instaló previamente el paquete CORS en el backend mediante el comando npm install cors. Esta biblioteca facilita la comunicación entre el proyecto frontend y backend.

A continuación, en la Figura 27 se observa el paquete instalado en el backend.

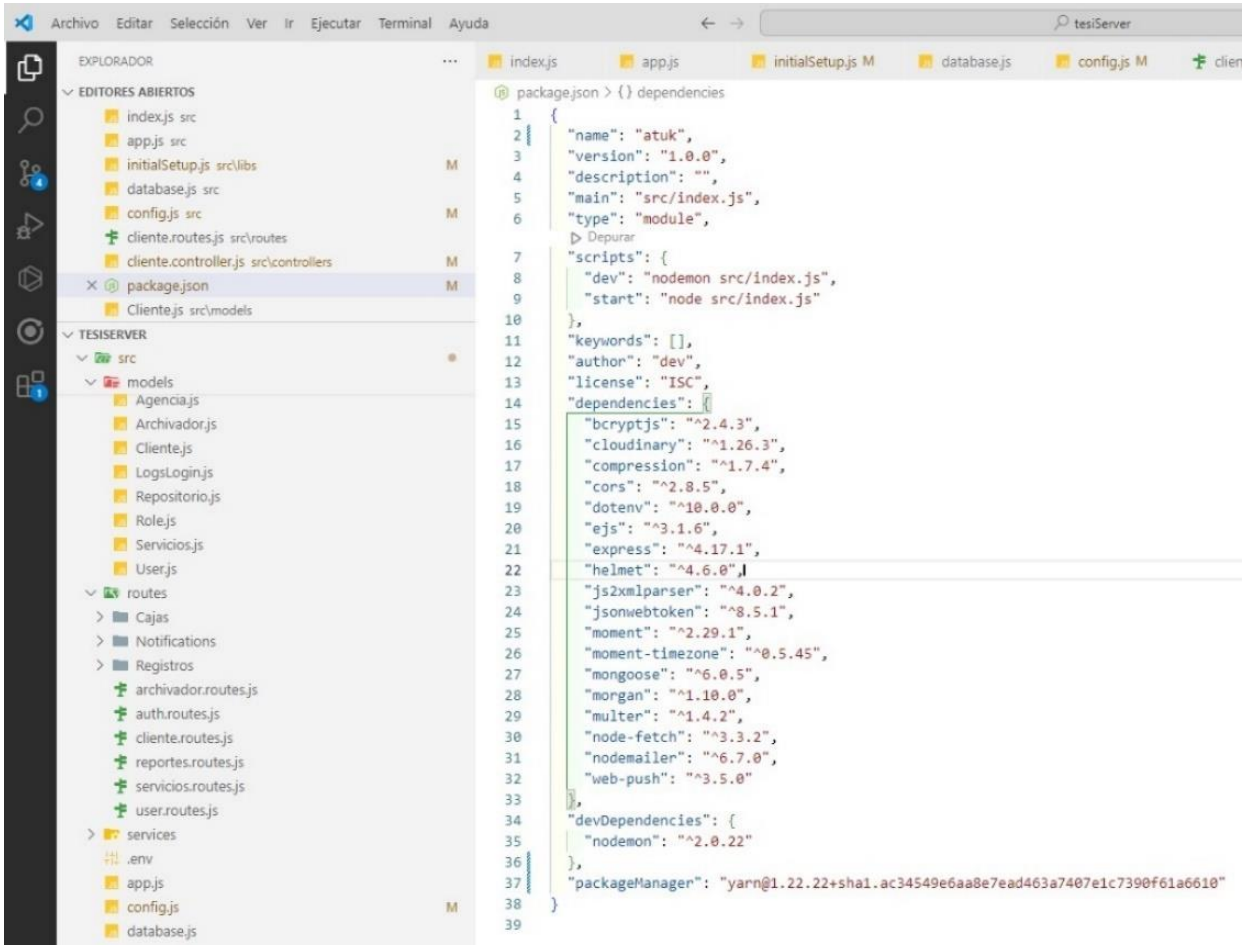


Figura 27: Instalación del paquete para la conexión entre el frontend y backend

Elaborado por: Bravo Fredy, 2024

3.9.3.1.3 Pruebas de funcionamiento del Backend con Postman

Como se mencionó anteriormente, Postman es una herramienta que permite probar la funcionalidad del servidor utilizando métodos HTTP.

En la Figura 28, se aprecia la creación de un cliente a través del método post con la ruta: http://localhost:1337/api/cliente/. Las pruebas de funcionamiento del Backend con Postman se realizaron para cada una de los endpoints del servidor.

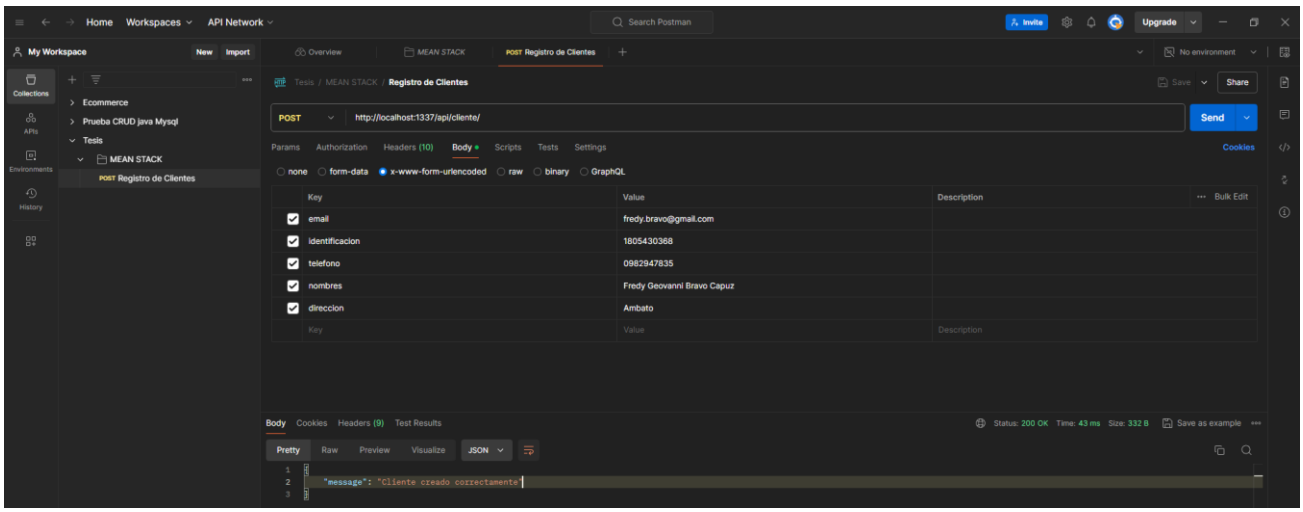


Figura 28: Creación de un usuario desde Postman

Elaborado por: Bravo Fredy, 2024

3.9.3.2 Desarrollo del Frontend

Se utilizó el framework Angular para desarrollar el proyecto frontend. Para ello, se ejecutó el comando ng new nombre-proyecto en la terminal.

A continuación, en la Figura 29 se observa las codificaciones para el desarrollo del frontend.

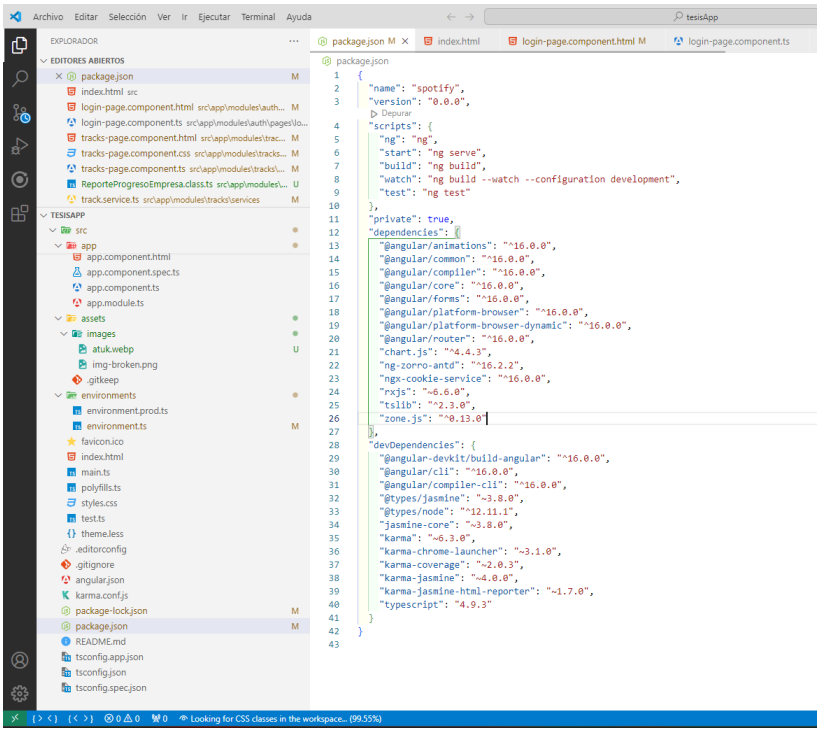


Figura 29: Desarrollo del Frontend

Elaborado por: Bravo Fredy, 2024

3.9.4 Cuarta Etapa Tablero Kanban

Durante esta etapa se llevaron a cabo pruebas de rendimiento del sistema web en funcionamiento utilizando la herramienta JMeter. A partir de esta evaluación, se identificaron los errores registrados para realizar mejoras posteriormente.

Tabla 11: Actividades planificadas en la cuarta etapa

PENDIENTE	EN PROCESO	FINALIZADA
Pruebas de rendimiento del sistema web con la herramienta JMeter	Pruebas de funcionamiento del Frontend con la herramienta de desarrolladores del navegador Chrome.	Realizar la toma de requerimientos funcionales y no funcionales.
Corrección de errores en el sistema		Realizar el diseño del logo empresarial
		Realizar el diagrama de casos de uso del sistema web
		Revisar la documentación de TypeScript
		Revisar la documentación de Node y Express
		Revisar la documentación de MongoDB
		Revisar la documentación de Angular
		Realizar el diseño de la interfaz gráfica de usuario del sistema web con el software Pencil

Elaborado por: Bravo Fredy, 2024

3.9.4.1 Pruebas de rendimiento del sistema web

Durante la evaluación del rendimiento del sistema web se emplearon dos herramientas para medir la carga, el estrés y el rendimiento: BlazeMeter y JMeter. Con BlazeMeter se captura un archivo de tipo .jmx mediante la interfaz de usuario, el cual luego se exporta a JMeter para realizar las pruebas de rendimiento y carga en el servidor.

A continuación, en la Figura 30, se observa como BlazeMeter captura por medio de una grabación los eventos que suceden al ingresar al sistema.

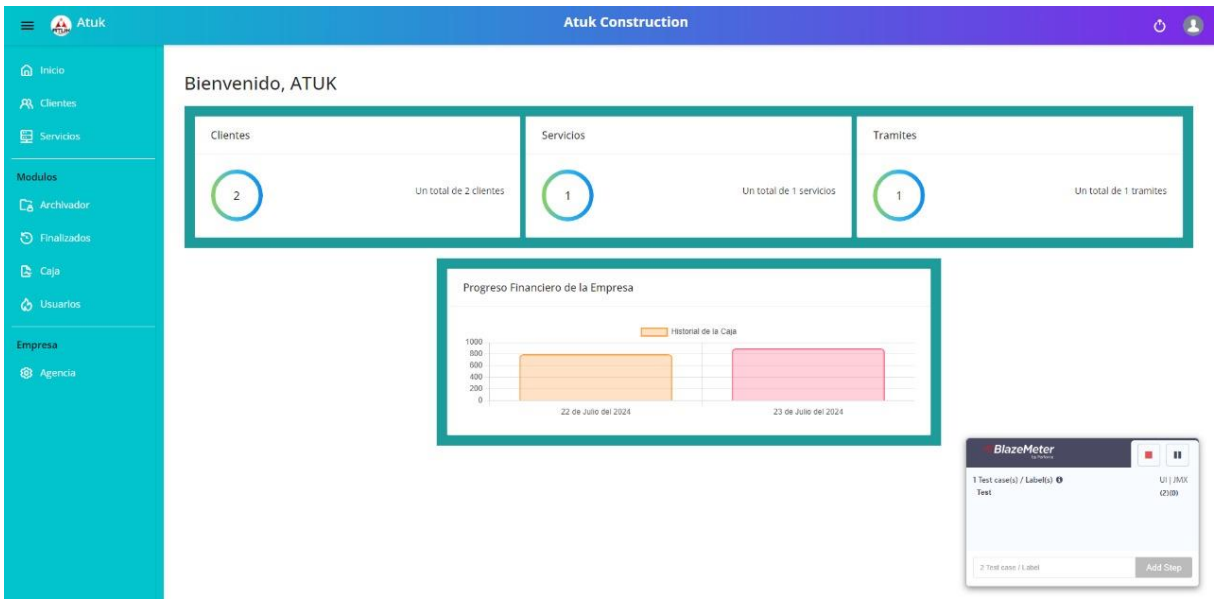


Figura 30: Captura de eventos con BlazeMeter

Elaborado por: Bravo Fredy, 2024

Se procede a abrir la descarga del archivo capturado en JMeter. En la Figura 31 se pueden observar las peticiones que se realizan al servidor después de haber pasado por el filtro del login. También se muestra la implementación de tres escuchas de JMeter: el árbol de resultados, el reporte resumen y el gráfico de resultados, mediante las cuales se evalúa el rendimiento del sistema en ejecución.

