Alafia: Desarrollo de una Progressive Web Application para la gestión de pedidos y de experiencias inmersivas para un restaurante africano

Proyecto Fin de Grado

Grado en Ingeniería de Computadores

Autor:

Daniel García Alonso

Tutores:

Carlos Camacho Gómez

Curso académico

2021/2022

# Resumen

Cada día más, la sociedad va avanzando hacia un mundo más unificado, las fronteras son cada vez más sutiles. Esto pasa cada vez más sobretodo con culturas más evolucionadas, del primer mundo. Pero también es notable el hecho de que juntarnos con unas culturas menos desarrolladas, a veces, puede generar cierto rechazo.

Por ese motivo nace Alafia, un movimiento que pretende retirar ciertos estigmas que nos retrasa el desarrollo como sociedad. Alafia es un restaurante cuyo foco principal gastronómico es el de las culturas Africanas. Comida típica de la zona de Nigeria, realizada con ingredientes de origen Africano y empleados provenientes de la zona es lo que ofrece Alafia. Además de la fusión con el mundo contemporáneo con el uso de últimas tecnologías en la mesa, ya que toda la gestión de solicitudes de los clientes, es puramente digital a través de una Tablet que dispone cada cliente en la mesa.

El dispositivo en cuestión, muestra una aplicación web, con el que cada cliente podrá gestionar comida, bebida y servicios del restaurante, de forma que todos los movimientos quedan registrados digitalmente.

Por tanto Alafia es, la excusa perfecta para el cometido de este proyecto, el desarrollo de una aplicación web con su servidor, para la gestión de un restaurante.

# Abstract

Every day more often, society progress to a more unified world, borders between countries are even more fine-drawn. This happens over all with the more developed cultures of the first world. Despite of, is true that meet with less developed cultures, sometimes, could generate reject.

For that reason, Alafia born. A social movement whose intend is remove some possible stigmas that our actual society could have. Alafia is a restaurant whose gastronomic primary focus is African cultures. Food coming directly of Niger, made with ingredients with African origin and employees coming of the Niger area is that Alafia offers to clients. Furthermore, fusion with the contemporary world, using latest technologies in the restaurant table. All the claims of the clients are done digitally with a Tablet that each client will have available on the restaurant table.

The device, shows a web application, and each client could manage food, drinks and all services that the restaurant can offer to the clients. As set out all movements are registered in the system digitally.

Henceforth Alafia is the perfect pretext to this project, develop a web application with a server behind, to manage a restaurant.

Tabla de contenido

[Resumen 2](#_Toc92020036)

[Abstract 3](#_Toc92020037)

[Índice de Figuras 5](#_Toc92020038)

[Índice de Tablas 6](#_Toc92020039)

[Estudio del mercado 7](#_Toc92020040)

[Tecnologías en el mercado 8](#_Toc92020041)

[React.js 9](#_Toc92020042)

[Angular 9](#_Toc92020043)

[Flutter 9](#_Toc92020044)

[Node.js 10](#_Toc92020045)

[Spring Boot 10](#_Toc92020046)

[Ruby on Rails 10](#_Toc92020047)

[MySQL 11](#_Toc92020048)

[MongoDB 11](#_Toc92020049)

[Elección de tecnologías 12](#_Toc92020050)

[Front End (Angular) 13](#_Toc92020051)

[Back End (Spring Boot) 14](#_Toc92020052)

[Base de datos (MongoDB) 15](#_Toc92020053)

[Funcionalidades de la aplicación 16](#_Toc92020054)

[El problema de sincronización 17](#_Toc92020055)

[Impacto social 18](#_Toc92020056)

[Conclusiones 19](#_Toc92020057)

[Agradecimientos 20](#_Toc92020058)

[Bibliografía 21](#_Toc92020059)

# Índice de Figuras

# Índice de Tablas

# Estudio del mercado

En el mercado actual, todos hemos visto como cada vez más, los empleados de los comercios de restauración (bares, restaurantes, etc.) incorporan en su atuendo un dispositivo digital para la gestión de las mesas del comercio o de las mesas que tenga asignadas al menos.

Alafia pretende llevar esto al siguiente nivel. No solo los empleados tendrán su dispositivo personal para monitorizar cualquier situación del restaurante, cada cliente dispondrá de un dispositivo justo en frente de su sitio en la mesa que tenga asignada.

