Foto del alumno

(foto actual)

MÓDULO PROYECTO

Ciclo Superior Desarrollo de Aplicaciones Web

Departamento: Informática y Comunicaciones

IES “María Moliner”

Curso: 2023/2024

Grupo: S2I

Proyecto: Comida En Marcha

**Fernando Juan Estrada Gallardo**

Email: fern.j.e.g@gmail.com

Tutor individual: María José González García

Tutor colectivo: Enrique Carballo Albarrán

Fecha de presentación: (fecha de presentación)

Contenido

[Descripción general del proyecto 4](#_Toc165484125)

[Objetivos 4](#_Toc165484126)

[Cuestiones metodológicas 4](#_Toc165484127)

[Entorno de trabajo 4](#_Toc165484128)

[Descripción general del producto 6](#_Toc165484129)

[Límites del sistema 6](#_Toc165484130)

[Funcionalidades básicas 6](#_Toc165484131)

[Usuarios 6](#_Toc165484132)

[Hosting 6](#_Toc165484133)

[Planificación y presupuesto 9](#_Toc165484134)

[Planificación 9](#_Toc165484135)

[Metodología 9](#_Toc165484136)

[Bocetos 10](#_Toc165484137)

[Interfaces finales 12](#_Toc165484138)

[Página de error 12](#_Toc165484139)

[Presupuesto 13](#_Toc165484140)

[Modelo de negocio 13](#_Toc165484141)

[Documentación técnica 14](#_Toc165484142)

[Entidades 14](#_Toc165484143)

[Rutas API 14](#_Toc165484144)

[Estructura del código 15](#_Toc165484145)

[Librerías 15](#_Toc165484146)

[Pruebas 16](#_Toc165484147)

[Manuales de usuario 17](#_Toc165484148)

[Instalación 17](#_Toc165484149)

[Uso 17](#_Toc165484150)

[Conclusiones y ampliaciones 18](#_Toc165484151)

[Conclusiones 18](#_Toc165484152)

[Ampliaciones 18](#_Toc165484153)

[Relación de ficheros en formato digital 19](#_Toc165484154)

[Bibliografía 20](#_Toc165484155)

[Glosario 21](#_Toc165484156)

# Descripción general del proyecto

## Objetivos

Se pretende la realización de una aplicación web que sirva para varios establecimientos y consiga reemplazar a parte del capital humano que se dedica a pedir nota. Para visualizarlo mejor un ejemplo parecido son las típicas pantallas de sitios que sirven comida rápida.

Como objetivos secundarios se pretende permitir ver todos los pedidos, configurar el sitio con datos del restaurante y otras customizaciones.

En el apartado de conclusiones y ampliaciones habrá más información sobre objetivos futuros.

## Cuestiones metodológicas

## Entorno de trabajo

La herramienta de control de versiones usadas es GitHub. El enlace al repositorio es el siguiente: <https://github.com/FerZeg/TFG>

Como herramienta principal del desarrollo de la aplicación se usará Visual Studio Code. También cabe destacar MongoDBCompass, una interfaz para gestionar la base de datos.

Para planificar la aplicación se ha usado Notion, un software de gestión de proyectos y para tomar notas. Con esta aplicación se ha organizado tanto la temporalidad del proyecto como las diferentes tareas a realizar.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

# Descripción general del producto

## Límites del sistema

La aplicación es totalmente autónoma, es decir, no depende de otro proyecto web. Al ser una aplicación web puede ser accesible desde cualquier lado en cualquier momento.

## Funcionalidades básicas

La aplicación se define por las siguientes funcionalidades básicas:

* Gestión de tickets. (Ver, borrar)
* Gestión de usuarios y roles.
* Gestor de pedidos de comida.
* Gestión de datos internos.

## Usuarios

La aplicación para la posible futura ampliación se ha desarrollado de tal manera que pueda haber usuarios con más privilegios que otros. En el contexto de la aplicación hay 2 tipos de usuarios; los superadmin y los usuarios normales. Para la presentación de esta aplicación un usuario superadmin solo podría interactuar con la API de la plataforma, no se ha diseñado el frontend para soportarlo.

Dentro de los usuarios normales existen otros dos tipos de usuarios hasta la fecha. Los cocineros tienen acceso únicamente a las funcionalidades básicas del restaurante, como ver los pedidos o editar los platos. Por otro lado, los administradores tienen acceso completo al entorno de su restaurante, lo que les permite modificar los datos del establecimiento, gestionar los usuarios y mesas, y acceder a los tickets.

## Hosting

Hay 3 hostings utilizados, diferenciando la parte del front(interfaces de usuario), la del servidor y la de la base de datos.

* Para las diferentes interfaces se usa Cloudflare.
* Para el servidor se usará Northflank.
* Para la base de datos se usará Mongo Atlas.