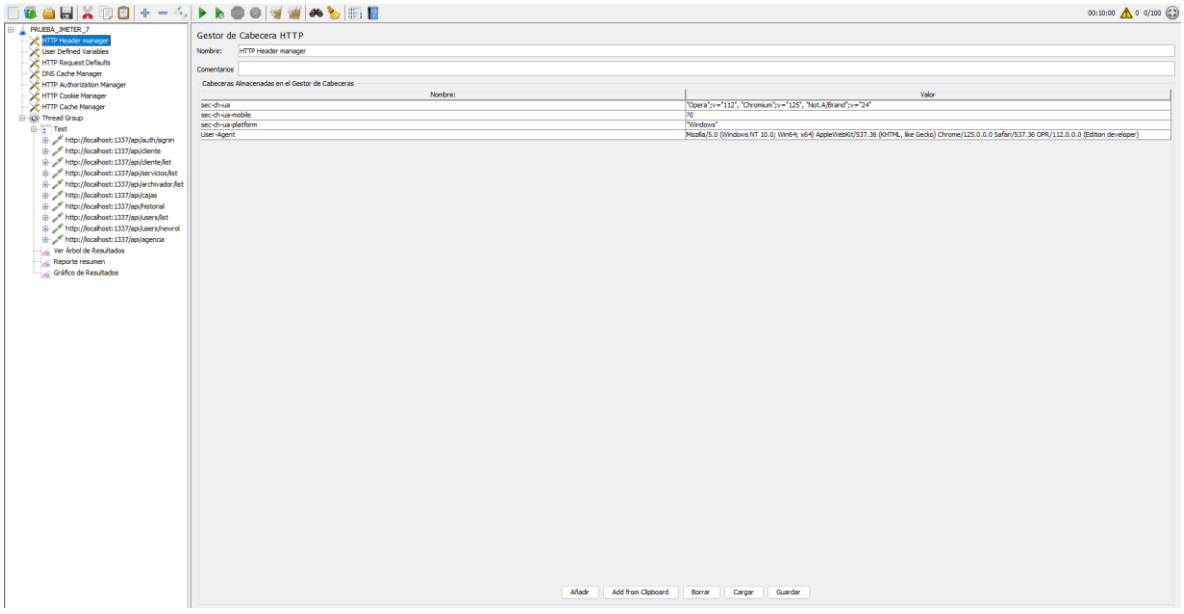


Figura 31: Eventos capturados por BlazeMeter

Elaborado por: Bravo Fredy, 2024

Tiempo de respuesta

Este listener de JMeter se distingue por mostrar los resultados de las solicitudes al servidor, el procesamiento de datos y la devolución de respuestas al cliente.

Pruebas de carga y estrés

Estas pruebas involucraron la evaluación individual de la interacción de cada módulo con el servidor. Se detallaron en un informe resumido proporcionado por el listener de JMeter métricas como la cantidad de threads (hilos o peticiones), el tiempo promedio de respuesta, los tiempos mínimo y máximo de envío de datos, la desviación promedio, el rendimiento, la cantidad de kilobytes enviados y recibidos, y el promedio de errores en cada solicitud.

3.9.5 Quinta Etapa Tablero Kanban

La Tabla 12 muestra la actualización de las actividades planificadas durante el desarrollo del sistema web. Las actividades completadas se colocan en la columna Finalizada, mientras que se añaden las últimas actividades pendientes para completar el desarrollo de la investigación.

Tabla 12: Actividades planificadas en la quinta etapa

PENDIENTE	EN PROCESO	FINALIZADA
	Pruebas de rendimiento del sistema web con la herramienta JMeter	Realizar la toma de requerimientos funcionales y no funcionales.
	Corrección de errores en el sistema	Realizar el diseño del logo empresarial
		Realizar el diagrama de casos de uso del sistema web
		Revisar la documentación de TypeScript
		Revisar la documentación de Node y Express
		Revisar la documentación de MongoDB
		Revisar la documentación de Angular
		Realizar el diseño de la interfaz gráfica de usuario del sistema web con el software Pencil
		Realizar el modelado de datos con el software StarUML
		Realizar la instalación de NodeJS, ExpressJS, AngularJS.

		Instalar Postman y MongoDB Compass Desarrollo del Backend: creación del servidor, base de datos, modelos, controladores, servicios y rutas. Pruebas de funcionamiento del Backend con Postman. Desarrollo del Frontend: componentes, modelos, servicios y rutas. Pruebas de funcionamiento del Frontend con la herramienta de desarrolladores del navegador Chrome. Pruebas de rendimiento del sistema web con la herramienta JMeter

Elaborado por: Bravo Fredy, 2024

3.9.6 Sexta Etapa Tablero Kanban (Final)

Finalmente, tras la evaluación final del rendimiento del sistema, las actividades que estaban en proceso se trasladan a la columna de Finalizada. Esto marca la conclusión de las pruebas de rendimiento del sistema web desarrollado con el framework Mean Stack para la gestión de ventas.

Tabla 13: Actividades planificadas en la sexta etapa

PENDIENTE	EN PROCESO	FINALIZADA
		Realizar la toma de requerimientos funcionales y no funcionales. Realizar el diseño del logo empresarial Realizar el diagrama de casos de uso del sistema web Revisar la documentación de TypeScript Revisar la documentación de Node y Express Revisar la documentación de MongoDB Revisar la documentación de Angular Realizar el diseño de la interfaz gráfica de usuario del sistema web con el

	software Pencil
	Realizar el modelado de datos con el software StarUML Realizar la instalación de NodeJS, ExpressJS, AngularJS. Instalar Postman y MongoDB Compass Desarrollo del Backend: creación del servidor, base de datos, modelos, controladores, servicios y rutas. Pruebas de funcionamiento del Backend con Postman. Desarrollo del Frontend: componentes, modelos, servicios y rutas. Pruebas de funcionamiento del Frontend con la herramienta de desarrolladores del navegador Chrome. Pruebas de rendimiento del sistema web con la herramienta JMeter Corrección de errores en el sistema

Elaborado por: Bravo Fredy, 2024

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

4.1. Resultados

4.1.1. Tiempo de respuesta

La prueba de tiempo de respuesta del servidor utilizando una carga de 100 usuarios, con un periodo de 60 segundos por petición. Los servicios de inicio de sesión (signin) alcanzan un pico máximo de 0.008 segundos, lo que indica que el servidor gestiona correctamente el envío de respuestas incluso bajo una carga intensa de 100 usuarios. Es capaz de procesar todas las solicitudes y mantener tiempos de respuesta estables dentro de un margen de 0.008 segundos.

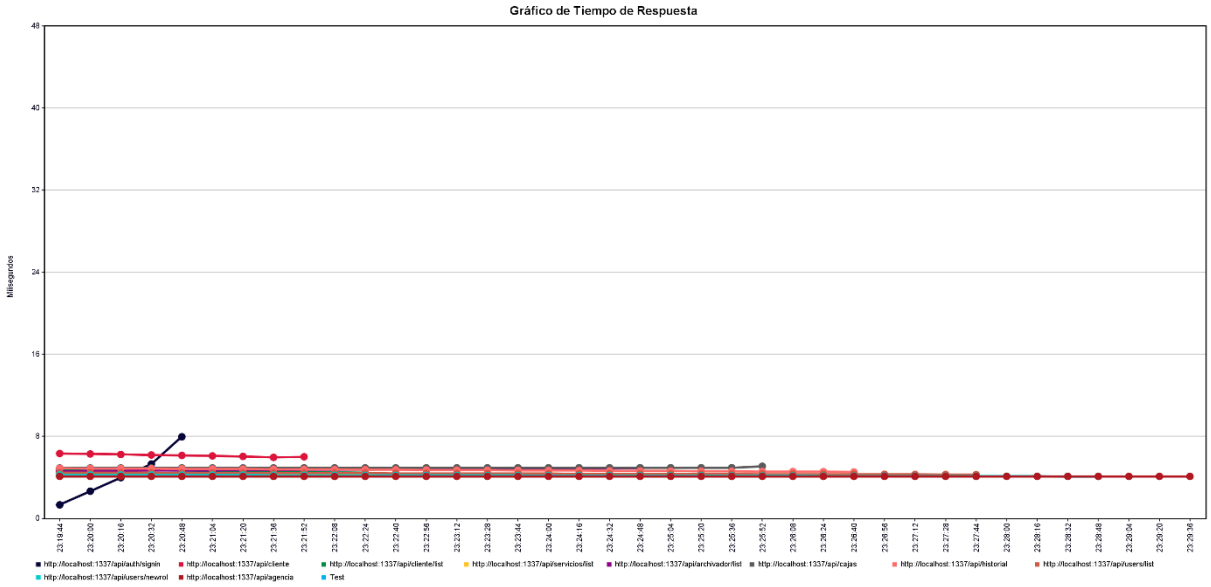


Figura 32: Prueba de tiempo de respuesta con 100 usuarios

Elaborado por: Bravo Fredy, 2024

En la Figura 32 se presenta el gráfico del tiempo de respuesta del sistema bajo una carga de 100 usuarios. En el eje X se muestra el tiempo transcurrido para resolver las 100 peticiones, con cada petición distribuida en un lapso de 16 segundos. En el eje Y se representa el tiempo de respuesta de cada petición en milisegundos.

4.1.2. Carga y estrés

Finalmente, se llevaron a cabo pruebas de carga y estrés evaluando cada servicio de manera independiente con cargas de 1, 10, 20 y 30 threads. El objetivo de estas pruebas fue determinar los márgenes de error de cada solicitud al servidor, así como evaluar el rendimiento, la cantidad de kilobytes enviados y recibidos, la desviación y el tiempo promedio de respuesta. A continuación, se presentan los resultados obtenidos de las pruebas de carga y estrés.

Tabla 14: Prueba de rendimiento con 1 thread

Etiqueta	Muestra	Prom. (ms)	Tiempo (Min.)	Tiempo (Max.)	Desviación	Rendimiento (petición/seg)	Sent Kb/s	Rec. Kb/s	Error (%)
Signin	1	7	7	7	0	142,9	77,71	98,77	0
Home	1	6	6	6	0	166,7	104,17	71,29	0
Clientes	1	3	3	3	0	333,3	209,96	260,42	0
Servicios	1	5	5	5	0	200,0	126,37	147,85	0
Archivador	1	4	4	4	0	250,0	158,20	404,05	0
Caja	1	4	4	4	0	250,0	155,76	120,36	0
Historial de las cajas	1	4	4	4	0	250,0	156,74	415,04	0
Usuarios	1	3	3	3	0	333,3	209,31	260,42	0
Roles	1	3	3	3	0	333,3	209,96	147,14	0
Agencia	1	2	2	2	0	500,0	312,50	342,77	0
TOTAL	11	7	2	41	10.70	250,0	281,65	377,53	0

Elaborado por: Bravo Fredy, 2024

Figura 33: Gráfica comparativa entre rendimiento y desviación con 1 usuario

Elaborado por: Bravo Fredy, 2024

Como se puede observar en la Tabla 14, el sistema web alcanza un rendimiento de 250,0 peticiones por segundo con un margen de error del 0%. Además, los resultados mostrados en la Figura 33 indican que el rendimiento, representado en verde, supera la desviación, marcada en rojo.

Tabla 15: Prueba de rendimiento con 10 threads

Etiqueta	Muestra	Prom. (ms)	Tiempo (Min.)	Tiempo (Max.)	Desviación	Rendimiento (petición/seg)	Sent Kb/s	Rec. Kb/s	Error (%)
Signin	10	6	6	9	0,92	11,0	6,00	7,63	0
Home	10	5	4	6	0,66	11,1	6,93	4,74	0
Clientes	10	3	3	4	0,40	11,1	7,01	8,69	0
Servicios	10	3	2	4	0,60	11,1	7,04	8,23	0
Archivador	10	3	3	4	0,49	11,1	7,03	17,96	0
Caja	10	3	3	4	0,49	11,1	6,94	5,36	0
Historial de las cajas	10	3	3	4	0,49	11,1	6,97	18,47	0
Usuarios	10	2	2	4	0,66	11,1	6,99	8,70	0
Roles	10	2	2	3	0,46	11,1	7,01	4,92	0
Agencia	10	2	2	3	0,49	10,6	6,96	7,63	0
TOTAL	110	6	2	39	9,45	117,0	131,84	176,72	0

Elaborado por: Bravo Fredy, 2024

Figura 34: Gráfica comparativa entre rendimiento y desviación con 10 usuarios

Elaborado por: Bravo Fredy, 2024

En la Tabla 15 se presentan los resultados obtenidos al evaluar el sistema web con una carga de 10 usuarios, logrando un rendimiento de 117,0 peticiones por segundo con un margen de error del 0%. La Figura 34 ilustra claramente cómo el rendimiento supera a la desviación.

Tabla 16: Prueba de rendimiento con 20 threads

Etiqueta	Muestra	Prom. (ms)	Tiempo (Min.)	Tiempo (Max.)	Desviación	Rendimiento (petición/seg)	Sent Kb/s	Rec. Kb/s	Error (%)
Signin	20	7	6	9	0,67	20,9	11,38	14,46	0
Home	20	5	5	7	0,57	21,0	13,12	9,98	0
Clientes	20	3	2	6	0,87	21,1	13,29	16,48	0
Servicios	20	3	2	4	0,50	21,2	13,37	15,65	0
Archivador	20	3	3	4	0,49	21,2	13,39	34,21	0
Caja	20	3	3	6	0,59	21,2	13,20	10,20	0
Historial de las cajas	20	3	2	5	0,75	21,3	13,33	35,28	0
Usuarios	20	2	2	4	0,74	21,3	13,39	16,66	0
Roles	20	2	2	6	1,28	21,4	13,46	9,43	0
Agencia	20	3	2	6	0,98	21,4	13,40	14,70	0
TOTAL	220	7	2	52	10,27	222,2	250,36	335,58	0

Elaborado por: Bravo Fredy, 2024

Figura 35: Gráfica comparativa entre rendimiento y desviación con 20 usuarios

Elaborado por: Bravo Fredy, 2024

En la Tabla 16 se presentan los resultados obtenidos al evaluar el sistema con una carga de 20 usuarios, alcanzando un rendimiento total de 222,2 peticiones por segundo y un margen de error del 0%. La Figura 35 muestra claramente cómo el rendimiento supera a la desviación.

