Artículo de Investigación. Revista Killkana Técnica. Vol. 6, No. 1, pp. 31-42, enero-abril, 2022.
ISSN 2528-8024. ISSN Elect. 2588-0888. Universidad Católica de Cuenca

Aplicaciones web modernas con stack MEAN: Un caso de estudio

Modern web applications with MEAN stack: A case study

Jaime Sayago Heredia[1], Fernanda Revelo Bautista[1]

[1] Pontificia Universidad Católica del Ecuador. Sede Esmeraldas, Sede Esmeraldas, Espejo y subida a Santa Cruz Casilla 08-01-0065, 0984787662,

* jaime.sayago@pucese.edu.ec
DOI: https://doi.org/10.26871/killkanatecnica.v6i1.989

Resumen

En la actualidad, las tecnologías como JavaEE estan presentes en la materia de desarrollo basado en plataformas de la Pontificia Universidad Católica del Ecuador, Sede Esmeraldas – Escuela de Sistemas y Computación y Tecnologías de la Información. La parte más crucial en un proyecto de desarrollo web basado en servicios REST es la elección de las herramientas correctas para el front-end, back-end y entorno de base de datos. El objetivo principal de esta investigación es presentar la arquitectura de aplicaciones web modernas basadas en el stack MEAN junto con

sus componentes e integración con otras tecnologías web y su comparación con la implementación del del stack JavaEE. Hemos realizado un análisis comparativo de la implementación de conformada por MongoDB (base de datos), Node.js. (servidor web), Express (back-end) y Angular (front-end). El resultado de la comparativa y el respectivo análisis de las herramientas seleccionadas servirán a los desarrolladores de software a realizar una mejor elección de la tecnología y la arquitectura adecuadas, en función de los requisitos de la aplicación que están desarrollando.

Palabras clave: *Arquitectura software, Javascript, REST, Servicio Web, Java EE.*

Abstract

Nowadays, technologies such as JavaEE are present in the different platform-based development courses of the Pontificia Universidad Católica del Ecuador, Sede Es-meraldas - School of Systems and Computing and Information Technologies. The most crucial part in a web development project based on REST services is the choice of the right tools for the front-end, back-end and database environment. The main objective of this research is to present the architecture of modern web applications based on the MEAN stack along with its components and integration with other web technologies and its comparison with the implementation of the JavaEE stack. We have performed a comparative analysis of the implementation of the MEAN stack consisting of MongoDB (database), Node.js. (web server), Express (back-end) and Angular (front-end). The result of the comparison and the respective analysis of the selected tools will help software developers to make a better choice of the appropriate technology and architecture, depending on the requirements of the application they are developing.

Keywords: *Software architecture, Javascript, REST, Web Service, Java EE.*

1. Introducción

Nuestra motivación para esta investigación basada en el análisis de las nuevas tecnologías JavaScript, es mejorar la materia de la carrera Tecnologías de la Información en la materia de desarrollo basado en plataformas y encontrar arquitecturas de software alternativas para la implementación una aplicación web moderna. El problema radica que al momento de elegir la arquitectura apropiada de tecnologías para desarrollo, integración y ejecución de aplicaciones web. La elección no es fácil, muchos parámetros tienen que considerarse en la base de datos, en el lado del servidor o el lado del cliente, etc. Diversos autores presentan a la stack MEAN como una solución [1] [2] [3]. JavaScript [4]ha ganado una creciente popularidad en el de unos pocos años, lo cual es una hazaña en sí misma. El lenguaje JavaScript es compatible con la arquitectura de software Modelo-Vista-Controlador que mantiene un código legible y separa claramente las partes del código del programa. Angular como uno de los frameworks más populares de JavaScript se construyó para realizar un trabajo rápido y ágil trabajo en equipos de software [5]. Los autores realizaron un análisis exhaustivo de los frameworks existentes basados en JavaScript [6], y se puede concluir que Angular es uno de los frameworks más populares, utilizados y robustos [7] y es necesario que la materia de desarrollo basado en plataformas se encuentre a la vanguardia de las tecnologías para desarrollo web moderno. Además de conocer si MEAN es la solución más efectiva para el desarrollo de aplicaciones web modernas. Se realizó la implementación de una aplicación web para el manejo de la hoja de vida y gestión docente para el departamento de talento humano de la Pontificia Universidad Católica del Ecuador PUCE, Sede Esmeraldas – Ecuador, que permite almacenar, gestionar y analizar la información de los docentes y su gestión docente semestral. Como conclusión del caso de estudio es proponer la aplicación de estas tecnologías dentro de la materia desarrollo basado en plataformas en la universidad y recomendar su uso en el desarrollo de aplicaciones web modernas.

