GotenJS: Framework para generar la estructura base de una aplicación web

Cintia Tatiana Capece[1], Reyna Der Boghosian[1], Ciro Edgardo Romero[1], Pablo Hazan[1], and Jon Colque[1]

C&S Informática S.A. Departamento de Investigación, Desarrollo e Innovación
{ccapece, rderboghosian, cromero, phazan, jcolque} @cys.com.ar

Abstract. Dentro del mercado de software, orientar el desarrollo en la resolución de problemáticas es importante para ser competitivo. Enforcar los esfuerzos en el análisis de problemáticas es complejo cuando no se cuenta con una arquitectura pre definida asi como generar una estructura de un proyecto desde cero, puede ser una tarea que ocupe tiempo no necesariamente productivo. Desde esta premisa, el departamento de Investigación, Desarrollo e Innovación de C&S analiza el desarrollo de una herramienta propia que permita crear una estructura base, orientada a proyectos de software web.

Keywords: Javascript · CLI · Fullstack · MEAN stack · MERN stack · ágil.

1 Introducción

Hoy en día posicionarse competitivamente en el mercado, cubrir una necesidad existente o descubrir en los usuarios una nueva necesidad que no conocían, hace que ser pionero en el lanzamiento de una aplicación sea algo relevante en un mercado competitivo.La creación de aplicaciones para internet ocupa una parte sustancial del desarrollo de software en la actualidad, porque requiere continuamente de herramientas que faciliten, agilicen y eleven la calidad del proceso de desarrollo web. [1]
Es en este marco donde C&S [2], una empresa de tecnología, brinda soluciones de ingeniería de software a organizaciones que buscan aumentar su productividad y eficiencia. Desde el área de I+D+i (investigación, desarrollo e innovación de software) en un contexto de innovación, surge el querer facilitar el desarrollo de aplicaciones web mediante la creación de un framework Javascript. El mismo recibe el nombre de GotenJS y se orienta en el marco de desarrollo de aplicaciones web.
Este artículo se divide en cuatro secciones, la primera sección trata sobre el contexto en el cuál surge GotenJS . La segunda sección introduce los conceptos básicos para definir el marco teórico de GotenJS. La tercera sección describe la propuesta de C&S y su arquitectura . Y en la cuarta sección se describen las conclusiones del presente artículo.

2 Conceptos de base

Se define un marco teórico para ubicar conceptos relevantes a los cuales se hace referencia a lo largo de todo el presente artículo. Describiendo los mismos de forma escalar, desde conceptos generales hasta los más específicos.

2.1 Javascript y el concepto de stack

Javascript es un lenguaje bastante sencillo y rápido, permite la ejecución de miles de líneas de código en cuestión de segundos. Además, es soportado por la mayoría de los navegadores actualmente. [3]

Se define un "stack" como cualquier combinación de lenguajes de programación y tecnologías, o una combinación de productos de software. Además, el término Full Stack va ligado tanto al dominio, como al conocimiento y al interés de los distintos stacks, que tiene cualquier modelo de software usando tecnologías que posee a su alcance. [1]

2.2 MEAN Stack

Se define el término "MEAN Stack" como un conjunto de tecnologías basadas en Javascript que se utilizan para desarrollar los sitios web complejos y las aplicaciones web (progresivas o receptivas). En otras palabras, es un framework de Javascript de full-stack, que simplifica y acelera el desarrollo de las apps y web debido a que ayuda a los desarrolladores y equipos a través de herramientas que reducen el tiempo de administración del sistema y permiten su despliegue de manera rápida. Es decir, MEAN es una combinación entre MongoDB, Express, Angular.js y Node.js, que conjugados ofrecen un kit de desarrollo completo y extremadamente útil. Una de las principales ventajas de MEAN es que emplea el mismo lenguaje de programación en todas las partes de la aplicación lo que permite que una persona pueda manejarse en todos los ámbitos de una aplicación web moderna aunque se especialice en uno de ellos. [1]

2.3 MERN Stack

Dentro de las combinaciones de tecnologías JS utilizadas para crear apps web existe MERN Stack que consta de diferentes componentes de código abierto: MongoDB (una base de datos de código abierto basada en documentos), Express (un marco web rápido, no motivado y minimalista para Node.js), React (una biblioteca de front-end de Javascript para crear interfaces de usuario) y Node.js (es un tiempo de ejecución de Javascript creado en el motor de Javascript V8 de Chrome. Node.js trae Javascript al servidor).

MERN Stack es muy similar a MEAN Stack. La única diferencia es que MEAN Stack está haciendo uso de Angular para crear la app web front-end, pero en

MERN Stack se está utilizando React en su lugar.