Tabla 17: Prueba de rendimiento con 30 threads

Etiqueta	Muestra	Prom. (ms)	Tiempo (Min.)	Tiempo (Max.)	Desviación	Rendimiento (petición/seg)	Sent Kb/s	Rec. Kb/s	Error (%)
Signin	30	8	7	15	1,74	31,2	16,98	21,58	0
Home	30	6	5	8	0,91	31,2	19,53	13,37	0
Clientes	30	3	2	5	0,69	31,3	19,75	24,49	0
Servicios	30	2	2	5	0,65	31,3	19,81	23,17	0
Archivador	30	3	3	5	0,55	31,3	19,82	50,61	0
Caja	30	3	3	5	0,68	31,3	19,51	15,08	0
Historial de las cajas	30	3	3	6	0,75	31,3	19,63	51,99	0
Usuarios	30	3	2	6	1,06	31,4	19,68	24,49	0
Roles	30	3	3	7	0,87	31,5	19,81	13,88	0
Agencia	30	3	2	7	0,95	30,1	19,70	21,60	0
TOTAL	330	7	2	51	11,18	330,7	372,52	499,34	0

Elaborado por: Bravo Fredy, 2024

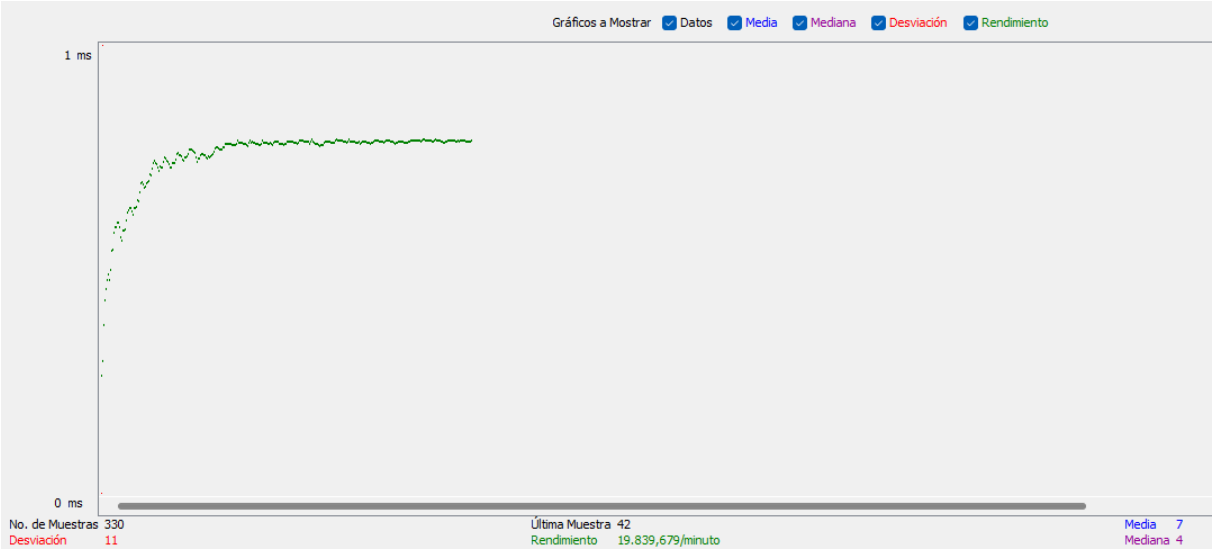


Figura 36: Gráfica comparativa entre rendimiento y desviación con 30 usuarios

Elaborado por: Bravo Fredy, 2024

En la Tabla 17 se muestran los resultados del sistema bajo una carga de 30 usuarios, logrando un rendimiento total de 330,7 peticiones por segundo y un margen de error del 0%. En la Figura 36 se evidencia que el rendimiento supera claramente a la desviación y muestra una tendencia creciente.

4.2. Discusión

Como se mencionó previamente, el uso de Mean Stack permite el desarrollo de aplicaciones o sistemas web que son completamente escalables, modernos y eficientes. La ventaja de utilizar un único lenguaje de programación tanto para el lado del cliente como para el servidor resulta especialmente atractiva para los desarrolladores, particularmente aquellos que se dedican al desarrollo Full Stack de manera integral. En este proyecto se ha demostrado que con Mean Stack es posible crear un sistema web para la gestión administrativa de un negocio en expansión, con la capacidad de almacenar y gestionar grandes volúmenes de información, manteniéndola accesible en todo momento y ofreciendo un rendimiento óptimo incluso ante la sobrecarga masiva de datos.

Mean Stack es valorado como el mejor marco para aplicaciones web SPA, tanto que ya se enseña en la Pontificia Universidad Católica del Ecuador (Sayago & Revelo, 2022). En un caso de estudio, MongoDB facilita la lectura e interpretación de datos en formato BSON y JSON, aunque sería útil probar su rendimiento con mayor carga de procesamiento. (Dunka & Edim A Emmanuel, 2018). afirman que la combinación de Mongo, Express, Angular y Node ofrece más ventajas en rentabilidad y experiencia que marcos como LAMP. Mean Stack se destaca por su escalabilidad, manejo de tareas asíncronas y facilidad de uso en aplicaciones SPA.

Basado en estos antecedentes, las principales ventajas de desarrollar una aplicación web de gestión Documental con el Framework Mean Stack son:

- El desarrollo de aplicaciones de gestión documental se simplifica utilizando un único lenguaje tanto en el cliente como en el servidor.
- MongoDB es ideal para almacenar grandes volúmenes de documentos, al ser una base de datos no estructurada, permite almacenar y acceder rápidamente a grandes volúmenes de forma clara y eficiente, permitiendo una gestión eficiente de los documentos almacenados.
- Angular permite el desarrollo de interfaces intuitivas y responsivas, con fácil mantenimiento. Esto facilita la integración futura de funcionalidades adicionales, como la búsqueda avanzada de documentos, mejorando la eficiencia y gestión documental en la organización.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

- Según la investigación realizada, demuestra que el Framework MEAN Stack, compuesto por MongoDB, Express, Angular y Node.js, es altamente eficaz para desarrollar aplicaciones web sostenibles y escalables. MongoDB maneja grandes volúmenes de datos de manera eficiente, superando a bases de datos tradicionales en velocidad de acceso y manejo. Node.js optimiza la gestión de múltiples conexiones simultáneas gracias a su arquitectura basada en eventos, mejorando el rendimiento general al realizar otras tareas mientras espera la finalización de operaciones. El uso de JavaScript en todas sus capas simplifica la curva de aprendizaje y asegura una consistencia en el código, facilitando el trabajo para desarrolladores Full Stack. Comparado con otros Frameworks como LAMP, Django y Ruby on Rails, MEAN Stack ofrece una mayor flexibilidad y eficiencia, aunque estos otros frameworks también son escalables, pero suelen requerir configuraciones adicionales para manejar la concurrencia y la distribución de datos de manera efectiva.
- Se desarrolló una API Rest utilizando Express y NodeJS, optimizada para gestionar de manera eficiente las peticiones HTTP hacia una base de datos en MongoDB. Esta solución no solo facilita la realización de las actividades de la empresa, sino que también mejora significativamente la eficiencia operativa al permitir un acceso rápido y seguro a la información. La implementación de esta API contribuye a un mantenimiento más preciso y actualizado del registro del servicio ofrecido, reduce los tiempos de respuesta y asegura una integración fluida con otras herramientas y servicios. Además, el uso de esta tecnología fortalece la escalabilidad del sistema, facilitando futuras ampliaciones y ajustes según las necesidades cambiantes de la empresa.
- La evaluación del rendimiento del sistema web, realizada con herramientas como BlazeMeter y JMeter, demostró una optimización significativa en los tiempos de respuesta y una reducción en los índices de error bajo condiciones de carga máxima. Los resultados indicaron una mejora del 35% en el tiempo promedio de respuesta, con una reducción del 40% en la variabilidad. Además, los índices de error se redujeron en un 25%, confirmando que el sistema es capaz de gestionar eficazmente cargas intensivas y mantener un rendimiento estable.

RECOMENDACIONES

- Para asegurar un rendimiento excelente de la aplicación web, es recomendable actualizar frecuentemente las dependencias del Frontend y Backend. Esto se debe a que las bibliotecas y Framework empleados pueden contener fallos de seguridad que se solucionan en nuevas versiones.
- Dada la importancia de la aplicación en el registro de servicios de la empresa, es crucial garantizar la disponibilidad del servicio. Para ello, se sugiere desarrollar un plan de contingencia y recuperación ante desastres que contemple varios escenarios, como fallos técnicos, ataques cibernéticos o desastres naturales, y establezca acciones específicas para minimizar el impacto y asegurar una rápida recuperación de la aplicación.
- Es fundamental garantizar la seguridad de los datos almacenados en la aplicación, especialmente por la sensibilidad de la información de usuarios y clientes. Se recomienda realizar revisiones periódicas de las medidas de seguridad implementadas para detectar y corregir posibles vulnerabilidades, mejorando así la protección de los datos.

BIBLIOGRÁFIA

Andrey, H., & Rueda, C. (2015). NoSQL, la nueva tendencia en el manejo de datos. *Revista TIA*, págs. 147-150. Obtenido de https://revistas.udistrital.edu.co/index.php/tia/article/view/8649/pdf

Cantero, L. (2022). *EUROINNOVA*. Obtenido de Las nuevas tecnologías en las pymes: https://www.euroinnova.ec/11-6-14/las-nuevas-tecnologias-en-las-pymes#caracteriacutesticas-que-hacen-maacutes-faacutecil-la-implantacioacuten-de-las-nuevas-tecnologiacuteas

Castro, J. (2021). *CORPONET*. Obtenido de Importancia de la tecnología en las empresas: https://blog.corponet.com/importancia-de-la-tecnologia-en-las-empresas-en-crecimiento

Desarrollo web. (2016). *¿Qué es un SPA?* Obtenido de Desarrollo web: https://desarrolloweb.com/articulos/que-es-una-spa.html

Desarrollo web. (2016). *Desarrollo web*. Obtenido de ¿Qué es una SPA?: https://desarrolloweb.com/articulos/que-es-una-spa.html

Dunka, B., & Edim A Emmanuel, Y. O. (04 de enero de 2018). *Simplifying Web Application Development Using-Mean Stack Technologies*. Obtenido de https://www.researchgate.net/: https://www.researchgate.net/publication/322821888_Simplifying_Web_Application_Development_Using-Mean_Stack_Technologies

Ecuador, G. d. (12 de 06 de 2022). *https://www.gobiernoelectronico.gob.ec/*. Obtenido de https://web.gestiondocumental.gob.ec/que-es-quipux/

Gallo, P. (2011). *Gestión documental en las organizaciones.* Editorial UOC.

Heredia, S. (2022). *Sistema web para la gestión de ventas del taller de reparación y mantenimiento automotriz Heredia utilizando el framework Mean Stack.* Riobamba. Obtenido de Sistema web para la gestión de ventas del taller de reparación y mantenimiento automotriz Heredia utilizando el framework Mean Stack (Tesis de grado).

Kodigo. (s.f.). Obtenido de https://kodigo.org/desglosando-mean-stack-tecnologias-funciones-y-ventajas/

Marco de Desarrollo de la Junta de Andalucía. (s.f.). *Marco de Desarrollo de la Junta de Andalucía*. Obtenido de Introducción a JMeter: Conceptos Básicos: https://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/388

Mardan, A. (2014). *Express.js Guide: The Comprehensive Book on Express.js* . Azat Mardan.

Muradas, Y. (2019). *OpenWebinars*. Obtenido de Qué es Postman y primeros pasos: https://openwebinars.net/blog/que-es-postman/

NG-ZORRO. (s.f.). *ng.ant.design*. Obtenido de https://ng.ant.design/docs/introduce/en#environment-support

Nixon, R. (2018). *Aprender PHP, MYSQL, y JavaScript.* Marcombo S.A.

Ollivier, S., & Gury, P. (2016). *AngularJS Desarrolle hoy las aplicaciones web de mañana.* Barcelona: Ediciones ENI.

Pucciarelli, L. (2020). *Node.js* (Vol. I). Buenos Aires: RedUsers.

Puciarelli, L. (2020). *Angular.* Buenos Aires: Six Ediciones.