2. Metodología

La metodología de esta investigación incluyó las siguientes fases:

- Análisis bibliográfico de las arquitecturas y tecnologías Javascript para desarrollo web moderno: MongoDb, Express, Angular, NodeJS.
- Modelado de la aplicación en web basada en las tecnologías del estudior: MEAN
- Aplicación de un modelo dentro de la materia desarrollo basada en plataformas junto con las tecnologías JavaEE estudiadas.
- Los pros y los contras del análisis basado en la literatura abierta, los documentos y los modelos de aplicación web realizados.

2.1 Arquitectura

Cada sistema informático, grande o pequeño, está formado por piezas que están unidas entre sí. Puede haber un pequeño número de estas piezas, o tal vez solo una, o puede haber docenas o cientos; y este vínculo puede ser trivial, o muy complicado, o en algún punto intermedio, es decir cada sistema tiene una arquitectura [8]. Al tratar de definir el concepto de Arquitectura de Software existen varias definiciones alternativas o contrapuestas. Pero hay algunas que son reconocidas, a continuación, las citamos. Se puede definir la arquitectura de software como la descomposición de un sistema de nivel superior en cada uno de sus componentes principales, las interfaces y su comunicación [9]. La

IEEE la define de la siguiente manera: Es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución [10]. Dentro de las ventajas de la arquitectura de software, podemos mencionar que: simplifica la capacidad de comprender sistemas complejos planteando limitaciones en el diseño, reutiliza componentes en múltiples niveles, separa elementos, lo que permite mayor facilidad en el cambio y mantenimiento del desarrollo, aplica utilización de patrones [11]. El uso de arquitectura de software puede realizarse de una manera eficiente en función del tiempo y ser rentable ya que permite la reutilización de los componentes y patrones de diseño en los proyectos [12]. Al implementar una arquitectura de software es fundamental el uso de los patrones arquitectónicos que brindan un principio general de estructura. Un patrón arquitectónico expresa un esquema de organización estructural fundamental para los sistemas de software. Proporciona un conjunto de subsistemas predefinidos, especifica sus responsabilidades e incluye reglas y pautas para organizar las relaciones entre ellos [13].

2.2 Arquitectura REST

REpresentational State Transfer (REST) es un estilo arquitectónico propuesto por Fielding [14]. REST es para sistemas hipermedias distribuidos a gran escala y que logra que la World Wide Web (WWW) sea escalable. Fielding argumenta que en REST es la existencia de recursos (elementos de información), que pueden ser accedidos utilizando un identificador global (un Identificador Uniforme de Recurso). Para manipular estos recursos, los componentes de la red (clientes y servidores) se comunican a través de una interfaz estándar (HTTP) e intercambian representaciones de estos recursos (los ficheros que se descargan y se envían). El cliente puede navegar esencialmente a través de una amplia gama de recursos existentes, siguiendo los enlaces de un recurso a recurso [15]. Un principio clave de REST es la interacción sin estado entre los participantes en una conversación. Un estado en este caso significa el estado de la aplicación/sesión. El estado se mantiene como parte del contenido transferido del cliente al servidor/servicio y viceversa [16]. Más concretamente, en el caso de los servicios, los clientes que desean utilizar un servicio acceden a una representación particular de los recursos que representan el servicio mediante la transferencia de contenido de la aplicación utilizando un conjunto pequeño y definido globalmente de métodos remotos [16]. Estos métodos describen la acción a realizar sobre los recursos. Los métodos HTTP para crear, leer, actualizar y borrar recursos, cada uno identificado por un URI, son (PUT, POST, GET, y DELETE en HTTP 1.0, mientras que HTTP 1.1 permite extensiones) [17]. En REST, cada solicitud enviada a un objeto resulta en la transferencia de una representación de este objeto por ejemplo, texto, XML, JSON, etc. [15]. REST se ha convertido en la implementación más utilizada en la actualidad [18].