De forma que toda comunicación cliente – restaurante queda registrada y no hace necesaria la comunicación habitual que se da hoy en día. Evitando así malinterpretaciones por parte de los clientes y por supuesto del restaurante.

# Tecnologías en el mercado

Para el desarrollo de la aplicación, hoy más que nunca tenemos una gran variedad de tecnologías a poder elegir. Como se trata de un desarrollo del aplicativo completo, es decir, que no solo se va a construir la parte visual (Front End), sino que va a ser totalmente funcional y va a llevar el respaldo de un servidor (Back End) para el procesado y guardado de los datos en la base de datos del sistema.

Por tanto, tenemos que estudiar posibles tecnologías para 3 cometidos diferentes. Abordar en Front End, abordar el Back End y abordar la base de datos. Para quitar cualquier atisbo de dudas, primero definamos que es cada pieza.

El Front End, en pocas palabras, se trata de una serie de tecnologías que son usadas para el desarrollo de la interfaz visual de aplicaciones web, con el fin de diseñar, estructurar, animar o adaptar la interfaz a los usuarios de una forma atractiva para ellos.

El Back End, como adelantamos en párrafos anteriores, es el servidor que va a estar en comunicación con el Front End, ejecutando procesos con datos que le lleguen del Front End por ejemplo. Hace tambien de interfaz para la base de datos, y atiende a las peticiones que le lleguen del Front End.

Y por último, la base de datos, como su propio nombre indica, es el conjunto de datos almacenados en un computador, que nos permite realizar consultas, inserciones, modificaciones y borrado de los datos almacenados.

Teniendo todo definido, expongamos tecnologías disponibles en el mercado de cada uno, empezando por el Front End:

## React.js

Se trata de una librería open-source licenciada por el MIT y lanzada al mercado en el 2013. Probablemente sea el framework de JavaScript más usado, y con cada actualización mejora cada vez más.

La mantiene el gigante Facebook, y destaca por la facilidad de reusabilidad de sus componentes, ya que hace que las aplicaciones se subdividan fácilmente en componentes.

Algunos ejemplos de uso comercial son Airbnb, Dropbox, BBC, Facebook, New York Times, y Reddit.

## Angular

Es la segunda librería más usada en el mercado, y compite cara a cara con la anterior, React.js. Es una librería open-source licenciada por el MIT y lanzada en el año 2009.

La mantiene otro gigante de la tecnología, Google. Destaca por su misión de estandarizar las aplicaciones web, usando módulos y estructuraciones de código estandarizadas.

Ejemplos de uso comercial son Paypal, Gmail, IBM, Netflix y The Guardian.

## Flutter

Fundada por Google como Angular, permite a los desarrolladores crear interfaces responsive y vistosas. Licenciada bajo BSD (*Berkeley Software Distribution*).

Su principal atractivo es que como parte de la aplicación web, Flutter permite al desarrollador formar la página con widgets, lo cual la hace muy responsive. Además permite introducir cambios en el código y hacer un refresco de la página en caliente, lo cual agiliza mucho los desarrollos.

Hookel, Birch Finance, Alibaba, son algunos ejemplos que han optado por esta tecnología para exponer sus Front End a sus clientes.

Expuestos algunos ejemplos de tecnologías de Front End, pasemos a exponer algunos ejemplos de tecnologías de Back End que están disponibles en el mercado y que son usadas con gran éxito:

## Node.js

Es un framework de JavaScript como bien deja ver el propio nombre, y eso mismo lo hace tan viable para los desarrolladores que buscan convertirse en Full-Stack, ya que la parte de JavaScript la tienen muy vista.

Se trata básicamente de escribir el código del servidor con JavaScript, lo cual lo convierte en el framework más amigable para los desarrolladores de Front End. Es decir, que convierte a JavaScript en un lenguaje end-to-end en una aplicación web.

Ejemplos de usos comerciales son LinkedIn, Netflix, Uber, PayPal, NASA y eBay.

## Spring Boot

Es un framework de Java, y su intencionalidad se basa en solventar los problemas que tiene Spring para el desarrollo de aplicaciones web, al igual que hace Spring con Java EE.

Spring por otro lado, es básicamente un inyector de dependencias para Java, lo cual nos brinda de muchas funcionalidades extra fácilmente, como exponer un API en el lado del servidor por ejemplo usando la dependencia de spring-web.