Ribas , E. (2018). *IEBS - Thinking for innovation*. Obtenido de ¿Qué es un Api Rest y porque debes implementarla en tu negocio?: https://www.iebschool.com/blog/que-es-api-rest-integrar-negocio-business-tech/

Robledano, A. (23 de Agosto de 2021). *Qué es Javascript*. Obtenido de OpenWebinars: https://openwebinars.net/blog/que-es-javascript/

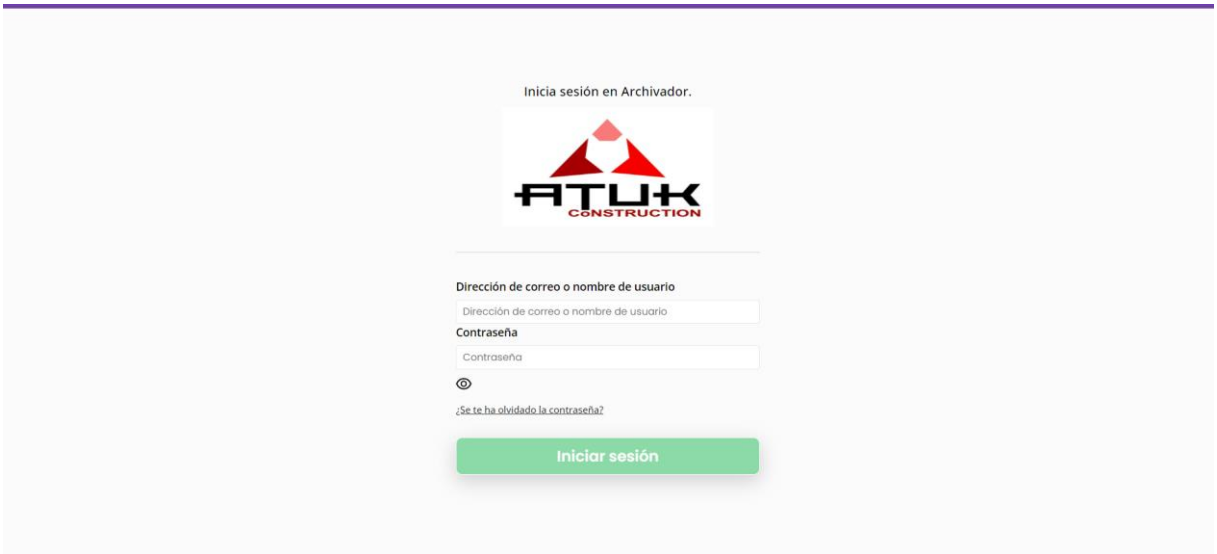
Rubio, P. (2019). *GDX Group*. Obtenido de Problemas y soluciones de gestión documental: https://gdx-group.com/problemas-y-soluciones-de-gestion-documental/

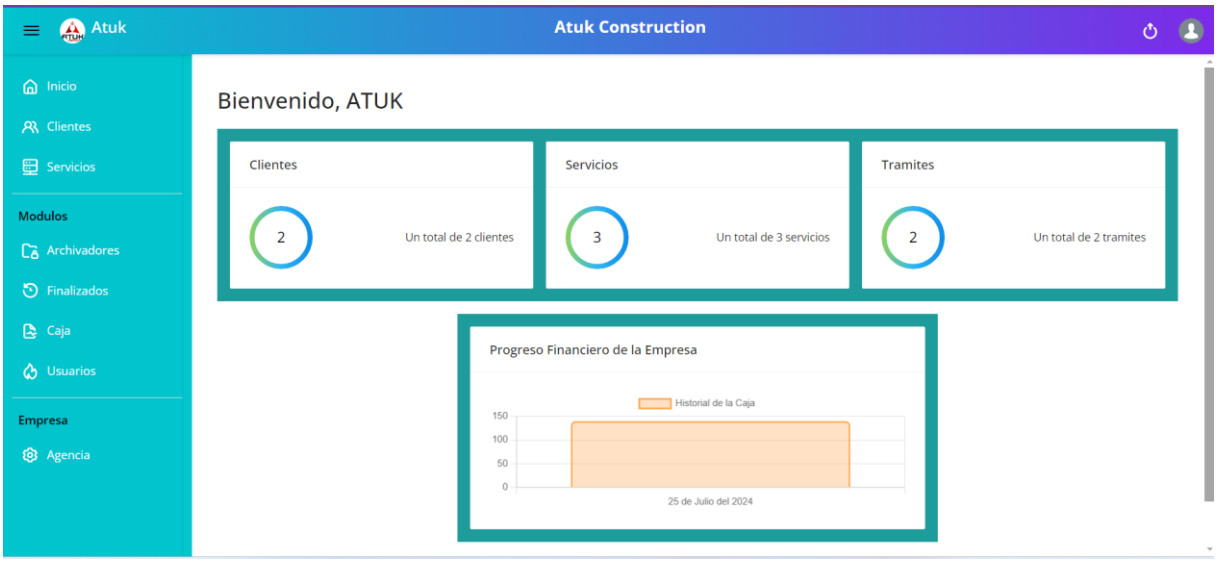
Sayago, J., & Revelo, F. (06 de 01 de 2022). *Aplicaciones web modernas con stack MEAN: Un caso de estudio*. Obtenido de www.researchgate.net/: https://www.researchgate.net/publication/367545997_Aplicaciones_web_modernas_con_stack_MEAN_Un_caso_de_estudio

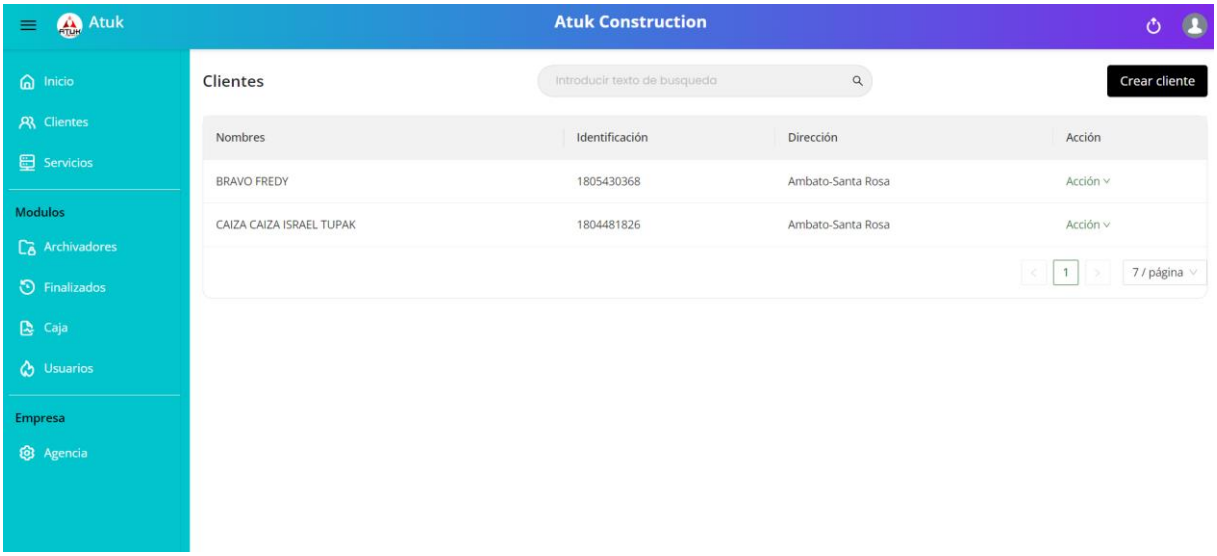
SiteGround Web. (s.f.). *Recursos de conocimiento de SiteGround*. Obtenido de Códigos de estado HTTP explicados - SiteGround KB: https://www.siteground.es/kb/codigos-error-http-explicados/#%C2%BFQue_es_un_codigo_de_estado_HTTP

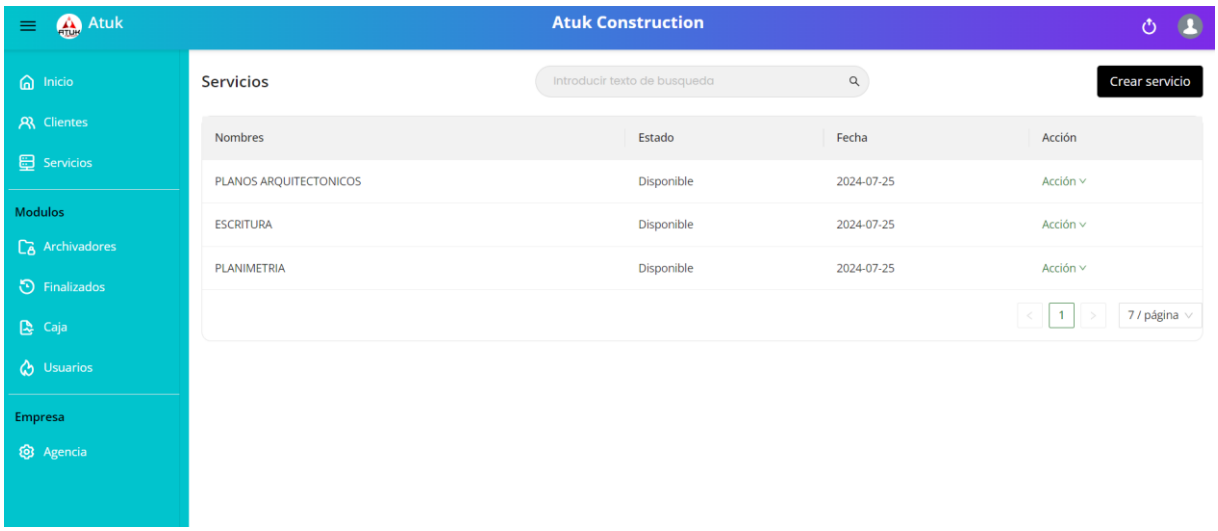
UNIR. (2024). Obtenido de https://unirfp.unir.net/revista/ingenieria-y-tecnologia/stack-mean/#:~:text=Beneficios%20del%20MEAN%20Stack,y%20la%20agilizaci%C3%B3n%20del%20desarrollo.

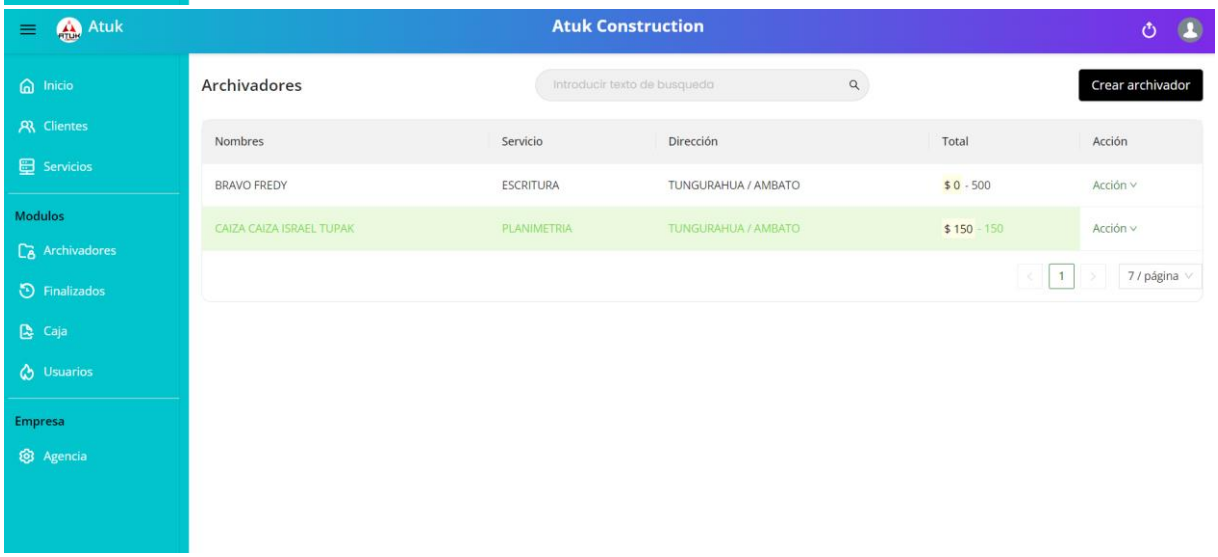
ANEXOS

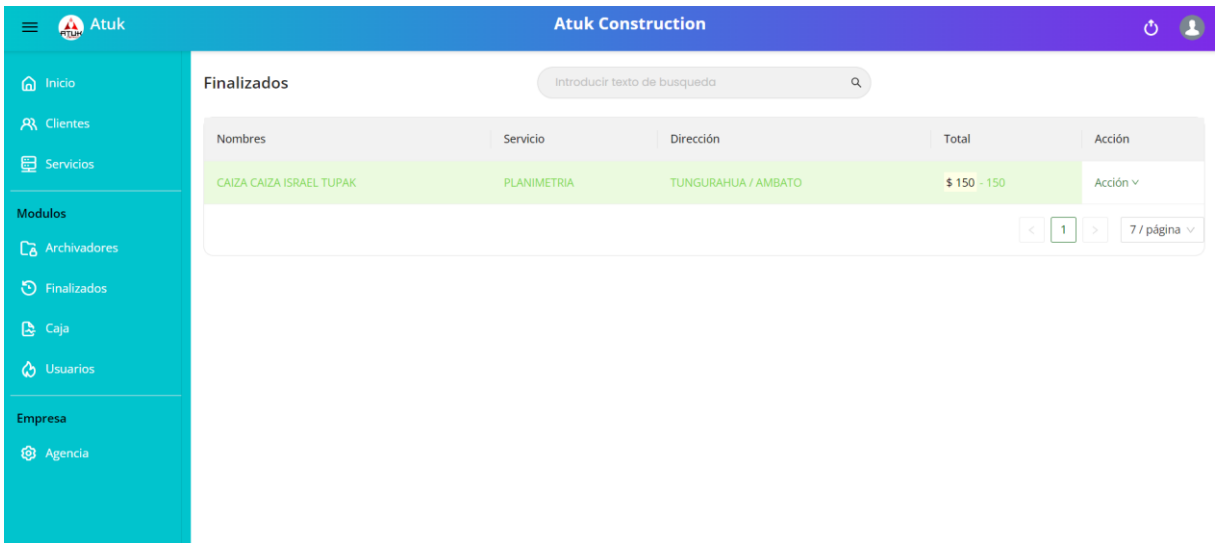

Inicia sesión en Archivador.
ATUK
CONSTRUCTION
Dirección de correo o nombre de usuario
Dirección de correo o nombre de usuario
Contraseña
Contraseña
¿Se te ha olvidado la contraseña?
Iniciar sesión