3. MEAN

La stack MEAN es una solución potente y completa. Utiliza el lenguaje de programación JavaScript [19]. Comprende cuatro bloques principales: MongoDB como base de datos, Express como marco de trabajo del servidor web, AngularJS como marco de trabajo del cliente web y Node.js como plataforma del servidor. Estos componentes están desarrollados por diferentes equipos e involucran a una comunidad fuerte de desarrolladores y defensores impulsando el desarrollo y documentación de cada componente. Sin embargo, un

problema que podría afectar dramáticamente su proceso de desarrollo y presentar problemas de escalamiento y arquitectura es la conexión de estas herramientas [20]. La principal fortaleza de MEAN radica en su centralización de JavaScript como el principal lenguaje de programación, ya que cada componente está escrito en JavaScript, incluso la base de datos almacena los datos en formato JavaScript Object Notation (JSON) que es el único script que JavaScript entiende completamente [21]. Por lo tanto, JavaScript no sólo se utiliza como lenguaje de scripting del lado del cliente, sino que también se utiliza a lo largo de la aplicación, es decir, en el lado del cliente, del servidor y de la base de datos [1]. El uso de JavaScript como lenguaje principal de programación tanto en el lado del cliente como en el lado del servidor hace que la pila MEAN sea más potente y reduce el tiempo en la construcción de la aplicación [2]. El uso de todo el JavaScript permite dividir la funcionalidad y las tecnologías utilizadas de la siguiente manera [21]:

- Base de datos objetos JSON utiliza MongoDB.
- Servidor web utiliza Node.js.
- Framework web (back-end) utiliza Express.
- Cliente (front-end) utiliza Angular.

3.1 MongoDB

MongoDB es un modelo de almacenamiento de documentos NoSQL potente, adaptable y escalable [22]. En MongoDB, los datos se almacenan en la base de datos en un formato JSON llamado BJSON, que significa JSON binario, en lugar de filas y columnas como en la base de datos relacional. MongoDB tiene una capacidad para escalar con varias características que la base de datos relacional proporciona como índices secundarios, clasificación y consultas [23], que proporciona alto rendimiento, alta disponibilidad y escalabilidad [20].

3.2 Express

Express es un framework web maduro y flexible para construir aplicaciones web sobre el ecosistema Node. Por defecto, el framework Express utiliza el motor Pug para soportar plantillas [24]. Express es un framework relativamente pequeño que se encuentra en la parte superior de la funcionalidad del servidor web de Node para simplificar sus APIs y añadir nuevas funciones útiles. Facilita la organización de la funcionalidad de su aplicación con middleware y enrutamiento; agrega utilidades útiles a los objetos HTTP de Node y el renderizado de vistas HTML dinámicas; define un estándar de extensibilidad fácilmente implementado [25] . Express es fácil de configurar, implementar, controlar y proporcionar varios componentes clave para manejar las solicitudes web. Express ayuda en la creación de aplicaciones web y servidores HTTP simples ya que es un framework mínimo y flexible [25]. En MEAN, Express funciona como un medio para transferir las solicitudes de un cliente a una base de datos y envía las respuestas de la base de datos al cliente [21].

3.3 AngularJS

AngularJS es una librería escrita en JavaScript para el desarrollo de aplicaciones web, mantenida por Google, es un framework JavaScript de código abierto y aborda los retos de las single-page aplications (SPAs) [26]. Una aplicación web AngularJS sigue el patrón de diseño MVC, que resulta en el desarrollo de aplicaciones web ampliables, mantenibles, comprobables y estandarizadas [26]. La unión de datos AngularJS y la inyección de dependencias lo convierten en un socio ideal para cualquier tecnología de servidor, ya que elimina gran parte del código que de otro modo tendrías que escribir, y todo sucede dentro del navegador [21].