Hoy en día, los stack MEAN / MERN son adoptados para el desarrollo de aplicaciones web por la eficiencia y sencillez de la plataforma. [5]

3 Propuesta Creativa: GotenJS

Elegir la infraestructura adecuada para cada negocio no es una decisión fácil. La tecnología está en constante cambio, la cual hace que al intentar mejorar la infraestructura se presenten desafíos. Algunos proyectos de C&S están encontrando oportunidades para poder desarrollar partes de los sistemas en nuevas tecnologías. Una de las alternativas es el stack de Javascript. Si bien en C&S se han realizado proyectos con estas herramientas, aún no hay un criterio unificado sobre una arquitectura de referencia y guías de desarrollo que permitan encarar un nuevo proyecto con una estructura predefinida. Para ello, desde el área de I+D+i se decidió crear un proyecto: GotenJS, el cuál define una arquitectura base de manera tal que el proceso de construcción de aplicaciones sea estandarizada, habiendo definido ciertas estrategias de diseño y desarrollo que se detallan en este artículo.

GotenJS es un framework que se estructura en módulos que interactuan dependiendo de la tecnología a utilizar como se puede visualizar en la Fig.1.

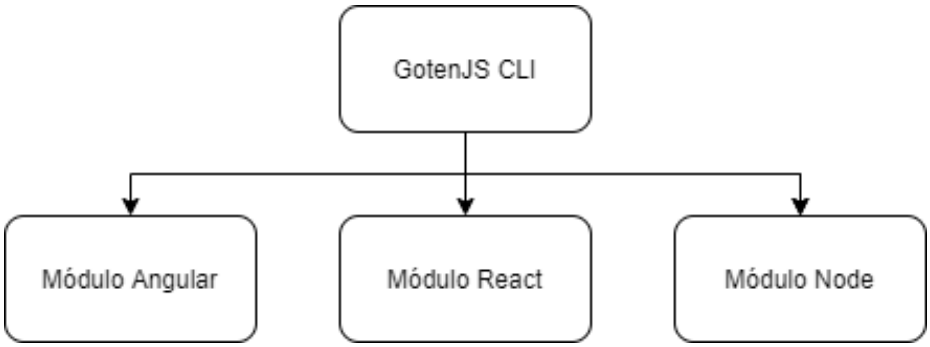

Fig. 1. Estructura modular de GotenJS

Como primera fase se desarrolló para tecnologías como lo son Angular, React y NodeJS. A continuación se describen los módulos de las tecnologías asociadas.

3.1 GotenJS CLI

El CLI (Command-line Interface) de GotenJS comenzó como el primer objetivo del framework. Este consisten en generalizar código de una aplicación base, cuyo

proposito es la creación de proyectos. Esto implicó un trabajo del área de I+D+i para que en los proyectos futuros pueda ser usado por toda la empresa C&S; en los proyectos que lo requieran.

Actualmente, se tiene una interfaz de consola que genera proyectos con una estructura predefinida. Estos proyectos son aplicaciones web conectadas a una base de datos (MySQL, MariaDB o MongoDB), un backend (en NodeJS, con Express), y un frontend (React, Angular).

Cada comando realiza una acción coherente con el nombre del propio comando y la tecnología especificada cómo parámetro. Se detallan a continuación las acciones que soporta el módulo GotenCLI:

Creación de Proyectos Base GotenJS provee el comando **goten new** para poder crear proyectos base dependiendo de la tecnología especificada.

Alta-Baja-Modificación El concepto de "Alta, Baja y Modificación", puede ser referenciado por las sigas ABM, obviando la operación de "obtener" siendo utilizadas por las operaciones CRUD (crear, leer, actualizar y borrar). De esta forma, se lleva a cabo la persistencia en las bases de datos. Para esto, GotenJS provee el comando **goten abm**.

Autenticación GotenJS provee el comando **goten auth** para agregar la autenticación con JWT y permisos basados en roles/atributos del lado del backend y del lado del frontend la pantalla correspondiente al Login para ya ingresar el username y el password generados desde el backend.

Ícono de página GotenJS provee el comando **goten favicon** para asignar un ícono en el browser de la Single Application (SPA) generada en React y/o Angular.

3.2 Módulo Angular

El módulo correspondiente a Angular trae las posibilidades de crear una aplicación con la estructura base del proyecto conteniendo lo generado por el comando ng new [9], e instala las dependencias de Bootstrap4 [10] y fontawesome [11].

En la Figura 2 se muestra la arquitectura basada en Angular y como GotenJS interactúa con sus componentes.

