Análisis y desarrollo de una herramienta tipo Marketplace utilizando técnicas de aprendizaje automático bajo algoritmos no supervisados

Trabajo de grado para optar por el título de ingeniero informático

Fernando Guerreo Campo

Asesor
Guillermo Lince Bonilla Mariota
Ingeniero de Sistemas

Corporación Universitaria Lasallista.
Facultad de Ingenierías
Ingeniería Informática
Caldas-Antioquia
2018

Contenido

Lista de tablas

Lista de ilustraciones

Introducción

Este documento tiene como objetico describir los procesos dentro de la compañía GRANDPA según mis conocimientos adquiridos como integrante del equipo de desarrollo en el periodo bajo el perfil de practicante.

GRANDPA es una empresa dedicada al desarrollo de software a la medida. En la actualidad, cuenta con un producto propio enfocado en la venta de servicios mediante una plataforma de venta de cupones, agregando valor al producto a través de la integración de una de las mejores pasarelas de paga a nivel mundial, como lo es *Stripe,* la cual permite a las empresas crear por su propia cuenta, los cupones o campañas sin necesidad de intermediarios. Otra de sus funciones es enfocarse en la atención al cliente con la ayuda de un software desarrollado con algoritmos no supervisados conocidos como *Machine Learning,* el cual se especializa en hacer booking de hoteles.

En el documento se encuentra de manera detallada la metodología *SCRUM* a través de la cual se abordan los procesos de desarrollo de *Raliz* y del *ChatBot.*

Justificación

En la actualidad, las plataformas *marketplaces* para la venta de servicios mediante cupones no cuenta con un canal dedicado a las empresas que les permita crear sus propios con un diseño igual al que verían los usuarios potencialmente compradores y en el cual la empresa esté al pendiente de cuanto está activo y cuando se ha comenzado a vender el cupón, además cuentan con un alto porcentaje en el descuento de cada de cada servicio.

Es por ese motivo que GRANDPA desarrollo *Raliz*, una plataforma en la cual las organizaciones estén al mando de sus servicios y sean ellas las que decidan las reglas de unos para el mismo.

Además, GRANDPA, se encuentra innovando en el campo de atención al cliente para el tema de *booking* de hoteles, gracias a la creación de un *ChatBot* desarrollado bajo algoritmos no supervisados, donde la inteligencia artificial es fundamental a la hora de atender al cliente; esto permite que las personas se comuniquen rápidamente con el sistema de *booking*, reciban respuestas inmediatas y experimente un soporte ágil y rápido. Con la integración de algoritmos nos supervisados, se minimiza la espera y el tiempo que pueden tomar los asesores en brindar apoyo.

Objetivos

Objetivo general

Optimizar los sistemas de búsqueda para las herramientas de software de gestión hotelera y cupones mediante el desarrollo de algoritmos no supervisados.

Objetivos específicos

- Conocer los diferentes procesos de desarrollo de software de la empresa GRANDPA.
- Dominar la arquitectura base de las herramientas *ChatBot* y *Raliz*
- Aplicar *SCRUM* como mitología ágil para garantizar entregas interactivas e incrementales.
- Aplicar los conocimientos adquiridos en *JavaScript, ReactJs, ApolloJs, GraphQL* y *NodeJs.*

Marco teórico

Microsoft Visual Studio Code

Es un editor de código fuente ligero y potente que se ejecuta en el escritorio, viene con soporte integrado para *JavaScript, TypeScript y NojdeJs* y tiene un rico ecosistema de extensiones para los lenguajes (*C++, C#, Java, Python, PHP Go*) y tiempo de ejecución (*.NET, y Unity*). (Microsoft, 2018)