3.4 NodeJS

NodeJS es un framework de desarrollo desarrollado originalmente en 2009 por Ryan Dahl, basado en el motor JavaScript V8 de Google [27]. Node es una plataforma basada en el tiempo de ejecución JavaScript de Chrome para crear aplicaciones en red rápidas y escalables [20]. Node utiliza un modelo controlado por eventos y de no bloqueo de E/S que lo hace ligero y eficiente, perfecto para aplicaciones que requieren grandes cantidades de datos y que se ejecutan en dispositivos distribuidos [21]. Node es un lenguaje de scripting del lado del servidor que puede ser usado en el lado del servidor, lado del cliente, e incluso puede ser un servidor web. Antes de que existiera el Node, JavaScript se usaba simplemente para la interacción del usuario como script del lado del cliente [28].

3.5 Arquitectura MEAN

Al desarrollar una aplicación web la elección de la mejor arquitectura y que para el usuario sea una experiencia fácil, agradable y funcional es trascendental. MEAN trabaja con una arquitectura MVC model-view-controller, que comprende tres capas esenciales: datos, lógica y vista. Una arquitectura MVC funciona así [21]:

- Una solicitud entra en la aplicación.
- La petición se enruta a un controlador.
- Si es necesario, el regulador realiza una petición al modelo.
- El modelo responde al controlador.
- El controlador envía una respuesta a una vista.
- La vista envía una respuesta al solicitante original.

En la arquitectura MVC, la lógica, los datos y la presentación se separan tres tipos de objetos, cada uno se encarga de sus propias tareas. La vista maneja la parte visual, que trata de la interacción con el cliente. El controlador responde a las peticiones del sistema y del cliente, haciendo que el modelo y la vista cambien apropiadamente. El modelo maneja la información, respondiendo a las demandas de datos o cambiando su estado de acuerdo a las instrucciones enviadas desde el controlador [20].

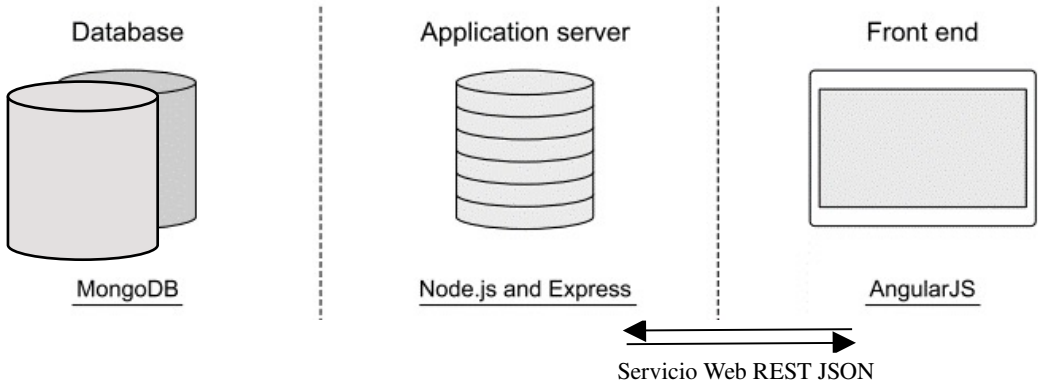

Figura 1. Arquitectura MEAN [21].

La figura 1 muestra la arquitectura de aplicación típica de una pila MEAN. AngularJS como Front-End con un MVC, que se comunica con el servidor Node a través de Express. Por cada solicitud de datos del Back-End, Node envía la solicitud a través del controlador nativo MongoDB o Mongoose. La

respuesta del servidor es enviada al cliente a desde Express. El modelo de datos que se utilizado es JSON y siendo RESTful la estructura de servicio para conectar las rutas desde Express, permitiendo la transferencia de petición y respuesta de ida y vuelta entre el cliente y el servidor [20] .

3.6 MEAN: Principales características

MEAN gana popularidad debido a las características de sus componentes: NodeJS, MongoDB, AngularJS, junto con Express y todo utilizando JavaScript que es una de sus mayores ventajas. Aunque podemos afirmar que MEAN no se recomienda en sistemas pesados de procesamiento computacional, debido a que el componente NodeJS es un entorno de un solo hilo y esto sería un inconveniente fuerte de esta tecnología. En una aplicación de una página (SPA), que requiere una alta interacción con el usuario, visualización y alta escalabilidad, MEAN es la mejor opción. La elección de la herramienta depende principalmente de algunas características que se detalla en la siguiente tabla [6]:

Característica	MEAN
Capa de servidor simplificada	Construcción de servidor localmente
Código isomórfico	Código puede ser ejecutado en el front-end como el back-end
Escalabilidad	La aplicación puede expandirse los eventos se ejecutan libremente de manera asíncrona
Arquitectura sin bloqueo	Maneja el uso de bucles de eventos sin bloqueo en un solo hilo.
Tiempo de desarrollo	Al utilizar el mismo lenguaje JavaScript se reduce el esfuerzo y tiempo de desarrollo
Transformación de datos y extensibilidad	Los datos manejados para el intercambio de información es JSON.