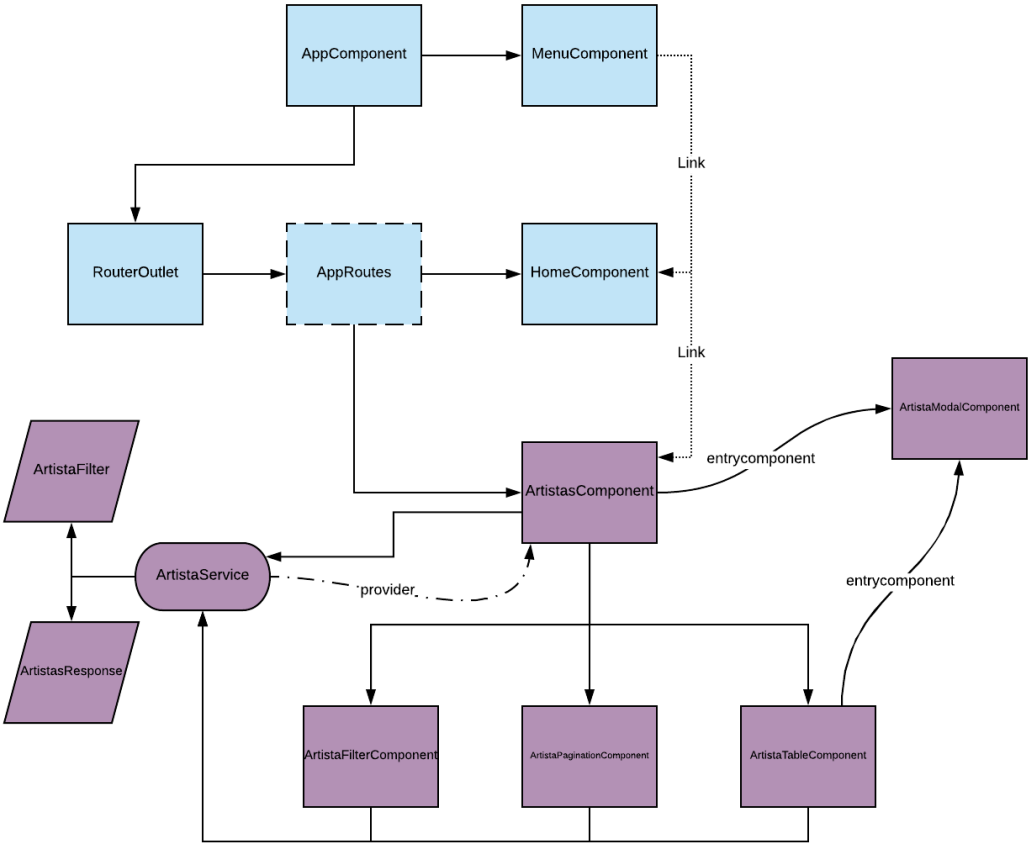

Fig. 2. Estructura del proyecto de Angular y la relación entre componentes

Cuando se ejecuta el comando *goten new-angular* se genera la estructura base del proyecto de Angular compuesto por directorios y archivos de base según la siguiente descripción:

- **proyecto/src/app/components/:** Directorio donde se agregan los componentes autogenerados.
- **proyecto/src/app/components/home.component/*:** Componente home, pantalla inicial de la aplicación.

- **proyecto/src/app/components/menu.component/*:** Componente menu, donde se registran los distintos tabs y a donde se enlazan los mismos.
- **proyecto/src/app/components/generics/*:** Componentes base que utilizan los componentes autogenerados.
- **proyecto/src/app/dtos/:** Directorio donde se agregarán los datos de las entidades que se consuman de la API.
- **proyecto/src/app/dtos/filters/:** Directorio donde se agregarán los filtros que se utilizan en los forms.
- **proyecto/src/app/services/:** Directorio en la que se ubican los servicios autogenerados.
- **proyecto/src/app/app.routes.ts:** En este archivo se registran las rutas y los componentes de referencia.

3.3 Módulo React

El módulo correspondiente a React facilita la posibilidad de generar la arquitectura del proyecto tanto con **Context** [6] como con **Redux** [7].

Context La estructura del proyecto creado con **Context** se distribuye entre los directorios config , layout (componentes reutilizables), modules (componentes que representan los datos del backend) e utils (funciones y constantes útiles). La aplicación se crea utilizando **create-react-app** y en la Figura 3 se muestra lo que genera el CLI de GotenJS.