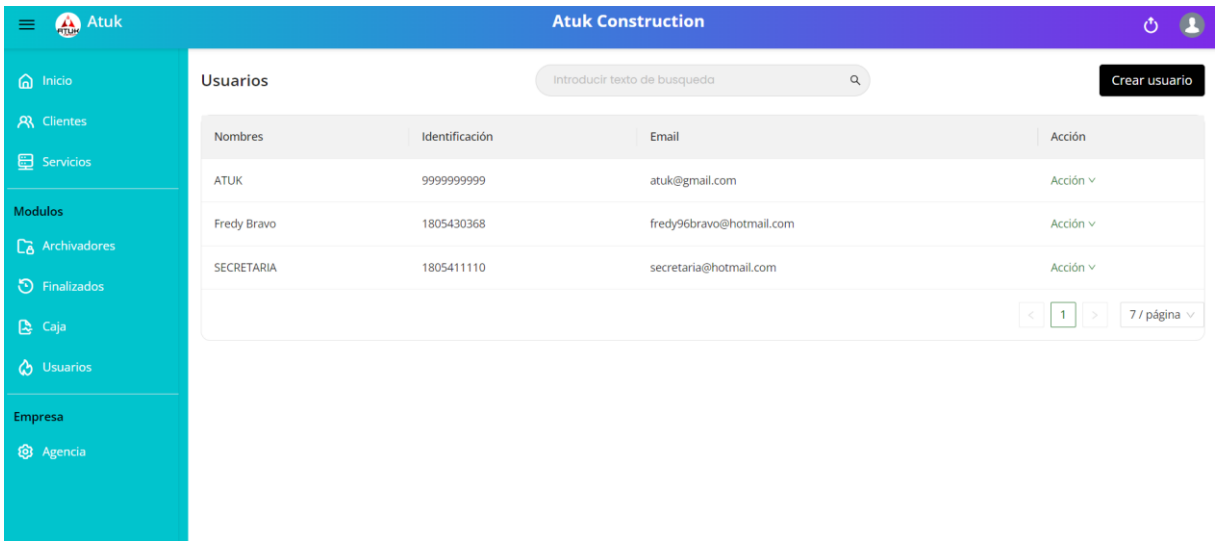

Atuk
Atuk Construction
Inicio
Clientes
Servicios
Modulos
Archivadores
Finalizados
Caja
Usuarios
Empresa
Agencia
Bienvenido, ATUK
Clientes
2
Un total de 2 clientes
Servicios
3
Un total de 3 servicios
Tramites
2
Un total de 2 tramites
Progreso Financiero de la Empresa
Historial de la Caja
150
100
50
0
25 de Julio del 2024

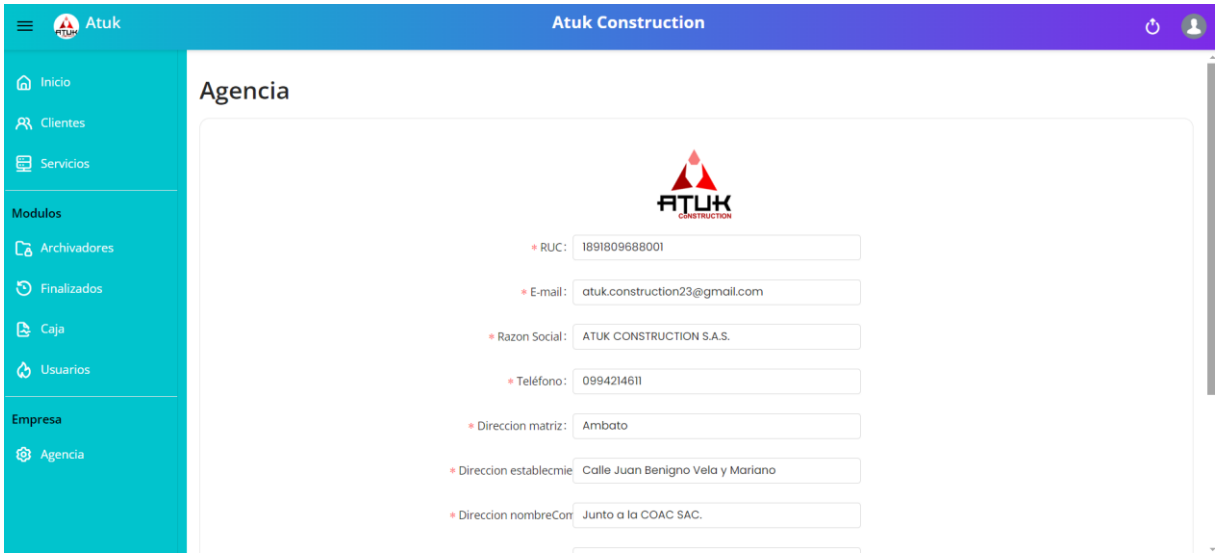

Atuk
Atuk Construction
Inicio
Clientes
Servicios
Modulos
Archivadores
Finalizados
Caja
Usuarios
Empresa
Agencia
Clientes
Introducir texto de busqueda
Crear cliente
Nombres
Identificación
Dirección
Acción
BRAVO FREDY
1805430368
Ambato-Santa Rosa
Acción
CAIZA CAIZA ISRAEL TUPAK
1804481826
Ambato-Santa Rosa
Acción
1
7 / página

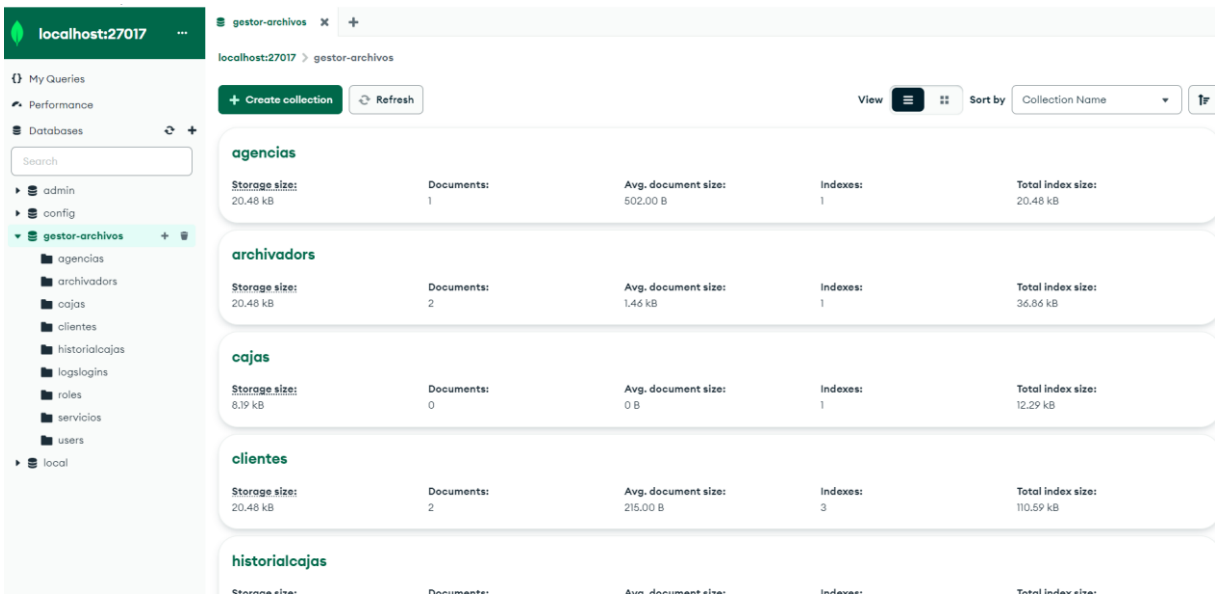

Atuk
Atuk Construction
Inicio
Clientes
Servicios
Modulos
Archivadores
Finalizados
Caja
Usuarios
Empresa
Agencia
Servicios
Introducir texto de busqueda
Crear servicio
Nombres
Estado
Fecha
Acción
PLANOS ARQUITECTONICOS
Disponible
2024-07-25
Acción
ESCRITURA
Disponible
2024-07-25
Acción
PLANIMETRIA
Disponible
2024-07-25
Acción
1
7 / página

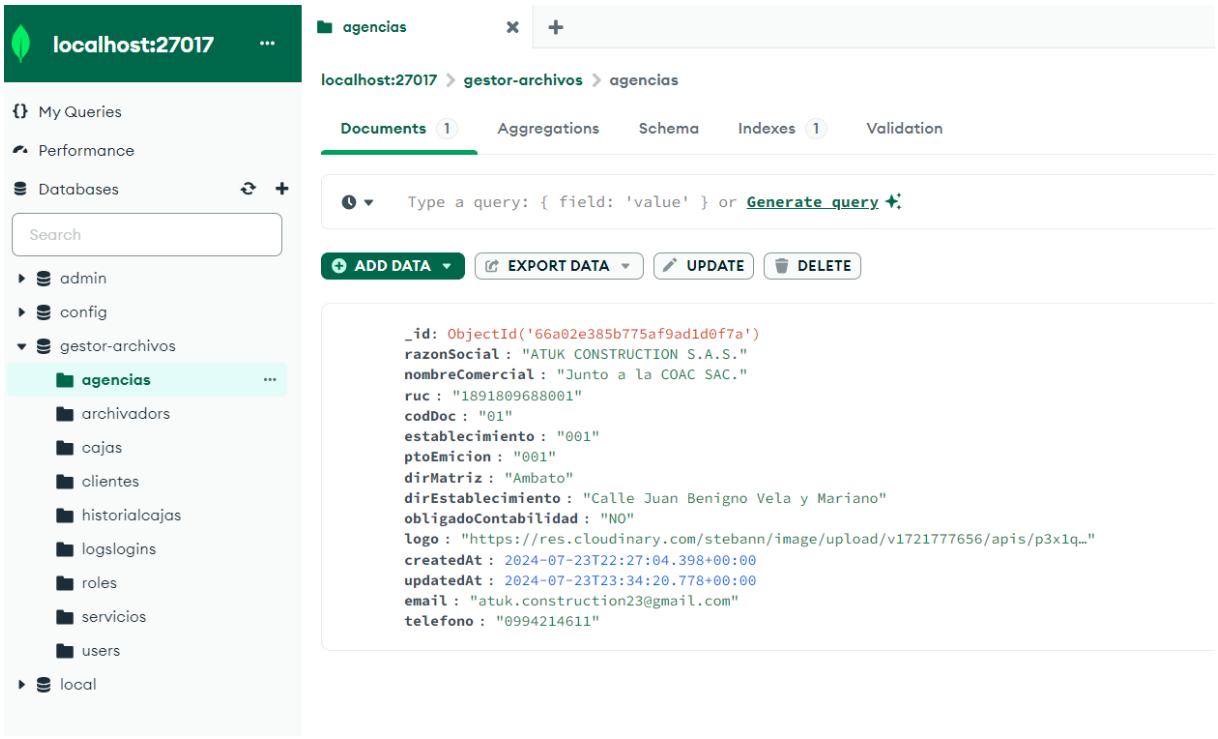

Atuk
Atuk Construction
Inicio
Clientes
Servicios
Modulos
Archivadores
Finalizados
Caja
Usuarios
Empresa
Agencia
Archivadores
Introducir texto de busqueda
Crear archivador
Nombres
Servicio
Dirección
Total
Acción
BRAVO FREDY
ESCRITURA
TUNGURAHUA / AMBATO
$ 0 - 500
Acción
CAIZA CAIZA ISRAEL TUPAK
PLANIMETRIA
TUNGURAHUA / AMBATO
$ 150 - 150
Acción
1
7 / página

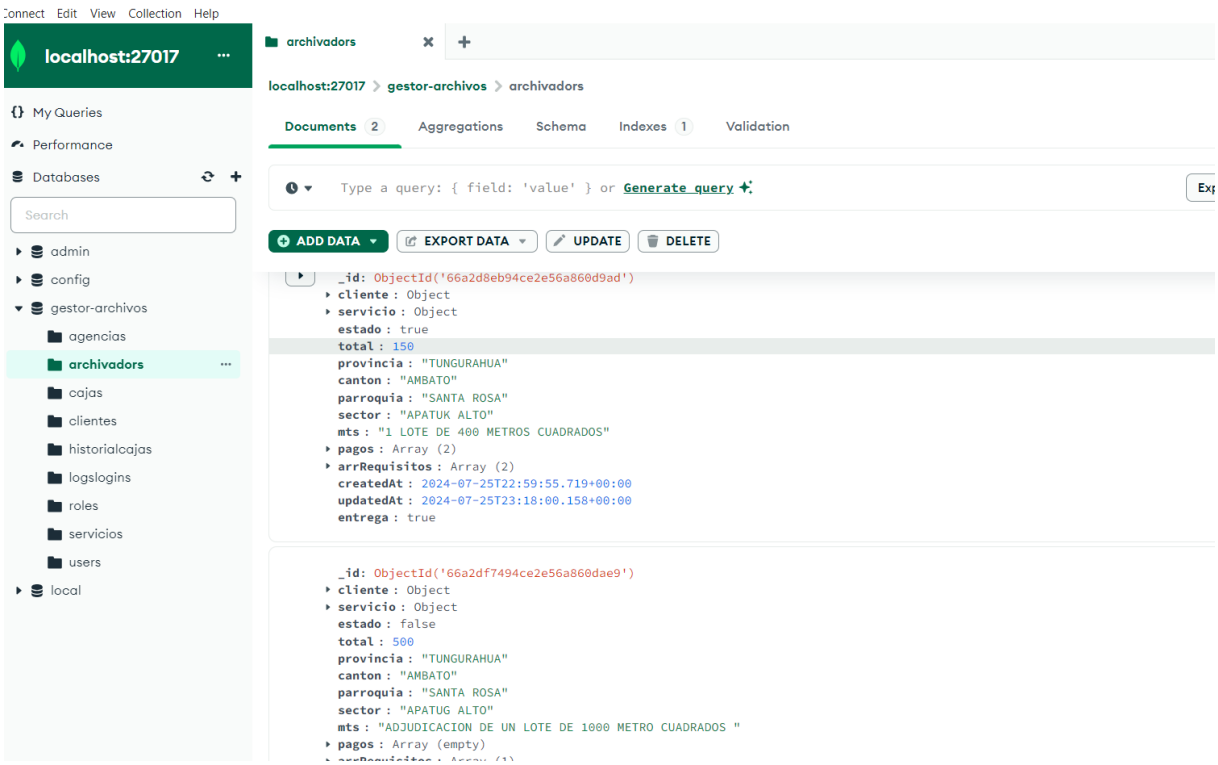

Atuk
Atuk Construction
Inicio
Clientes
Servicios
Modulos
Archivadores
Finalizados
Caja
Usuarios
Empresa
Agencia
Finalizados
Introducir texto de busqueda
Nombres
Servicio
Dirección
Total
Acción
CAIZA CAIZA ISRAEL TUPAK
PLANIMETRIA
TUNGURAHUA / AMBATO
$ 150 - 150
Acción
1
7 / página