Tabla 1. Resumen de características de las stacks.

4 Caso de Estudio: App web desarrollada

En esta sección se procederá se desarrolló la aplicación de la hoja de vida y gestión de docente para el departamento de talento humano de la Pontificia Universidad Católica del Ecuador, Sede Esmeraldas, que fue construido utilizando la tecnología MEAN. La aplicación que se desarrolló es un CRUD, con varios datos del docente para llenar su hoja de vida por ejemplo datos personales, títulos, experiencia docente, experiencia no docente, capacitaciones, investigaciones, artículos, congresos, libros, etc. Junto con un informe de actividades de fin de cada semestre con la información de alumnos tutorados, cursos, seminarios, congresos y proyectos de vinculación realizados. El desarrollo de esta aplicación es constatar la facilidad o dificultad de construir software web con esta tecnología. En la Figura 2, se puede observar el front-end de la aplicación

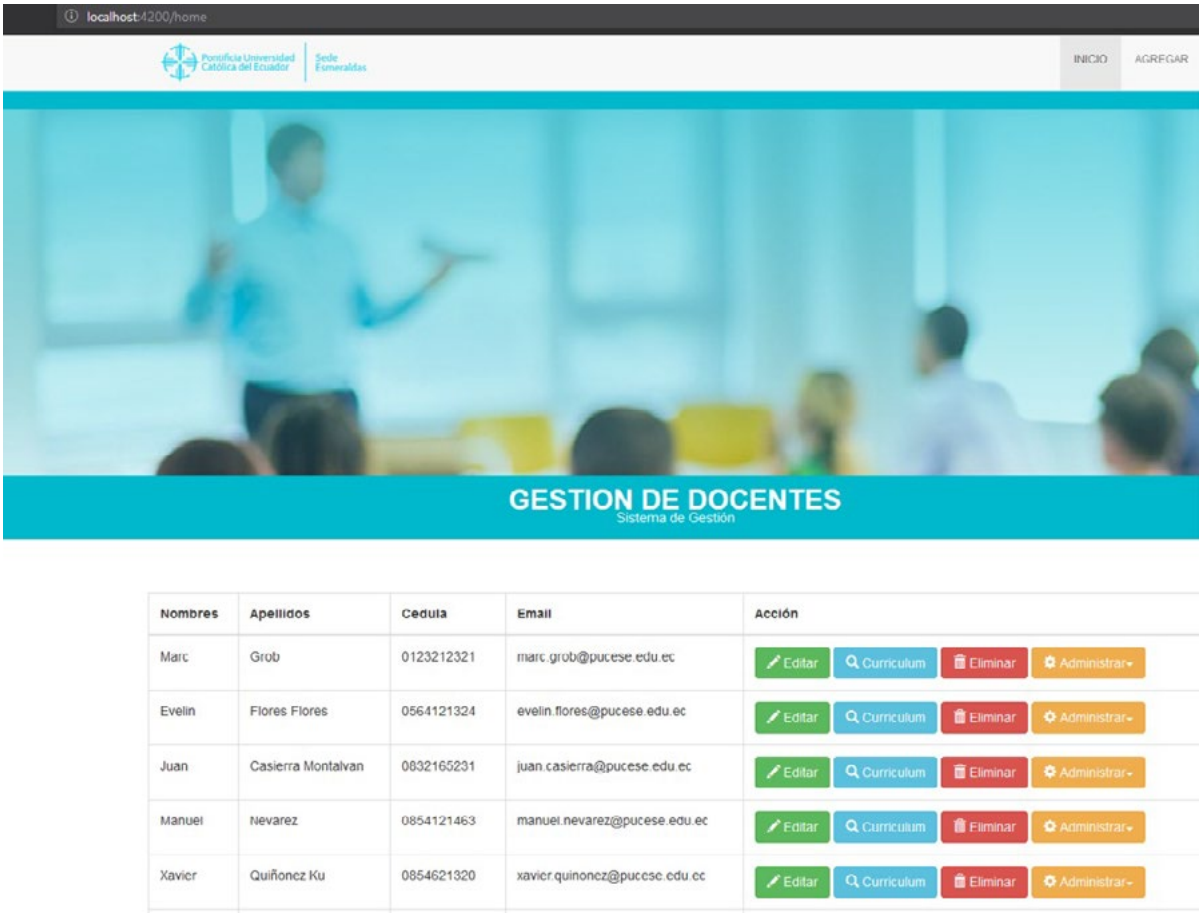

Figura 2. Captura de pantalla de sistema stack MEAN.

La tecnología utilizada para su implementación proporciona una rápida y necesaria flexibilidad y velocidad de las interacciones. También permitió incluir más características o funciones en la aplicación web y lo hizo más interactivo e intuitive. Esto ha sido posible gracias a la tecnología MEAN utilizada.

4.1 Resultados de la base de datos

La plataforma DbSchema permite exportar el modelo de las bases de datos MongoDB. La característica principal es obtener los resultados de la base de datos y presentarlos de forma agradable en un formato legible para el ser humano. El resultado de las distintas colecciones de la aplicación web en la Figura 3, muestra que los datos se almacenan utilizando bloques de JSON y BSON como formato de intercambio de JavaScript. A diferencia de las bases de datos relacionales, las tablas se llaman colecciones mientras que las filas se llaman documentos u objetos. Cada documento comienza con un _id y presenta los datos en forma de “nombre”: “valor”. Los valores en sí mismos pueden ser matrices de otros objetos anidados. MongoDB no obliga a actualizar el esquema como en RDBMS, lo que a veces puede resultar difícil. El resultado muestra que MongoDB nos permite almacenar objetos de una manera muy rica y dinámica. Esto hace que la presentación y la comprensión de las consultas a la base de datos sea muy fácil y la información requerida por el usuario.