- **Capa de presentación:** Presenta el sistema al usuario, comunica la información y a su vez la captura a través de un navegador Web (paginas HTML, CSS) y una dirección web URL asignada al sistema.
- **Capa de negocio:** Recibe las peticiones del usuario y envía las respuestas tras el proceso. También se alojan las páginas estáticas y dinámicas para las diferentes aplicaciones.
- **Capa de datos:** Es el lugar donde residen los datos. Esta capa está encargada de acceder a los mismos; formada por uno o más gestores de bases de datos que no solo realizan todo el almacenamiento, sino que además reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Los dos sistemas están siendo desarrollados en lenguaje *JavaScript* y en la implementación se utilizan las siguientes herramientas:

Tabla 1. Capa de presentación (Cliente)

Propósito	Producto
Navegador	Google Chrome, Internet 10 o superior, Microsoft Edge, Firefox
Librerías	React.js versión 16, CSS3, HTML5.

Tabla 2. Capa de Negocio.

Propósito	Producto
Lenguaje Utilizado	JavaScript
Aplicaciones	Node.js, Apollo.js, Express.js, npm, Babel.js, GraphQL, Passport.js, JWT.
Herramientas	Microsoft Visual Studio Code.

Tabla 3. Capa de Datos

Propósito	Producto
Base de Datos	MongoDB, ElasticSearch.

MongoDB

Es una base de datos *NoSQL* (no relacional), que permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan. Mongo es una base de datos orientada a documentos. Esto quiere decir que, en lugar de guardar datos en registros, los guarda en documentos. Dichos documentos son almacenados en *BJSON,* que son una representación binaria de *JSON*.

Mongoose.

> Es el puente de conexión entre el servidor Node.js con la base de datos Mongo, además proporciona una solución directa basada en esquemas para modelar los datos de la aplicación. Incluye casting de tipo incorporado, validación, creación de consultas, entre otras muchas vas herramientas. (Mongoose, 2018, 44).

NodeJs

> Node.js es un entorno de desarrollo *JavaScript* del lado del servidor, basado en eventos. Es un entorno de ejecución para *JavaScript* construido con el motor de *JavaScript V8 de Chrome*. Node usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes Node.js, *npm*, es el ecosistema de librerías de código abierto más grande del mundo. (NodeJs, 2018, 67).

NPM

"Es un administrador de paquetes o módulos para *Node.js*. facilita a los desarrolladores compartir el código que han creado para resolver problemas particulares, y para que otros desarrolladores re-utilicen ese código en sus propias aplicaciones". (npm, 2018, 37)

Express.js

Es una infraestructura de aplicaciones web *Node.js* mínima y flexible que proporciona un conjunto solido de características para las aplicaciones web y móviles. Con miles de métodos de utilidad *HTTP* y *middleware* a disposición, la creación de una *API* es rápida y fácil. (Node.js, 2018, 43).

Apollo.js

Apollo es un *framework* tanto para cliente como para el servidor. *Apollo Client* es el cliente *GraphQL* ultra flexible y orientado para la comunidad *React, JavaScript* y plataformas nativas. *Apollo Server* conecta cualquier *Backend* con *GraphQL*, crea fácilmente APIs *GraphQL* que se conectan a uno o más *API REST*. (Meteor Develement Group Inc, 2018).

GraphQL

Es un lenguaje de consulta *"query"*, y un tiempo de ejecución del lado del servidor para ejecutar consultas mediante el uso de un sistema de tipo definido para los datos. *GraphQL* no está vinculado a ninguna base de

> datos o motor de almacenamiento y, en cambio, está respaldado por su código de datos existente. (Facebook Inc, 2018).

Next.js

"Es un *framework* de *JavaScript* para aplicaciones hechas en *React* para hacer *server-rendered*, controlando el intercambio de datos con el servidor, enviando actualizaciones de código". (Zeit Inc, 2018).

Uploadcare

Maneja la carga de archivos, el almacenamiento, el procesamiento y la entra de un sitio web o aplicación a través de componentes incluidos widget y CDN.

Redux.js

Redux se encarga en cierta manera de desacoplar el estado global de una aplicación web (en *Front-End*) de la parte visual, es decir los componentes.