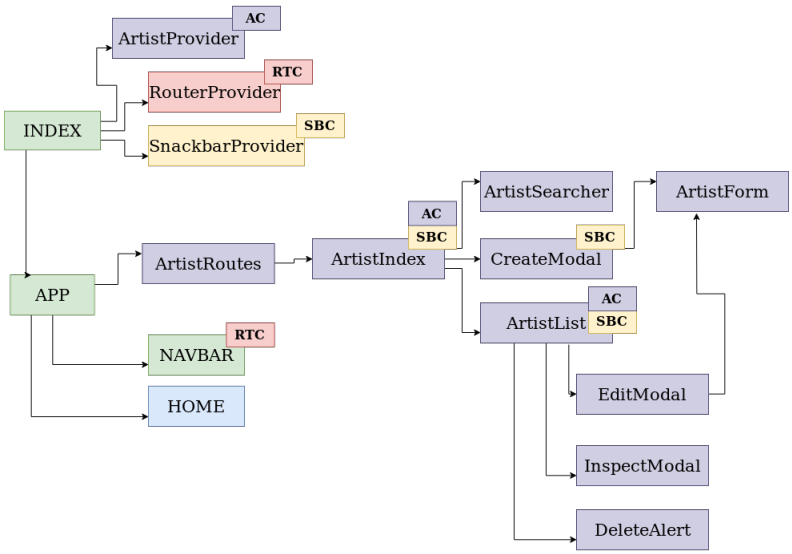

Fig. 3. Estructura del proyecto de Contex y la relación entre componentes.

En **verde**, estan los componentes que no hace falta que el usuario modifique. El componente de App y el Navbar son bastante estáticos, dado que el CLI se ocupa de generar las rutas al crear módulos, y de agregar los providers correspondientes a index.js.

En **azul** está el componente Home. Este tiene que ser modificado por quién sea que use el CLI, dado que no tiene configuración alguna y está hecho puramente como relleno (que se supone necesario para todo proyecto).

En **rojo** está representado el provider del contexto de Router(RTC).

En **amarillo** está representado el provider del contexto de Snackbar(SBC). En **violeta** está representado un modelo sobre el cuál se hizo el ejemplo de lo que generaría el CLI. Este módulo es de Artistas, y tiene ciertos componentes que representan el modelo definido en la base de datos.

Redux Luego, para el uso de **Redux** se define una estructura **'duck'**. Esta permite escalar la aplicación de forma incremental, de modo de no tener problemas con archivos de gran tamaño. La estructura a grandes rasgos se compone por una carpeta duck para cada módulo, independientes unas de otras y representarán cierta funcionalidad, y en estas se encontrará lo típico de Redux (constantes de tipos, acciones y action creators, y un concepto nuevo que se llama operations). La estructura es similar a la creada con Context.Está dividida en los modulos **config**, **layout** (componentes reutilizables), **modules** , **utils** (funciones y constantes útiles, entre ellas) y **redux**, En la Figura 4. se muestra lo generado por CLI de GotenJS.