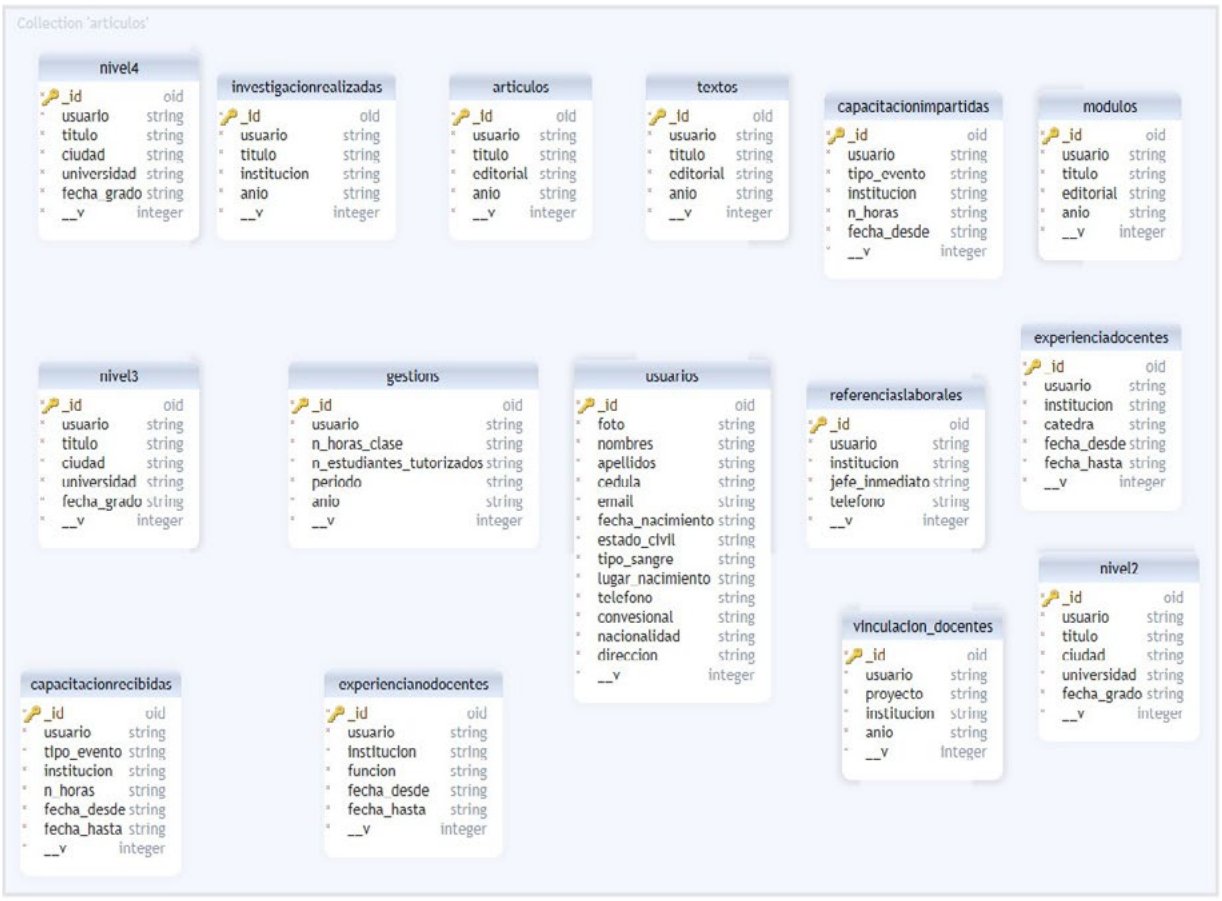

Figura 3. Colecciones de la base de datos MongoDB

4.2 Resultados de la materia desarrollo basado en plataformas

Los estudiantes de la materia desarrollo basado en plataformas tuvieron una impresión positiva hacia la tecnología MEAN, misma que se está impartiendo en la Pontificia Universidad Católica del Ecuador, Sede Esmeraldas, en su carrera de Ingeniería de Tecnologías de la Información. Además, los alumnos han tenido la oportunidad de construir un proyecto piloto e implementarlo en JavaEE y en MEAN. Los resultados de la introducción MEAN mostraron que, a pesar de tener menos bibliografía y lecciones aprendidas, construyeron el proyecto más rápido y con mejor interfaz en una moderna tecnología como es MEAN, en comparación con la pila de la tecnología JavaEE.

5. Conclusiones

En el estado del arte se hizo una descripción de las tecnologías para arquitecturas y servicios web, se describió y analizó las tecnologías para el desarrollo de aplicaciones web. MEAN es un paquete de software que combina MongoDB como la base de datos NoSQL, Express como un framework de NodeJS para el scripting para el desarrollo del back-end, Angular como plataforma MVC para la construcción del front-end, construido con código JavaScript. La creciente popularidad del uso de JavaScript como secuencias de comandos del front-end y del back-end ha hecho que MEAN sea una las combinaciones de tecnologías más utilizadas para desarrollo aplicaciones web. MEAN está construida exclusivamente en JavaScript, por lo