Flow.js

> Es un verificador de tipo estático para código *JavaScript*, verifica el código en busca de errores a través de anotaciones de tipo estático. Estos tipos le permiten decirle a *Flow* cómo quiere que funciono el código, *Flow* se encargará de que funcione de esa manera. (Facebook Inc, 2018).

Webpack.js

Es un sistema de *building* para preparar el desarrollo de una aplicación web para producción. En cierta medida se puede considerar un *Browserify* avanzado ya que tiene muchas opciones de configuración.

Babel.js

“Babel es una herramienta que permite transformar código *JavaScript* de última generación (con funcionalidades extras) a código *JavaScript* que cualquier navegador o versión de *Node.js* entienda. Funciona con la ayuda de plugins que le indican que cosas se quieren transformar”. (Babel,js, 2018).

React.js

Es una librería *JavaScript*, desarrollada por *Facebook* para facilitar la creación de interfaces de usuario (UI), funcionando como intermediario entre los componentes interactivos y reutilizables. Promueve un flujo muy claro de datos e eventos, facilitando la planeación y desarrollo de aplicaciones complejas. (Facebook Inc, 2018).

JSON Web Token (JWT)

Es un estándar abierto (RFC 7519) que define una forma compacta y automática que tiene como objetivo transmitir información de un modo seguro entre las partes cliente-servidor como un objeto *JSON*. Esta información puede ser verificada y confiable porque está firmada digitalmente. Los *JWT* se pueden firmar usando un par de clases publicas/privadas usando *RSA*. (Auth0 Inc, 2018).

Lodash.js

Es una librería de *JavaScript* que facilita el trabajo con matrices, objetos y arreglos, a través de métodos modulares.

DialogFlow

> Es una plataforma de compresión del lenguaje desarrollada por *Google*, que permite a los desarrolladores e integrar interfaces de usuario conversacionales inteligentes y sofisticadas en aplicaciones móviles, aplicaciones web, dispositivos y *bots*. Una vez implementado, el *bot* continúa aprendiendo de las conversaciones con los usuarios gracias a *Machine Learning* y a los algoritmos no supervisados. (Google Inc, 2018)

Algoritmos no supervisados o Machine Learning

Los algoritmos no supervisados son una rama de la inteligencia artificial, la cual pretende crear algoritmos capaces de generalizar comportamientos y reconocer patrones a partir de una información suministrada.

Se considera algoritmo no supervisado a aquello que la maquina puede aprender de la experiencia y del reconocimiento de patrones suministrados.

Se distingue del aprendizaje supervisado por el hecho de que no hay un conocimiento a priori.

ElasticSearch

Es un servidor de búsqueda basado en *Lucene*. Provee un motor de búsqueda de texto completo, distribuido y con capacidad de multi-tendencia con una interfaz *RESTFul* y con documentos *JSON.*

JSON

Acrónimo de *JavaScript Object Notation*, es un formato de texto ligero para el intercambio de datos. Básicamente *JSON* describe los datos como una sintaxis dedicada que se usa para identificar y gestionar los datos. Una de las mayores ventajas que tiene el uso de *JSON* es que puede ser leído por cualquier lenguaje de programación, por lo tanto, puede ser usado para el intercambio de información entre distintas tecnologías. (JSON Org, 2018).

Metodología

Para el desarrollo de los proyectos anteriormente mencionados, se aplican metodologías ágiles de desarrollo, puntualmente se implementa una denominada *SCRUM.*

Metodología SCRUM

Scrum es un *framework* adaptable, interactivo, rápido, flexible y eficaz que está diseñado para entregar valor al cliente durante todo el desarrollo del proyecto. El objetivo primordial es satisfacer las necesidades del cliente a través de un entorno de transparencia en la comunicación, responsabilidad colectiva y progreso continuo.