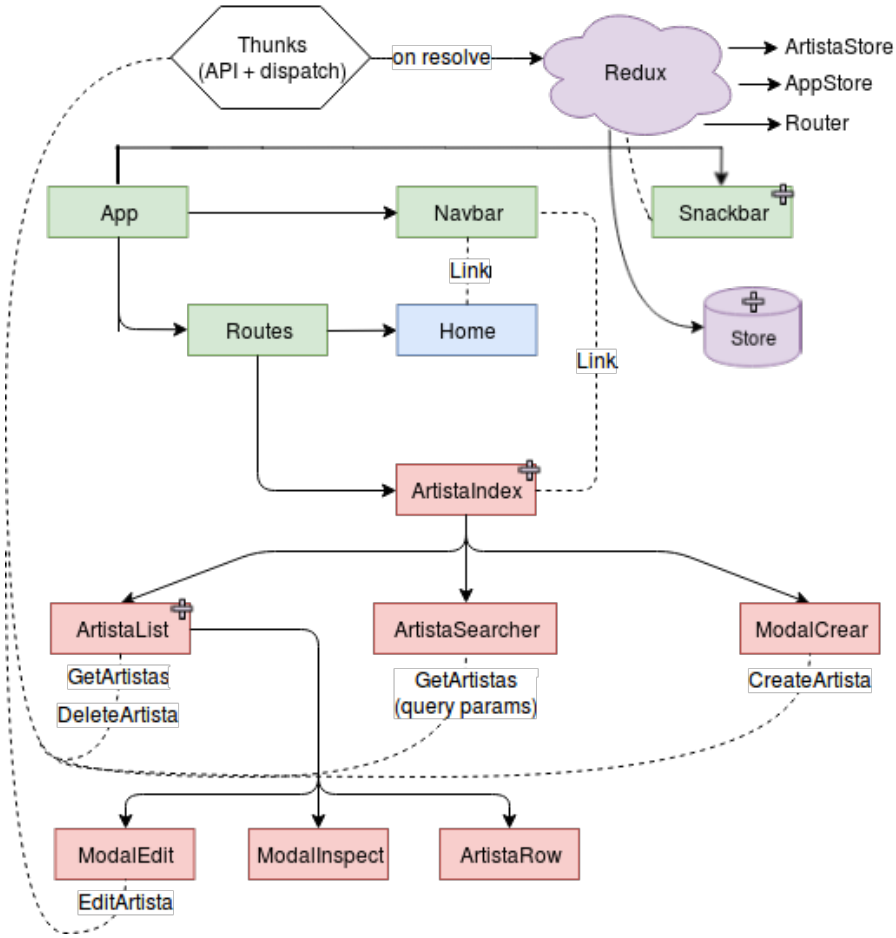

Fig. 4. Estructura del proyecto de Redux y la relación entre componentes.

En **verde**, los componentes que no serán modificados (o no necesariamente) por el usuario. En **azul**, el componente Home.

En **violeta** está representado Redux y su store. [8] Algunos componentes escuchan al store de Redux (se los marca con una pequeña cruz violeta arriba a la derecha) mientras que otros estarán haciendo dispatch para hacer pedidos a la API, o en el caso del snackbar para mostrar y cambiar su mensaje.

En **rojo** se representa el modelo generado por CLI. Este módulo es de Artistas, y tiene ciertos componentes que representan el modelo.

3.4 Módulo NodeJS

E l módulo Node permite generar la estructura base del proyecto dividiéndola en las carpetas config (configuraciones pertinentes), routes (definición de las

rutas correspondientes), controllers(conjunto de funcionalidades), services , dao (interacción directa con la base de datos), models (los modelos que se representan en la base de datos), dtos (recurso para asignar los atributos del modelo), assemblers (funciones utilizadas para crear los modelos), filters(creación de filtros para realizar búsquedas en la base de datos); tal y como se puede visualizar en la Figura 5.

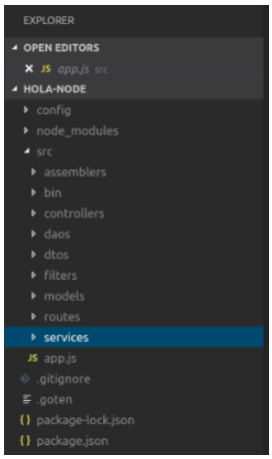

Fig. 5. Estructura del directorio del proyecto de NodeJS.

4 Conclusiones

Actualmente GotenJS está en desarrollo para alcanzar la madurez suficiente con el objetivo de formar parte del estándar de C&S. El propósito es aportar a la empresa mejoras en su método de desarrollo a través de la arquitectura y las estrategias de diseño que se generan como resultado de la utilización de la herramienta descrita en el presente artículo. Como trabajos a futuro, se quiere dar la robustez necesaria para poder lanzar una versión estable de uso público.

En algunos proyectos de desarrollo de C&S se encontró la oportunidad de desarrollar parte de sus sistemas utilizando el framework GotenJS. De esta manera, el equipo de I+D+i logró sus primeras pruebas de concepto en un proyecto real dentro de la compañía para así continuar evolucionando la herramienta.

References

1. Gonzalez Labarta, Rubiel. (2014). JAVASCRIPT SEAN STACK: UNA MIRADA AL FUTURO DEL DESARROLLO WEB.
2. C&S , `https://cys.com.ar/es/home/`

3. Germain Ramírez, Cristal Esmeralda. (2020) “Desarrollo de aplicaciones web utilizando JavaScript.”
4. Yebra Acero, Daniel. (2017) ”Desarrollo multiplataforma de tipo “Full Stack”: creación de un Back ”
5. Khue, Trinh Duy, Nguyen, Thanh Binh, Jang, UkJIn, Kim, Chanbin, Chung, Sun-Tae. (2017). Design and Implementation of MEARN Stack-based Real-time Digital Signage System. , 20(5), 808–826. https://doi.org/10.9717/KMMS.2017.20.5.808
6. React Context, `https://es.reactjs.org/docs/context.html`
7. React Redux, `https://react-redux.js.org/`
8. Store Redux, `https://redux.js.org/api/store`
9. Ng new Angular, `https://angular.io/cli/new`
10. Bootstrap, `https://getbootstrap.com/`
11. Fontawesome, `https://fontawesome.com/`