que es un lenguaje para gestionar el lado del cliente, servidor y base de datos. Todos los componentes utilizados en la stack MEAN son completamente de código abierto y actualmente son soportados por desarrolladores corporativos como MongoDB y Google. Todos los componentes de MEAN son relativamente ligeros. La stack MEAN es flexible y escalable. Angular es una muy buena opción para el desarrollo de una SPA y es responsiva. La buena impresión que ha dejado MEAN, se está impartiendo en la Pontificia Universidad Católica del Ecuador, Sede Esmeraldas, en su carrera de Ingeniería de Tecnologías de la Información. El análisis ha demostrado que MEAN es hoy la mejor combinación para unir el back-end de la aplicación con respecto al front-end realizado en tecnología Angular, teniendo en cuenta el rendimiento y la velocidad de la aplicación, la comunicación entre el cliente y el servidor. Un análisis de las deficiencias de MEAN nos proporcionaría con una mejor comprensión y su aplicabilidad. Sin embargo, este documento demostró como MEAN es óptima para el desarrollo de aplicaciones web modernas, hay limitaciones en nuestro estudio de caso. Por ejemplo, nuestro estudio de caso no muestra cómo MEAN reacciona con las aplicaciones intensivas y de gran procesamiento en CPU y memoria RAM. Por lo tanto, este documento podría beneficiarse de la introducción de un estudio de caso que se ocupe del uso intensivo y gran procesamiento de la CPU y memoria RAM.