Ambos son de uso gratuito, siendo Cloudflare un *CDN (Content Delivery Network)* bastante popular y ampliamente usado por muchas aplicaciones. Al ser el frontend una compilación en archivos estáticos nos podemos aprovechar de su red mundial para servir los archivos con muy baja latencia en cualquier parte del mundo.

Northflank es una plataforma Cloud relativamente nueva, que trae muchas funcionalidades para facilitar el despliegue de la aplicación. Con un dockerfile sencillo defino el entorno de ejecución de mi servidor.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Captura de pantalla de un celular

Descripción generada automáticamente

Entre otras funcionalidades este servicio ofrece:

* Panel de control donde ver los logs de la aplicación
* Diferentes tipos de métricas
* Una consola de la instancia que se está ejecutando
* Sistema para exponer puertos diferentes según las necesidades
* Herramienta para configurar dominios customizados, certificados de SSL automáticos al conectar el domino.
* Ejecutar código cada cierto tiempo (Cron Jobs) o manualmente.
* Bases de datos.
* Usuarios y equipos.
* Notificaciones.

Todo esto con una capa gratuita bastante generosa para probar proyectos sin un tráfico excesivo.

Captura de pantalla de computadora

Descripción generada automáticamente

Ambas plataformas ofrecen un servicio de *CD (Continuos Deployment),* lo que facilita muchísimo el despliegue de la aplicación al ser automatizado al hacer un push a la rama Main del repositorio.