Roles en SCRUM

Dentro de la metodóloga *SCRUM* se entienden dos categorías de roles:

Roles Centrales

“Los roles centrales son protagonistas debido a que su participación es indispensable para la realización del proyecto, están comprometidos, son responsables del éxito de cada *sprint* y del proyecto en general”. (Cortés, Pineda Yesica, Platzi, 2017)

Ilustración 1. Roles centrales

Fuente: Platzi

Cada uno de los estos roles tiene una función específica:

- **Product Owner (Dueño del Producto):** Es “la voz del cliente” y el responsable de desarrollar, mantener y priorizar las tareas del *backlog.*
- **Scrum Master:** Es el responsable de asegurarse de que el trabajo vaya bien siguiendo las bases del *Scrum*. Además, se encarga de remover cualquier obstáculo que pueda encontrar el equipo de desarrollo.
- **Development Team Meambers (Miembros del equipo de desarrollo):** Son los encargados de escribir y probar el código.

Roles no Centrales

Los roles no centrales son aquellos cuya participación en el proyecto es importante pero no depende de ellos el éxito o el fracaso del mismo. Siempre es importante identificar los individuos de esta categoría y mantenerlos siempre presentes, en cualquier momento su rol puede ser decisivo para el proyecto (por ejemplo, su *sponsor*). (Cortés, Pineda Yesica, Platzi, 2017).

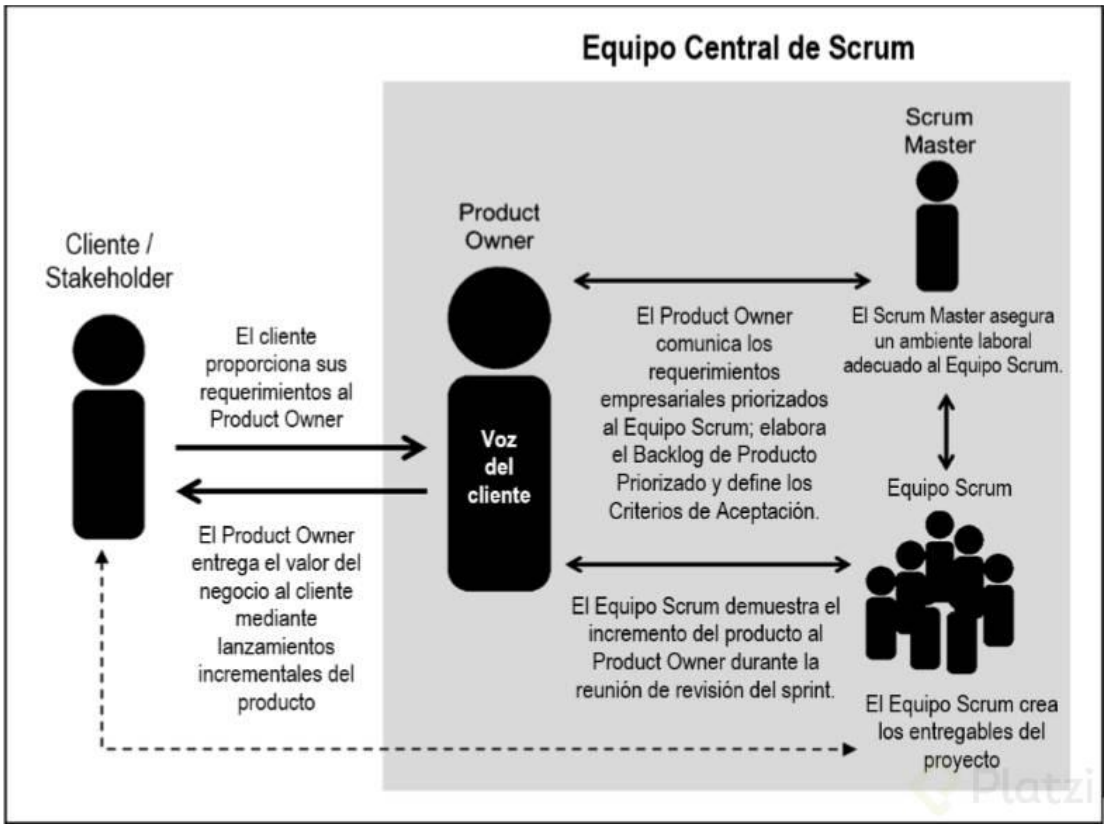
Estos son:

- StakeHolders.
    - Cliente.
    - Usuarios.
    - Patrocinador (*Sponsor*).
- Vendedores.
- Scrum Guidance Body.