Ejemplos de usos comerciales son Accenture, Zalando, MIT, Santander y Fitbit

## Ruby on Rails

Es el framework para los desarrolladores que no les atrae Java, JavaScript ni Python.

Se basa en el lenguaje Ruby como su nombre deja intuir, y permite renderizar plantillas HTML, actualizar bases de datos, enviar y recibir emails, provee de seguridad, etc. Es decir, lo hace todo, el Back y el Front End.

Ejemplos de usos comerciales son GitHub, Twitch, SoundCloud y Coinbase.

Expuestas algunas de las tecnologías más usadas para el desarrollo de Back End y Front End, es hora de analizar algunas de las tecnologías que tenemos disponibles en el mercado para escoger una base de datos adecuada.

## MySQL

MySQL es un gestor de Bases de relacionales de código abierto con un modelo cliente-servidor. Para que sea posible esta comunicación, se usa el lenguaje SQL (Structured Query Language). Y nos permite realizar operaciones sobre la base de datos tales como:

* Control de acceso a los datos: decidir quién y puede y quien no puede ver o usar los datos almacenados en la base de datos.
* Identidad de datos: definir la tipología de los datos y la definición del esquema y relaciones de la base de datos.
* Consulta de datos: solicitar la información que se desea obtener que esté almacenada en la base de datos.
* Manipulación de datos: realizar operaciones a los datos tales como agregar, modificar datos existentes, eliminar registros, ordenación de datos, etc.

Ejemplos de uso comercial son empresas tales a Facebook, Twitter, YouTube, Yahoo!...

## MongoDB

MongoDB es una base de datos no relacional que opera con documentos a diferencia de MySQL por ejemplo que trabaja con tablas. Los documentos que usa para el almacenamiento de la información tienen un formato JSON.

Dentro de un documento, nos permite no anclarnos a un único esquema, lo cual nos da más flexibilidad que una tabla de un sistema de base de datos relacional.

Las consultas que podemos realizar se basan tambien en JSON, ya que son llamadas a funciones sobre una consola JavaScript y lo que le pasaremos como parámetro será la consulta en un objeto JSON.

MongoDB por tanto es perfecto para cuando necesitamos almacenar datos semi estructurados, y se suele usar para aplicación CRUD (Create Read Update Delete) en muchos de los desarrollos web actuales.

Ejemplos de uso comerciales actuales son empresas como Ebay, Google, SquareSpace, Sega y Electronic Arts.

# Elección de tecnologías

Teniendo en cuenta la preselección de tecnologías hecha en el estudio de las disponibles actualmente en el mercado para el desarrollo de una aplicación web, toca escoger las más viables para el desarrollo de este proyecto.

Empecemos por el Front End.

Aunque para el propósito de Alafia cualquiera de las tres tecnologías expuestas anteriormente, React, Angular y Flutter, vamos a escoger Angular, ya que su principal cometido es estandarizar mediante el uso de módulos los desarrollos del Front End. Lo cual me parece un punto muy a favor de esta tecnología. Además, cuenta con una comunidad muy grande en las redes ya que su popularidad está en auge, y la facilidad de trabajo al usar metodologías más estándares es vital para los desarrolladores como yo que no tenemos experiencia creando aplicaciones web.

Pasemos a la elección de la tecnología de Back End.

En este apartado, haremos una poda inicial sobre las tres tecnologías elegidas, Node.js, Spring Boot y Ruby on Rails. Vamos a eliminar esta última, ya que su cometido es hacerlo todo (Front y Back End) ya que con el desarrollo de este proyecto se busca el uso de varias tecnologías para su aprendizaje entre otras cosas. Por tanto la elección se hace sobre Node.js y Spring Boot. Ambas dos nos sirven para desarrollar un servidor que haga de interfaz para la base de datos, pero ya que en el grado se hace más hincapié sobre el lenguaje Java que sobre JavaScript, vamos a elegir el framework que trabaja sobre Java, es decir, Spring Boot.

Por último, la elección de la base de datos.

Se han expuesto dos muy diferentes en el apartado de análisis de bases de datos disponibles en el mercado, una base de datos relacional, el modelo tradicional, y otra base de datos no relacional, más novedosa. El modelo de datos de Alafia, no es un modelo complejo como tal, lo veremos más a detalle en el capítulo [Base de datos (MongoDB)](#_Base_de_datos). La comunicación con el cliente, el Front End se hace mediante el uso de objetos JSON. Por tanto, y ya que no requerimos de transacciones complejas, y el modelo de datos no se ciñe a un esquema cerrado, sino que está abierto a posibles objetos ligeramente diferentes en estructura pero conteniendo información parecida, escogemos MongoDB para la capa de persistencia de la aplicación.