References

[1] M. Stajcer and D. Orescanin, "Using MEAN stack for development of GUI in real-time big data architecture," *2016 39th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2016 - Proc.*, pp. 524–529, 2016.

[2] A. J. Poulter, S. J. Johnston, and S. J. Cox, "Using the MEAN stack to implement a RESTful service for an Internet of Things application," *IEEE World Forum Internet Things,* WF-IoT 2015 - Proc., pp. 280–285, 2015.

[3] R. Salunkhe, S. Telang, P. Shrigondekar, and A. Tanpure, *Review of REST Ful Service Using MEAN Stack for Real Time Big Data Architecture*, vol. 3297, no. 11. Birmingham: Packt Publ, 2007.

[4] M. J. Collins, *Pro HTML5 with CSS, JavaScript, and Multimedia*. 2017.

[5] S. Holmes, "Introducing full-stack development," *Get. MEAN*, pp. 3–23, 2015.

[6] A. Mardan, Full stack javascript: *Learn Backbone.js, Node.js, and MongoDB*. 2018.

[7] M. Hajian and N. Oslo, "Progressive Web Apps with Angular Create Responsive, Fast and Reliable PWAs Using Angular-Majid Hajian," pp. 1–380, 2019.

[8] N. Rozanski and E. Woods, Software Systems Architecture: *Working with Stakeholders Using Viewpoints and Perspectives*. Addison-Wesley, 2005.

[9] D. Garlan and M. Shaw, "An Introduction to Software Architecture," *Knowl. Creat. Diffus. Util.*, vol. 1, no. January, pp. 1–40, 1994.

[10] S. Engineering and S. Committee, "IEEE Recommended Practice for Architectural Description of Software-Intensive Systems," 2000.

[11] D. Garlan and D. Garlan, "Software Architecture : a Roadmap Software Architecture : a Roadmap," 2000.

[12] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, Second Edi. Addison Wesley, 2003.

[13] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture - Volume 1: A System of Patterns*. Wiley Publishing, 1996.

[14] R. T. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," *Building*, vol. 54, p. 162, 2000.

[15] M. Zur Muehlen, J. Nickerson, and K. Swenson, "Developing web services choreography standards - The case of REST vs. SOAP," *Decis. Support Syst.*, vol. 40, no. 1 SPEC. ISS., pp. 9–29, 2005.

[16] D. Fensel, F. M. Facca, E. Simperl, and I. Toma, *Semantic Web Services*, 1st ed. Springer Publishing Company, Incorporated, 2011.

[17] C. Pautasso, O. Zimmermann, and F. Leymann, "Restful web services vs. 'big'web services: making the right architectural decision," *Proceeding 17th Int. Conf. World Wide Web*, pp. 805–814, 2008.

[18] S. Patni, *Pro RESTful APIs*. 2017.

[19] D. Flanagan, *JavaScript - The Definitive Guide*. 2011.

[20] A. Q. Haviv, *MEAN Web Development*, vol. 1. 2014.

[21] S. Holmes, *Getting MEAN with Mongo, Express, Angular, and Node*, 1st ed. Greenwich, CT, USA: Manning Publications Co., 2015.

[22] K. Chodorow, *Mongo DB: The Definitive Guide*. 2013.

[23] R. O. Obe and L. S. Hsu, *MongoDB in Action*. 2011.

[24] E. Brown, *Web Development with Node and Express: Leveraging the JavaScript Stack*. O'Reilly Media, 2014.

[25] E. Hahn, Express in Action: *Node Applications with Express and Its Companion Tools*, 1st ed. Greenwich, CT, USA: Manning Publications Co., 2015.

[26] R. K. Soni, *Full Stack AngularJS for Java Developers*. 2017.

[27] S. Davis, "Mastering MEAN: Introducing the MEAN stack," *IBM.com,* pp. 1–20, 2014.

[28] M. Cantelon, M. Harter, T. J. Holowaychuk, and N. Rajlich, *Node.Js in Action*, 1st ed. Greenwich, CT, USA: Manning Publications Co., 2013.

Recibido: *10 de enero de 2022*
Aprobado: *31 de enero 2022*