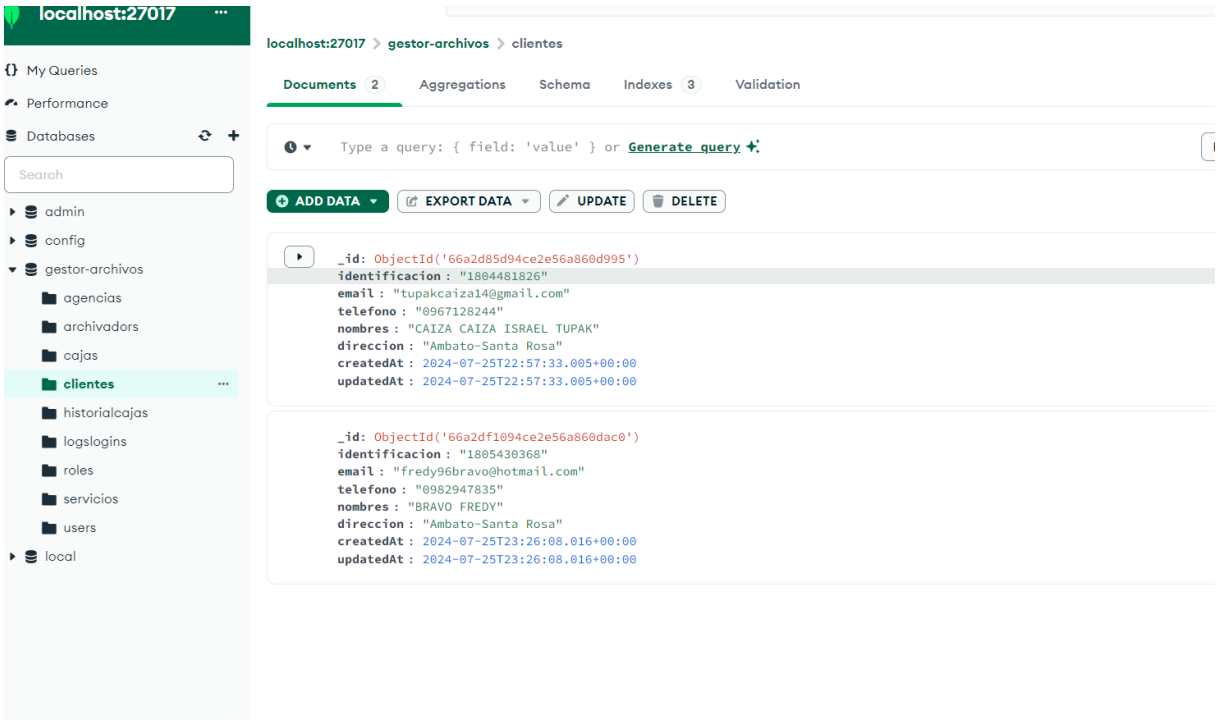

Atuk
Atuk Construction
Inicio
Clientes
Servicios
Modulos
Archivadores
Finalizados
Caja
Usuarios
Empresa
Agencia
Usuarios
Introducir texto de busqueda
Crear usuario
Nombres
Identificación
Email
Acción
ATUK
9999999999
atuk@gmail.com
Acción
Fredy Bravo
1805430368
fredy96bravo@hotmail.com
Acción
SECRETARIA
1805411110
secretaria@hotmail.com
Acción
1
7 / página

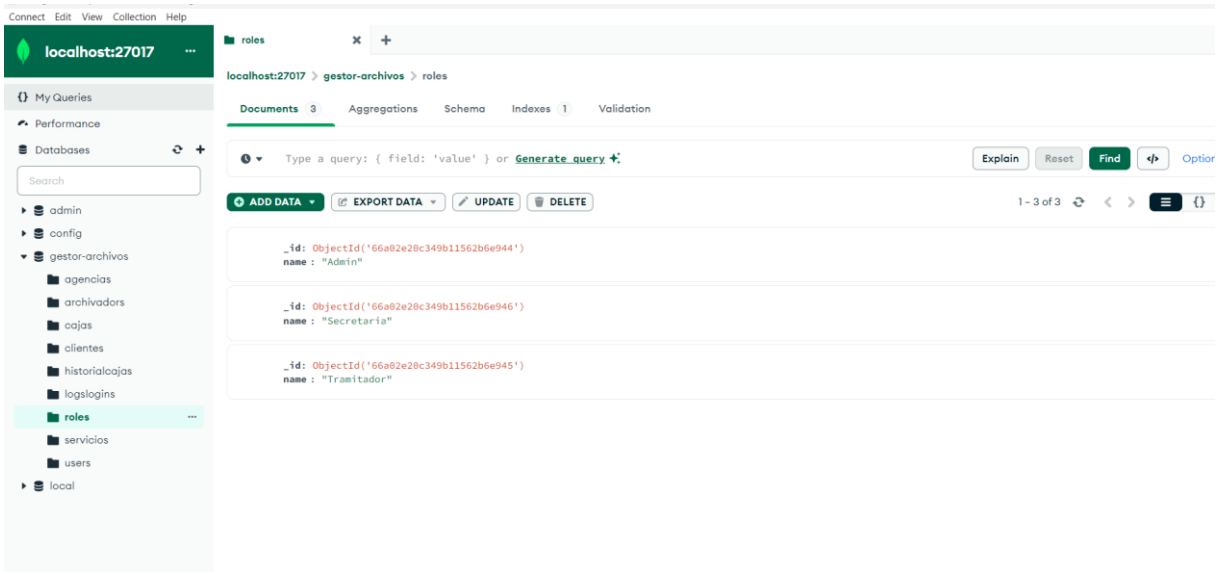

Atuk
Atuk Construction
Inicio
Clientes
Servicios
Modulos
Archivadores
Finalizados
Caja
Usuarios
Empresa
Agencia
Agencia
ATUK
CONSTRUCTION
* RUC:
1891809688001
* E-mail:
atuk.construction23@gmail.com
* Razon Social:
ATUK CONSTRUCTION S.A.S.
* Teléfono:
0994214611
* Direccion matriz:
Ambato
* Direccion establecmie
Calle Juan Benigno Vela y Mariano
* Direccion nombreCom
Junto a la COAC SAC.

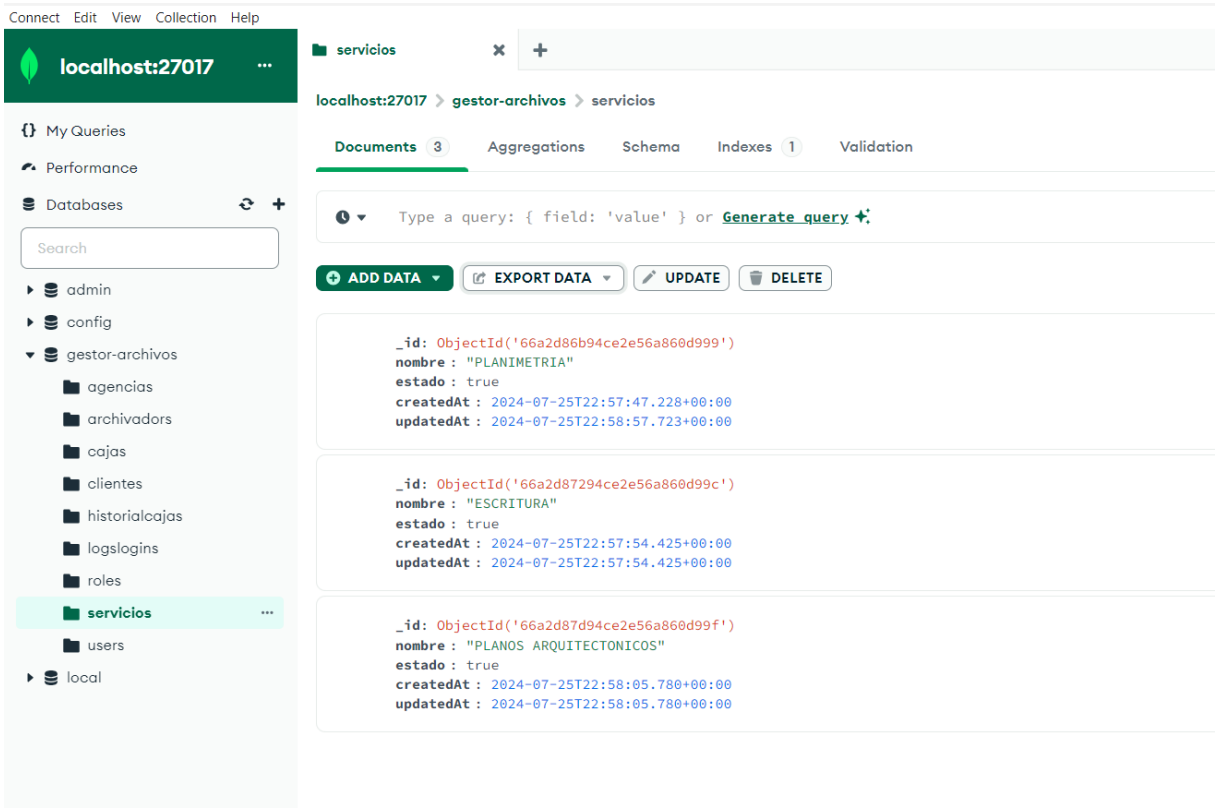

localhost:27017
My Queries
Performance
Databases
Search
admin
config
gestor-archivos
agencias
archivadors
cajas
clientes
historialcajas
logslogins
roles
servicios
users
local
gestor-archivos
localhost:27017 > gestor-archivos
Create collection
Refresh
View
Sort by
Collection Name
agencias
Storage size: 20.48 kB
Documents: 1
Avg. document size: 502.00 B
Indexes: 1
Total index size: 20.48 kB
archivadors
Storage size: 20.48 kB
Documents: 2
Avg. document size: 1.46 kB
Indexes: 1
Total index size: 36.86 kB
cajas
Storage size: 8.19 kB
Documents: 0
Avg. document size: 0 B
Indexes: 1
Total index size: 12.29 kB
clientes
Storage size: 20.48 kB
Documents: 2
Avg. document size: 215.00 B
Indexes: 3
Total index size: 110.59 kB
historialcajas

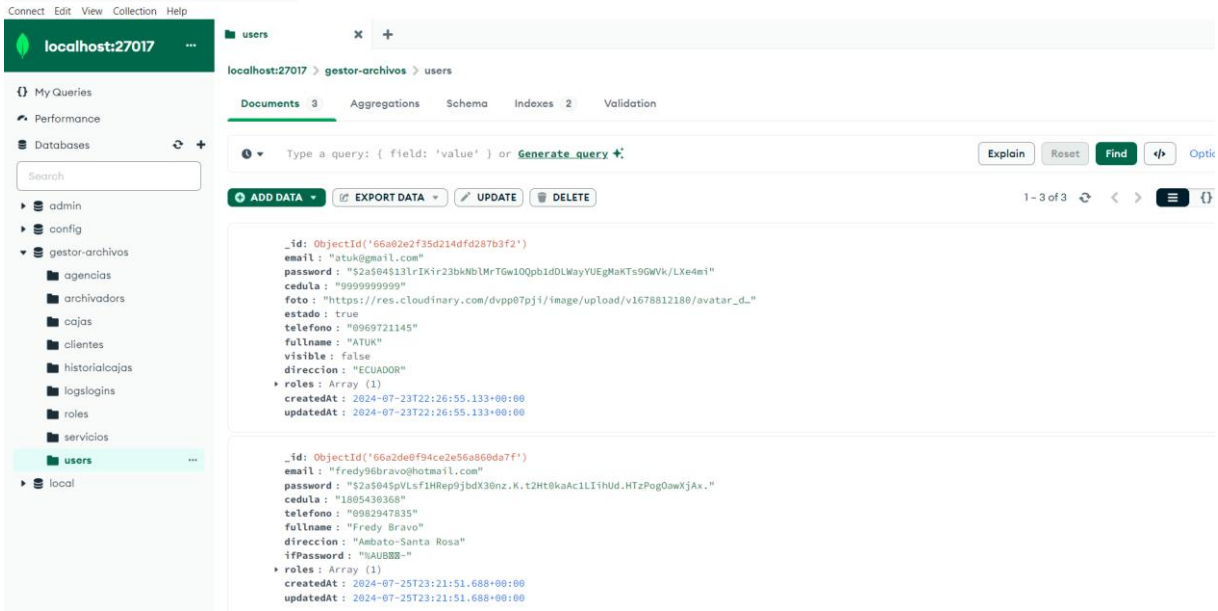

localhost:27017
My Queries
Performance
Databases
Search
admin
config
gestor-archivos
agencias
archivadores
cajas
clientes
historialcajas
logslogins
roles
servicios
users
local
agencias
localhost:27017 > gestor-archivos > agencias
Documents 1
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
ADD DATA
EXPORT DATA
UPDATE
DELETE
_id: ObjectId('66a02e385b775af9ad1d0f7a')
razonSocial : "ATUK CONSTRUCTION S.A.S."
nombreComercial : "Junto a la COAC SAC."
ruc : "1891809688001"
codDoc : "01"
establecimiento : "001"
ptoEmicion : "001"
dirMatriz : "Ambato"
dirEstablecimiento : "Calle Juan Benigno Vela y Mariano"
obligadoContabilidad : "NO"
logo : "https://res.cloudinary.com/stebann/image/upload/v1721777656/apis/p3x1q…"
createdAt : 2024-07-23T22:27:04.398+00:00
updatedAt : 2024-07-23T23:34:20.778+00:00
email : "atuk.construction23@gmail.com"
telefono : "0994214611"