# Front End (Angular)

# Back End (Spring Boot)

# Base de datos (MongoDB)

Aunque MongoDB sea una base de datos no relacional, no significa que no se puedan relacionar datos en su almacenamiento. El modelo de Alafia es un modelo parecido a un árbol, ya que cada nodo tiene hijos relacionados con el padre.

Para representarlo visualmente, veamos el diagrama Entidad Relación simplificado del modelo.

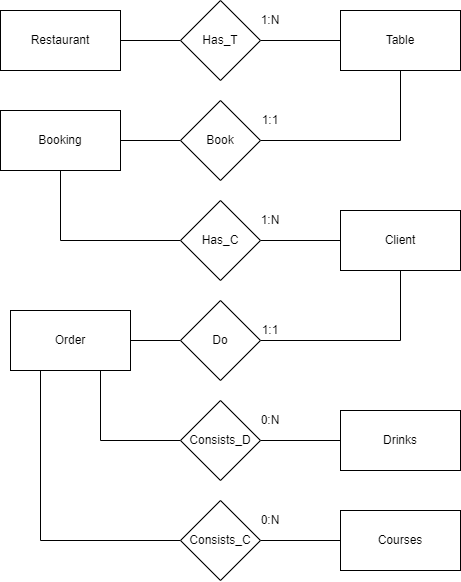


Figura 1. Diagrama Entidad Relación Alafia

Como podemos observar, todo parte de un documento padre, que es Restaurant. Este documento es necesario modelarlo ya que de cara al futuro, contemplo una posible expansión de Alafia a más de un restaurante, y por tanto es necesario el modelado de dicha información.

La estructura por tanto es, explicada de palabra, la siguiente. Un restaurante (**Restaurant**), contiene mesas (**Table**). Las mesas solo están en un restaurante, y un restaurante puede tener muchas mesas, por tanto relación 1:N. De cada restaurante se guarda su id único y las mesas que contiene, al igual que de cada mesa, únicamente su id único y el cliente que la reserva.

La mesa se reserva por un cliente que realiza la reserva (**Booking**), y no interesa mantener el histórico de las mesas reservadas por los clientes, por lo que con una relación 1:1 nos vale.

Cada cliente que realiza una reserva, la hace para él y para sus acompañantes (**Client**), que pueden ser uno, contándose a sí mismo o varios, lo cual nos deja una relación 1:N. De los clientes que hacen la reserva se guarda un id único, además de la información que se guarda de todos los clientes, como la de sí mismo como cliente. De los clientes se guarda un id único por cliente, su nombre y su correo electrónico en primera instancia. Se guardan además su comanda personal, sus respuestas al test de medición del impacto social y si ha confirmado o no su asiento en la mesa.

A cada cliente se le asigna una comanda (**Order**) que es única y efímera durante la estancia del cliente en el restaurante. Por tanto, relación 1:1. Y de cada comanda se guarda su identificador único, los platos contenidos durante la estancia, los platos que se le han servido ya, y las bebidas que ha pedido el cliente.

Cada cliente puede pedir varios platos y bebidas, o no pedir nada, por tanto relaciones 0:N. de cada plato (**Course**) se guarda su identificador único, una descripción del plato, su precio, los valores nutricionales, información de cómo se obtienen los ingredientes y la historia de ese plato. Y de cada bebida (**Drink**) se guarda un identificador único, la descripción y el precio.

# Funcionalidades de la aplicación

# El problema de sincronización

# Impacto social

# Conclusiones

# Agradecimientos

# Bibliografía

* <https://www.credencys.com/blog/front-end-technologies/>
* <https://cadabra.studio/blog/best-backend-technologies-list-comparison-examples>
* <https://medium.com/javarevisited/10-best-frontend-and-backend-frameworks-for-java-python-ruby-and-javascript-developers-cce3c951787a>
* <https://www.hostinger.es/tutoriales/que-es-mysql>
* <https://www.genbeta.com/desarrollo/mongodb-que-es-como-funciona-y-cuando-podemos-usarlo-o-no>
* <https://www.mongodb.com/es>
* <https://aws.amazon.com/es/redis/>
* <https://www.oracle.com/es/mysql/>