En la siguiente grafica se muestra una descripción general entre los roles centrales del equipo *Scrum*.

Ilustración 2. Equipo Central Scrum

Fuente: Platzi



Inicio del ciclo

El ciclo inicia con una reunión de interesados (*StakeHolders*) en la que se crea la descripción de la visión del proyecto. Luego el *product owner* crea la lista priorizada del producto (*prioritized product backlog*) que contiene la lista de requerimientos en orden de prioridad para el negocio y el proyecto en forma de historias de usuario. (Cortés, Pineda Yesica, Platzi, 2017)

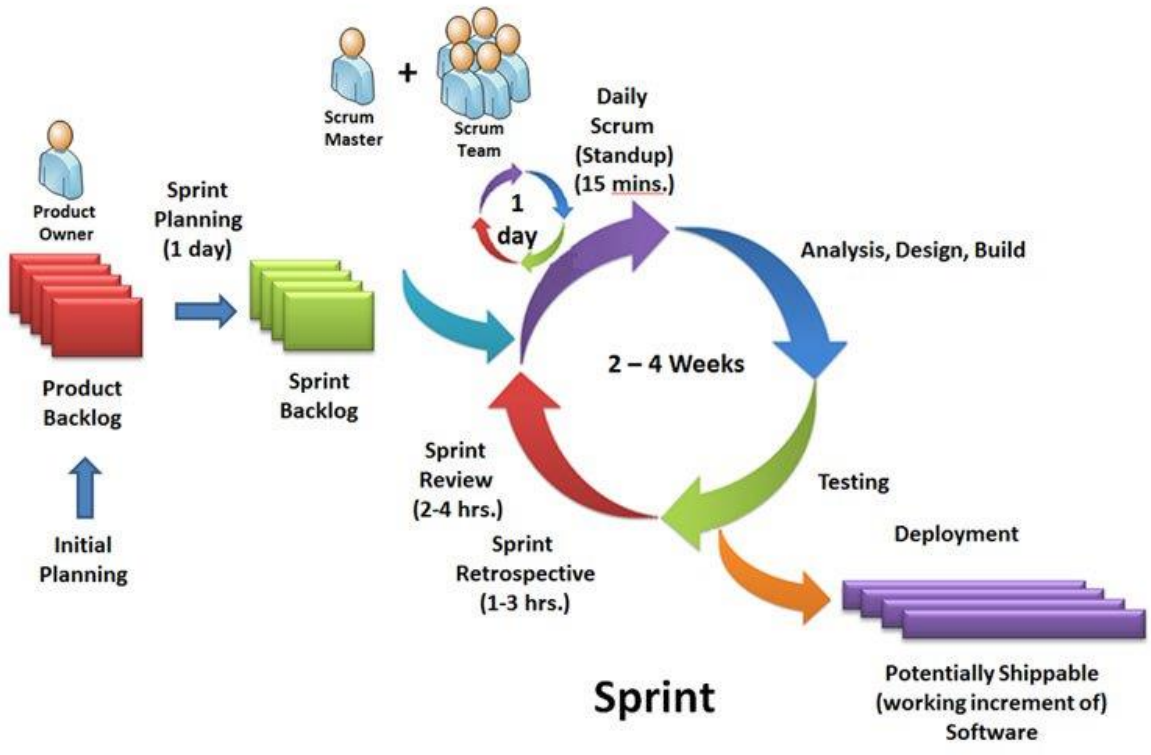
Sprint

Es la unidad básica de trabajo para un equipo *Scrum*. Esta es la característica principal y la que marca la diferencia entre *Scrum* y otros modelos de desarrollo ágil. Es una simple interacción llevada a cabo por los miembros del grupo. Un equipo puede completar varios *sprints* durante el desarrollo del proyecto. Un *sprint* inicia con un equipo que se compromete a realizar el trabajo y finaliza con la demostración de un entregable. El tiempo mínimo para un *sprint* es de una semana, y el máximo de 4 semanas. Dentro del desarrollo de un *sprint* se lleva acabo ciertos eventos, estos reciben el nombre de “Scrum Events o Eventos Scrum”. (Cortés, Pineda Yesica, Platzi, 2017).

Estos son:

Ilustración 3. Ciclo de un Sprint

Fuente: Platzi



Planteamiento del Sprint

Todos los involucrados en el equipo se reúnen para planificar el *Sprint*. Durante ese evento se decide que requerimientos o tareas se le asignará a cada uno de los integrantes del equipo. Cada integrante deberá asignar el tiempo que crea prudente para llevar a cabo su requerimiento. De esta manera se define el tiempo de duración del *Sprint*. (Lara, Walter, Platzi, 2017).

Reunión del equipo Scrum

> Los miembros del equipo y los clientes se reúnen para mostrar el trabajo de desarrollo de software que se ha completado. Se hace una demostración de todos los requerimientos finalizados dentro del *Sprint*. En este punto no es necesario que todos los miembros del equipo hablen. Pueden estar presentes pero la presentación está a cargo del Scrum Master y el Product Owner. (Lara, Walter, Platzi, 2017).

Retrospectiva del Sprint

En este evento, el Product Owner se reúne con todo su equipo de trabajo y su Scrum Master para hablar lo ocurrido durante el *Sprint*.