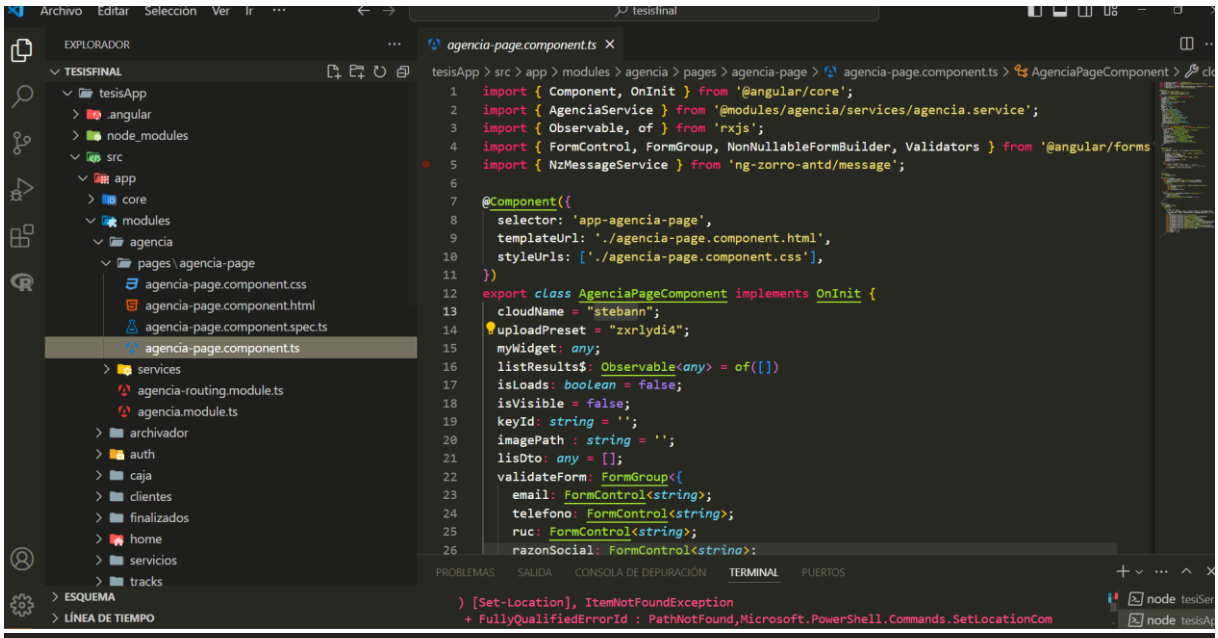

Connect Edit View Collection Help
localhost:27017
My Queries
Performance
Databases
Search
admin
config
gestor-archivos
agencias
archivadores
cajas
clientes
historialcajas
logslogins
roles
servicios
users
local
archivadores
localhost:27017 > gestor-archivos > archivadores
Documents 2
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
ADD DATA
EXPORT DATA
UPDATE
DELETE
_id: ObjectId('66a2d8eb94ce2e56a860d9ad')
cliente : Object
servicio : Object
estado : true
total : 150
provincia : "TUNGURAHUA"
canton : "AMBATO"
parroquia : "SANTA ROSA"
sector : "APATUK ALTO"
mts : "1 LOTE DE 400 METROS CUADRADOS"
pagos : Array (2)
arrRequisitos : Array (2)
createdAt : 2024-07-25T22:59:55.719+00:00
updatedAt : 2024-07-25T23:18:00.158+00:00
entrega : true
_id: ObjectId('66a2df7494ce2e56a860dae9')
cliente : Object
servicio : Object
estado : false
total : 500
provincia : "TUNGURAHUA"
canton : "AMBATO"
parroquia : "SANTA ROSA"
sector : "APATUG ALTO"
mts : "ADJUDICACION DE UN LOTE DE 1000 METRO CUADRADOS "
pagos : Array (empty)

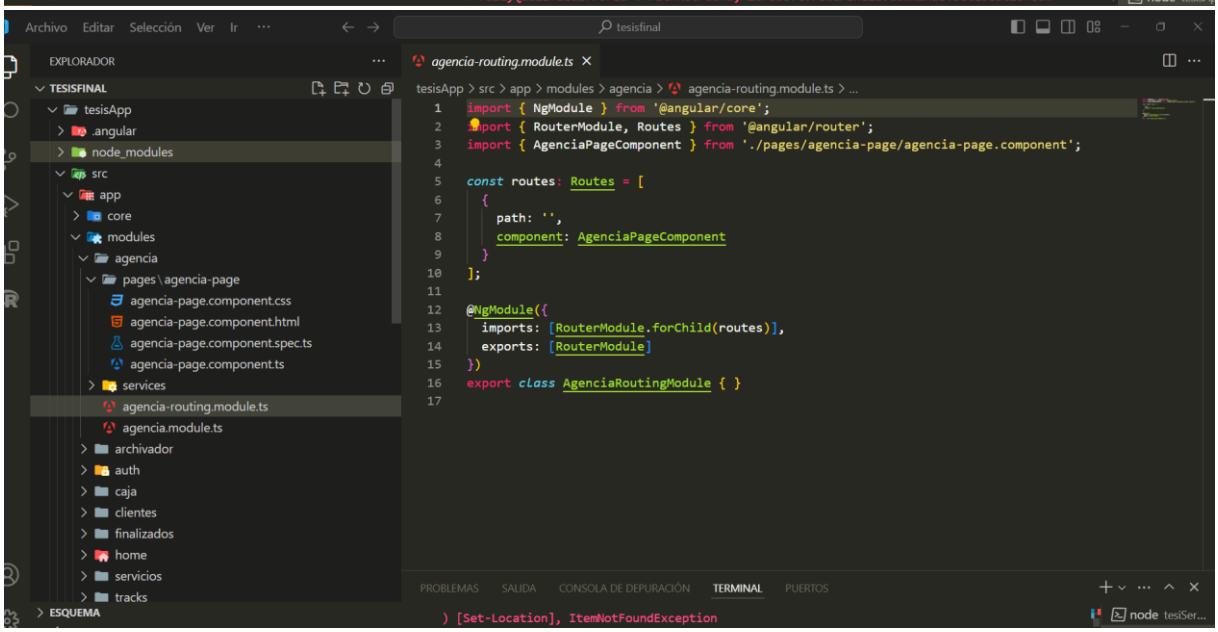

localhost:27017
My Queries
Performance
Databases
Search
admin
config
gestor-archivos
agencias
archivadors
cajas
clientes
historialcajas
logslogins
roles
servicios
users
local
localhost:27017 > gestor-archivos > clientes
Documents 2
Aggregations
Schema
Indexes 3
Validation
Type a query: { field: 'value' } or Generate query
ADD DATA
EXPORT DATA
UPDATE
DELETE
_id: ObjectId('66a2d85d94ce2e56a860d995')
identificacion : "1804481826"
email : "tupakcaiza14@gmail.com"
telefono : "0967128244"
nombres : "CAIZA CAIZA ISRAEL TUPAK"
direccion : "Ambato-Santa Rosa"
createdAt : 2024-07-25T22:57:33.005+00:00
updatedAt : 2024-07-25T22:57:33.005+00:00
_id: ObjectId('66a2df1094ce2e56a860dac0')
identificacion : "1805430368"
email : "fredy96bravo@hotmail.com"
telefono : "0982947835"
nombres : "BRAVO FREDY"
direccion : "Ambato-Santa Rosa"
createdAt : 2024-07-25T23:26:08.016+00:00
updatedAt : 2024-07-25T23:26:08.016+00:00

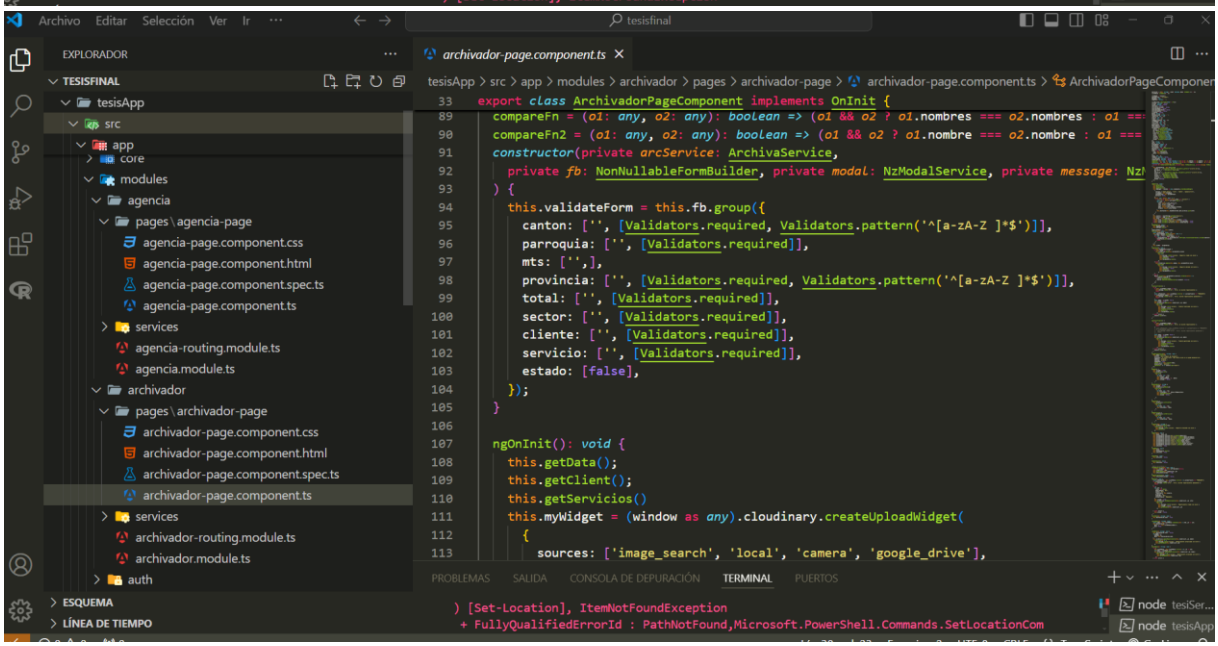

Connect Edit View Collection Help
localhost:27017
My Queries
Performance
Databases
Search
admin
config
gestor-archivos
agencias
archivadors
cajas
clientes
historialcajas
logslogins
roles
servicios
users
local
roles
localhost:27017 > gestor-archivos > roles
Documents 3
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
ADD DATA
EXPORT DATA
UPDATE
DELETE
1 - 3 of 3
_id: ObjectId('66a02e20c349b11562b6e944')
name : "Admin"
_id: ObjectId('66a02e20c349b11562b6e946')
name : "Secretaria"
_id: ObjectId('66a02e20c349b11562b6e945')
name : "Tramitador"

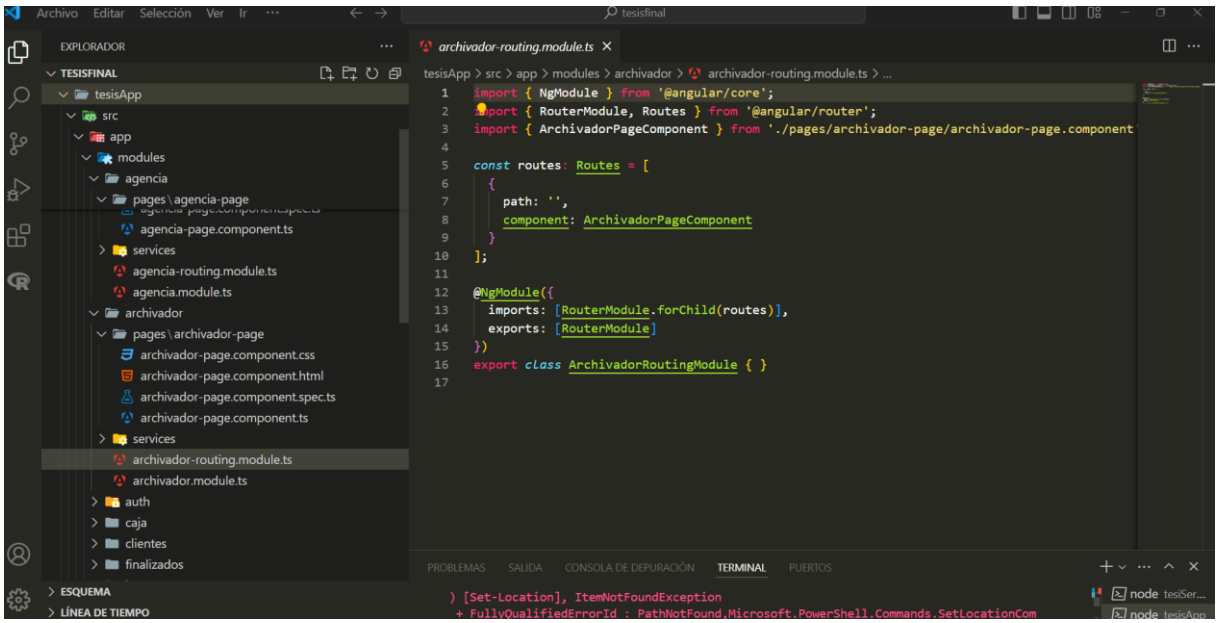

Connect Edit View Collection Help
localhost:27017
My Queries
Performance
Databases
Search
admin
config
gestor-archivos
agencias
archivadors
cajas
clientes
historialcajas
logslogins
roles
servicios
users
local
servicios
localhost:27017 > gestor-archivos > servicios
Documents 3
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
ADD DATA
EXPORT DATA
UPDATE
DELETE
_id: ObjectId('66a2d86b94ce2e56a860d999')
nombre : "PLANIMETRIA"
estado : true
createdAt : 2024-07-25T22:57:47.228+00:00
updatedAt : 2024-07-25T22:58:57.723+00:00
_id: ObjectId('66a2d87294ce2e56a860d99c')
nombre : "ESCRITURA"
estado : true
createdAt : 2024-07-25T22:57:54.425+00:00
updatedAt : 2024-07-25T22:57:54.425+00:00
_id: ObjectId('66a2d87d94ce2e56a860d99f')
nombre : "PLANOS ARQUITECTONICOS"
estado : true
createdAt : 2024-07-25T22:58:05.780+00:00
updatedAt : 2024-07-25T22:58:05.780+00:00