Herramientas Scrum

Backlog del producto

“Esto puede referirse a todo elemento que sea parte del proyecto. Puede ser un bug, una referencia o parte de un requerimiento. Brinda información a grandes rasgos del proyecto y muchas veces no son tomados como requerimientos oficiales”. (Lara, Walter, Platzi, 2017).

Historias de Usuarios

> Es un elemento especial del backlog del producto. Son llamados “historias” porque en ellas se proporciona información sobre como deber ser el comportamiento del requerimiento que se está trabajando. De igual manera, proporciona información directa del cliente en caso de existir algún

cambio. Generalmente estos si son tomados como requerimientos oficiales. (Lara, Walter, Platzi, 2017).

Backlog del Sprint

Es el conjunto de elementos tomados del backlog del producto que fueron priorizados, medidos y aceptados en las reuniones del *sprint planning*. Estos, en conjunto con sus respectivas historias de usuarios, forman oficialmente los requerimientos a elaborar en cada uno de los *sprint* que tendrá el proyecto.

Panel de tareas

Se encarga de mostrar las tareas que tiene asignada cada miembro del equipo. Esta tabla se divide en tres columnas que representan el estado de la tarea:

- Por hacer
- Haciendo
- Terminado

Al inicio del *sprint* todas están en la primera columna. Al momento de pasar una tarea a la columna número dos, indicara al Scrum master y al product owner qué está haciendo cada miembro del equipo y cuánto tiempo lleva trabajando en dicha tarea. Al finalizarla, esta debe cambiarse a la última columna, esto quiere decir que está listo para que *QA* haga las pruebas necesarias. (Lara, Walter, Platzi, 2017).

Conclusiones

Durante los seis meses de práctica en la empresa GRANDPA, las experiencias vividas en todos los campos fueron excelentes, estuve conviviendo con personas de increíble calidad humana sin dejar a un lado su profesionalidad, características fundamentales para un ambiente laboral tranquilo y ameno facilitando en gran medida, el buen desarrollo de mis funciones. En el área que me correspondía, aprendí la importancia de trabajar en un equipo diverso enmarcado en el desarrollo (full satck). Además, supe desenvolverme de manera práctica superando los obstáculos que se me presentaron durante el periodo de trabajo.

En cuanto al campo técnico, ha sido la experiencia laboral más enriquecedora hasta ahora, debido al acceso a recursos sin mencionar las herramientas disponibles para acoplarme a las tecnologías utilizadas en el desarrollo de aplicaciones tales como:

- JavaScript.
- React.js.
- Next.js.
- Node.js.
- Babel.js.
- Webpack.
- Apollo y Graphql.
- MongoDB entre otras.

Para mi eran nuevas herramientas, pero gracias al acompañamiento continuo por parte de mis compañeros, el aprendizaje se dio fácilmente lo que repercutió menormente

en mi crecimiento tanto personal como profesional, ya que, en el mercado actual, diversas organizaciones utilizan cada uno de los recursos mencionados.

Uno de mis mayores retos fue sin duda realizar parte de mi práctica académica en otro idioma, el inglés; no fue fácil pero cada día tuve la oportunidad de mejorar.

En cuanto a la parte metodológica, fue un nuevo reto ya que experimentaba por primera vez trabajar con una metodología ágil como lo es SCRUM. Pero a lo largo de mi trabajo, me fui acostumbrado de la mejor manera a las reuniones, a los sprint, a las tareas que semana a semana fui desarrollando. Realmente me sentí a gusto laborando bajo esta modalidad y espero seguir trabajando con ella.

En general, el tiempo de trabajo en la empresa fue productivo y positivo, pues carecía de la toxicidad y el estrés que en ocasiones caracterizan el ambiente laboral. El trato con los compañeros fue muy agradable y más que compañeros se convirtieron en grandes amigos.

Referencias

Auth0. (2018). Introduction to JSON web tokens. Recuperado de https://jwt.io/introduction/.

Babel.js. (2018). ES2015 and beyond. Recuperado de http://babeljs.io/.

Cortés, Pineta, Yesica. (2017). Que es Scrum y los roles en Scrum. Recuperado de https://platzi.com/blog/que-es-scrum-y-los-roles-en-scrum/.

Facebook Inc. (2014 – 2018). Introduction to type checking with Flow. Recuperado de https://flow.org/en/docs/getting-started/

Facebook Inc. (2018). Declarative. Recuperado de https://reactjs.org/

Facebook Inc. (2018). Introduction to GraphQL. Recuperado de http://graphql.org/learn/

Google Inc. (2018). Basics. Recuperado de https://dialogflow.com/docs/getting-started/basics

JSON. (2018). Introducing JSON. Recuperado de https://json.org/

Lara, Walter. (2015). Metodología Scrum Fases. Recuperado de https://platzi.com/blog/metodologia-scrum-fases/

Meteor Inc. (2018). What is Apollo Client and what does it do?. Recuperado de https://www.apollographql.com/docs/react/

Microsotf. (2018). Visual Studio Code en Acción. Recuperado de https://code.visualstudio.com/docs

MongoDB. (2018). Reinventado la gestión de los datos. Recuperado de https://www.mongodb.com/es

MongoDB. (2018). Schemas. Recuperado de http://mongoosejs.com/docs/guide.html

Node.js Fundation. (2018). Acerca de Node.js. Recuperado de https://nodejs.org/es/about/

NPM Inc. (2018). What is npm?. Recuperado de https://docs.npmjs.com/getting-started/what-is-npm

Redux.js. (2018). Redux. Recuperado de https://es.redux.js.org/

StrongLoop Inc. (2018). Express Infraestructura web rápida, minimalista y flexible para Node.js. Recuperado de http://expressjs.com/es/

Uploadcare. (2011-2018). Docs Intro. Recuperado de https://uploadcare.com/docs/

Webpack. (2018). Concepts. Recuperado de https://webpack.js.org/concepts/

Zeit Inc. (2018). Next.js 5: Universal Webpack , CSS Imports, Plugins and Zones. Recuperado de https://zeit.co/blog/next5