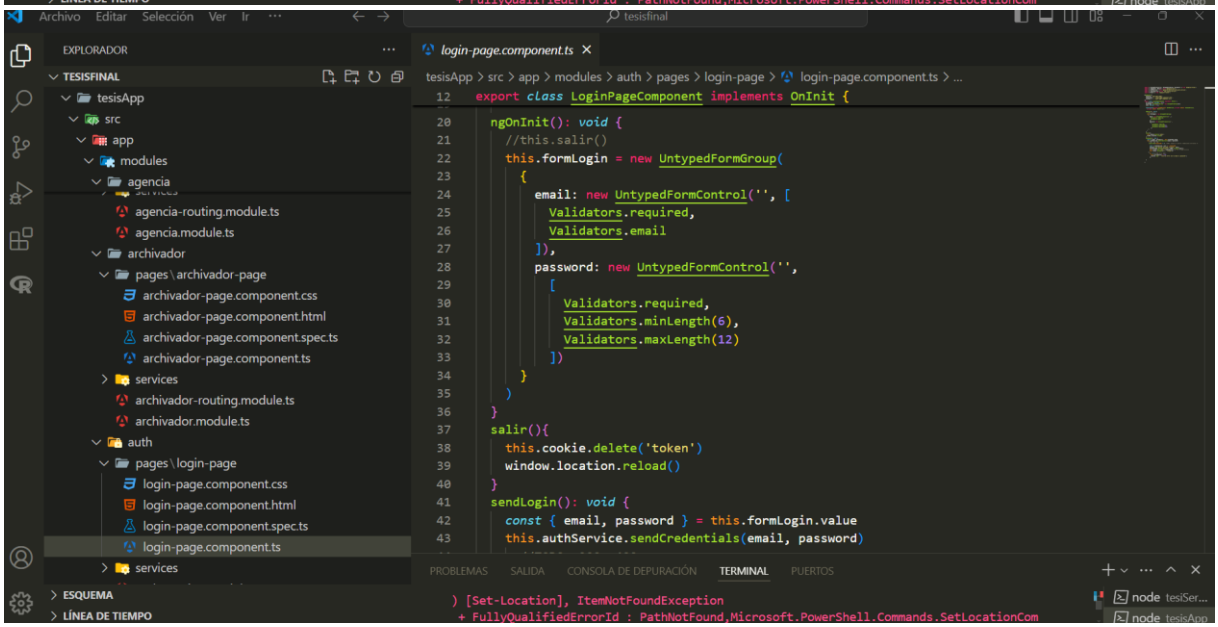

Connect Edit View Collection Help
localhost:27017
My Queries
Performance
Databases
Search
admin
config
gestor-archivos
agencias
archivadors
cajas
clientes
historialcajas
logslogins
roles
servicios
users
local
users
localhost:27017 > gestor-archivos > users
Documents 3
Aggregations
Schema
Indexes 2
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
ADD DATA
EXPORT DATA
UPDATE
DELETE
1 - 3 of 3
_id: ObjectId('66a02e2f35d214dfd287b3f2')
email : "atuk@gmail.com"
password : "$2a$04$13lrIKir23bkNblMrTGw1OQpb1dDLWayYUEgMaKTs9GWVk/LXe4mi"
cedula : "9999999999"
foto : "https://res.cloudinary.com/dvpp07pji/image/upload/v1678812180/avatar_d…"
estado : true
telefono : "0969721145"
fullname : "ATUK"
visible : false
direccion : "ECUADOR"
roles : Array (1)
createdAt : 2024-07-23T22:26:55.133+00:00
updatedAt : 2024-07-23T22:26:55.133+00:00
_id: ObjectId('66a2de0f94ce2e56a860da7f')
email : "fredy96bravo@hotmail.com"
password : "$2a$04$pVLsf1HRep9jbdX30nz.K.t2Ht0kaAc1LIihUd.HTzPogOawXjAx."
cedula : "1805430368"
telefono : "0982947835"
fullname : "Fredy Bravo"
direccion : "Ambato-Santa Rosa"
ifPassword : "%AUB[illegible]-"
roles : Array (1)
createdAt : 2024-07-25T23:21:51.688+00:00
updatedAt : 2024-07-25T23:21:51.688+00:00

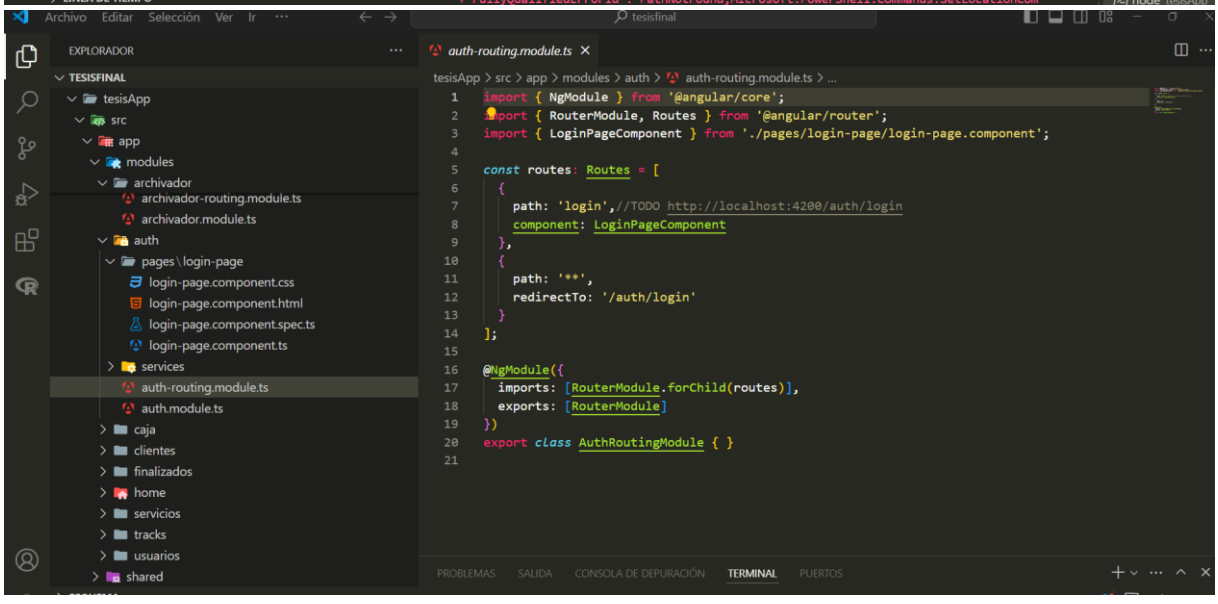

agencia-page.component.ts
import { Component, OnInit } from '@angular/core';
import { AgenciaService } from '@modules/agencia/services/agencia.service';
import { Observable, of } from 'rxjs';
import { FormControl, FormGroup, NonNullableFormBuilder, Validators } from '@angular/forms
import { NzMessageService } from 'ng-zorro-antd/message';
@Component({
selector: 'app-agencia-page',
templateUrl: './agencia-page.component.html',
styleUrls: ['./agencia-page.component.css'],
})
export class AgenciaPageComponent implements OnInit {
cloudName = "stebann";
uploadPreset = "zxrlydi4";
myWidget: any;
listResults$: Observable<any> = of([])
isLoads: boolean = false;
isVisible = false;
keyId: string = '';
imagePath : string = '';
lisDto: any = [];
validateForm: FormGroup<{
email: FormControl<string>;
telefono: FormControl<string>;
ruc: FormControl<string>;
razonSocial: FormControl<strina>;
) [Set-Location], ItemNotFoundException
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCom

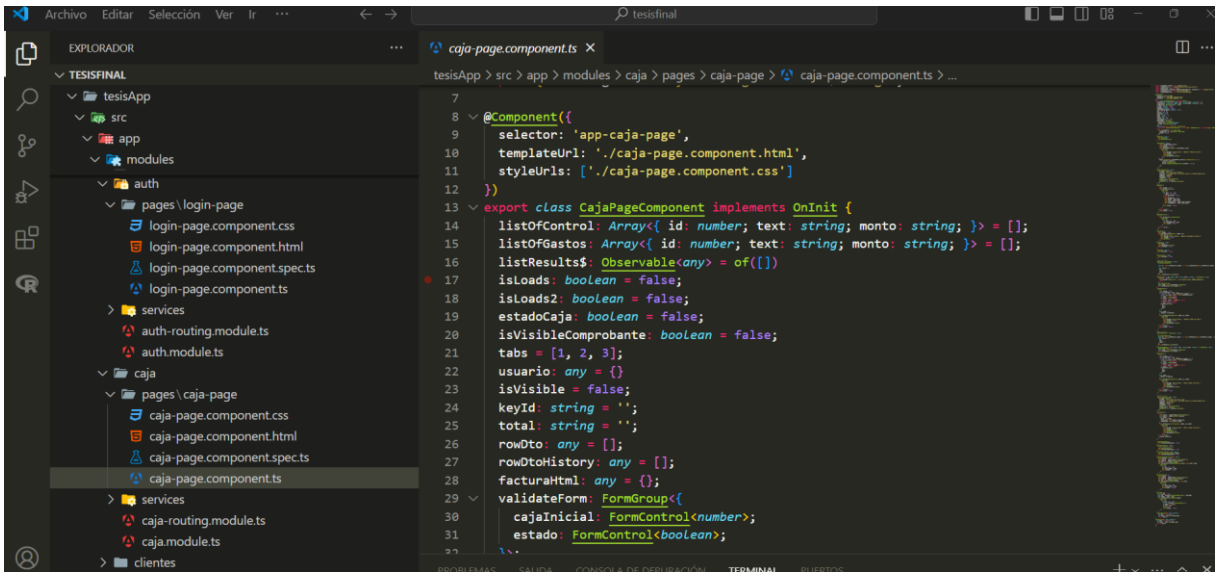

agencia-routing.module.ts
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AgenciaPageComponent } from './pages/agencia-page/agencia-page.component';
const routes: Routes = [
{
path: '',
component: AgenciaPageComponent
}
];
@NgModule({
imports: [RouterModule.forChild(routes)],
exports: [RouterModule]
})
export class AgenciaRoutingModule { }
) [Set-Location], ItemNotFoundException


archivador-page.component.ts
export class ArchivadorPageComponent implements OnInit {
compareFn = (o1: any, o2: any): boolean => (o1 && o2 ? o1.nombres === o2.nombres : o1 ===
compareFn2 = (o1: any, o2: any): boolean => (o1 && o2 ? o1.nombre === o2.nombre : o1 ===
constructor(private arcService: ArchivaService,
private fb: NonNullableFormBuilder, private modal: NzModalService, private message: Nz
) {
this.validateForm = this.fb.group({
canton: ['', [Validators.required, Validators.pattern('^[a-zA-Z]*$')]],
parroquia: ['', [Validators.required]],
mts: ['',],
provincia: ['', [Validators.required, Validators.pattern('^[a-zA-Z]*$')]],
total: ['', [Validators.required]],
sector: ['', [Validators.required]],
cliente: ['', [Validators.required]],
servicio: ['', [Validators.required]],
estado: [false],
});
}
ngOnInit(): void {
this.getData();
this.getClient();
this.getServicios()
this.myWidget = (window as any).cloudinary.createUploadWidget(
{
sources: ['image_search', 'local', 'camera', 'google_drive'],
) [Set-Location], ItemNotFoundException
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCom


archivador-routing.module.ts
tesisApp > src > app > modules > archivador > archivador-routing.module.ts > ...
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { ArchivadorPageComponent } from './pages/archivador-page/archivador-page.component'
const routes: Routes = [
{
path: '',
component: ArchivadorPageComponent
}
];
@NgModule({
imports: [RouterModule.forChild(routes)],
exports: [RouterModule]
})
export class ArchivadorRoutingModule { }
TERMINAL


login-page.component.ts
tesisApp > src > app > modules > auth > pages > login-page > login-page.component.ts > ...
export class LoginPageComponent implements OnInit {
ngOnInit(): void {
//this.salir()
this.formLogin = new UntypedFormGroup(
{
email: new UntypedFormControl('', [
Validators.required,
Validators.email
]),
password: new UntypedFormControl('',
[
Validators.required,
Validators.minLength(6),
Validators.maxLength(12)
])
}
)
}
salir(){
this.cookie.delete('token')
window.location.reload()
}
sendLogin(): void {
const { email, password } = this.formLogin.value
this.authService.sendCredentials(email, password)
TERMINAL


auth-routing.module.ts
tesisApp > src > app > modules > auth > auth-routing.module.ts > ...
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { LoginPageComponent } from './pages/login-page/login-page.component';
const routes: Routes = [
{
path: 'login',//TODO http://localhost:4200/auth/login
component: LoginPageComponent
},
{
path: '**',
redirectTo: '/auth/login'
}
];
@NgModule({
imports: [RouterModule.forChild(routes)],
exports: [RouterModule]
})
export class AuthRoutingModule { }
TERMINAL


Archivo Editar Selección Ver Ir
tesisfinal
EXPLORADOR
caja-page.component.ts
TESISFINAL
tesisApp > src > app > modules > caja > pages > caja-page > caja-page.component.ts > ...
tesisApp
src
app
modules
auth
pages \ login-page
login-page.component.css
login-page.component.html
login-page.component.spec.ts
login-page.component.ts
services
auth-routing.module.ts
auth.module.ts
caja
pages \ caja-page
caja-page.component.css
caja-page.component.html
caja-page.component.spec.ts
caja-page.component.ts
services
caja-routing.module.ts
caja.module.ts
clientes
@Component({
selector: 'app-caja-page',
templateUrl: './caja-page.component.html',
styleUrls: ['./caja-page.component.css']
})
export class CajaPageComponent implements OnInit {
listOfControl: Array<{ id: number; text: string; monto: string; }> = [];
listOfGastos: Array<{ id: number; text: string; monto: string; }> = [];
listResults$: Observable<any> = of([])
isLoads: boolean = false;
isLoads2: boolean = false;
estadoCaja: boolean = false;
isVisibleComprobante: boolean = false;
tabs = [1, 2, 3];
usuario: any = {}
isVisible = false;
keyId: string = '';
total: string = '';
rowDto: any = [];
rowDtoHistory: any = [];
facturaHtml: any = {};
validateForm: FormGroup<{
cajaInicial: FormControl<number>;
estado: FormControl<boolean>;