# Planificación y presupuesto

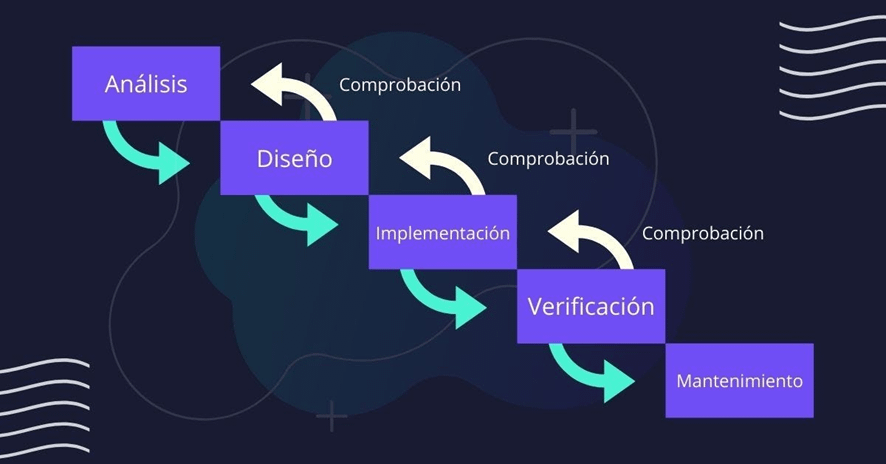
## Planificación

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

### Metodología

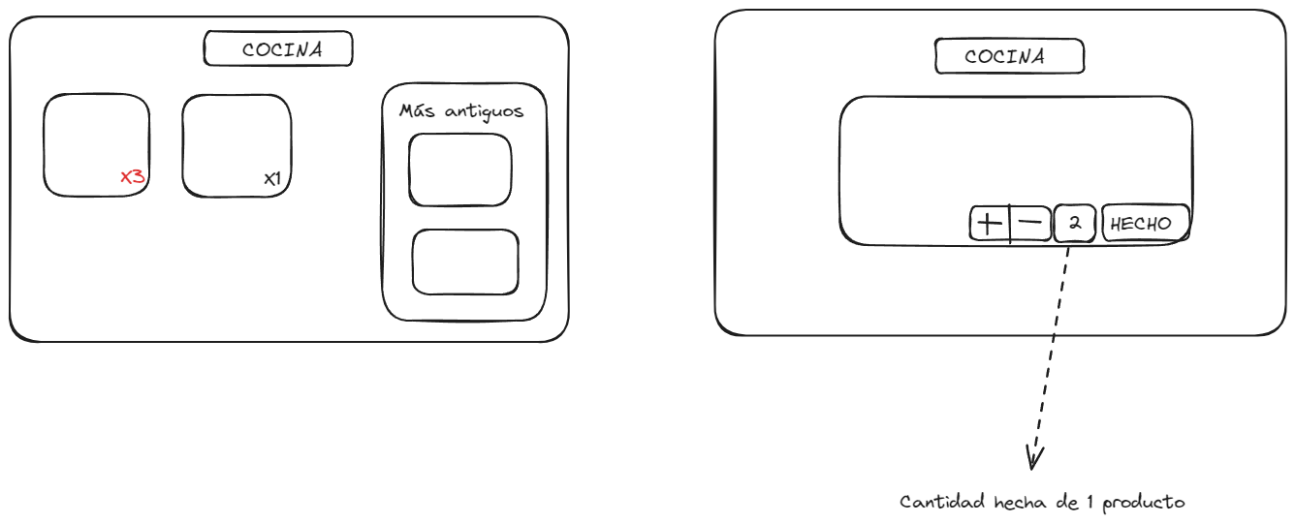
Método Waterfall o Cascada

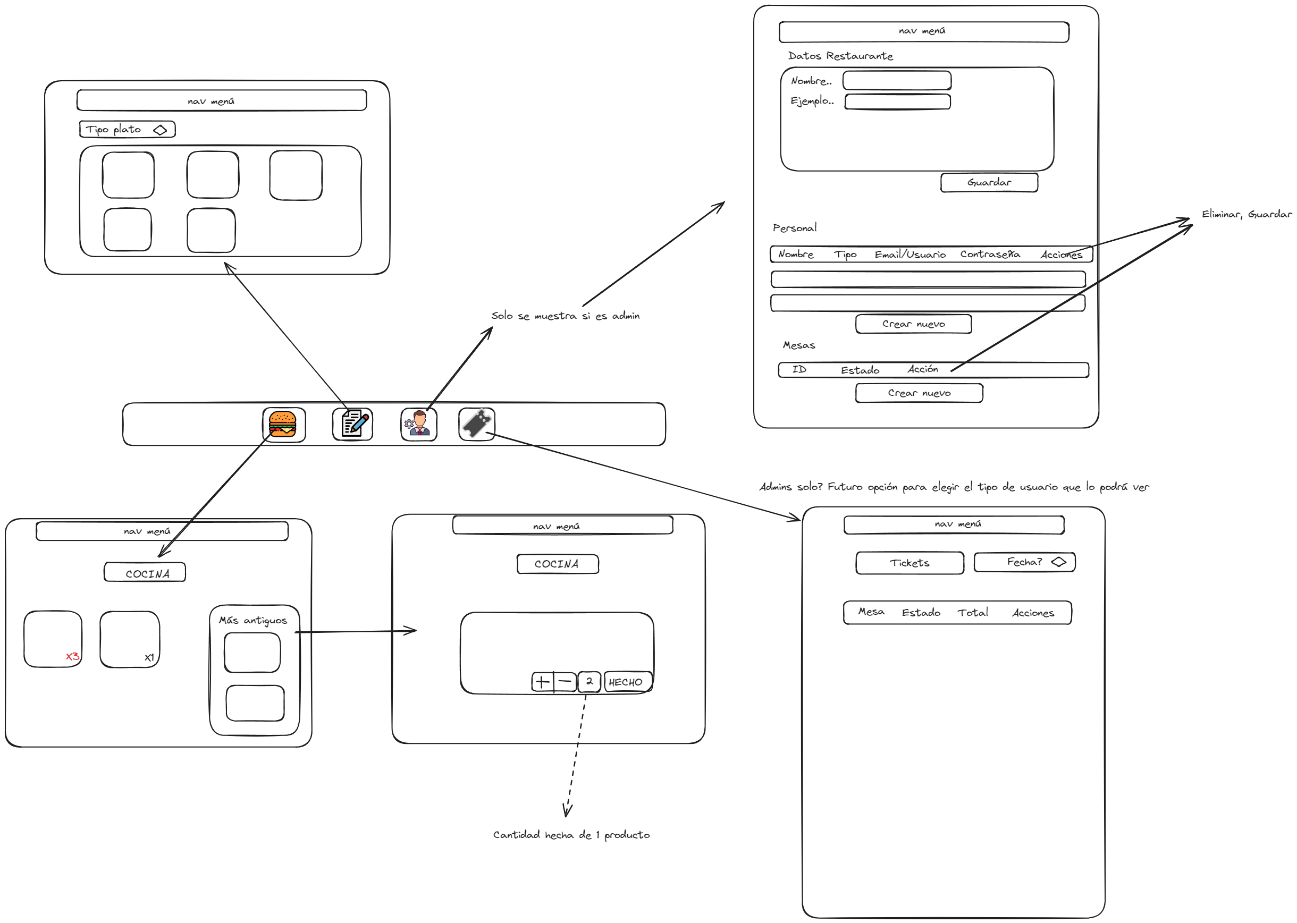


### Bocetos

Forma, Rectángulo

Descripción generada automáticamente





## Interfaces finales

### Página de error

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

## Presupuesto

El groso del coste del desarrollo de la aplicación es el capital humano ya que las herramientas utilizadas son todas software libre, así como los alojamientos están usando capas gratuitas de diferentes proveedores.

El único coste extra a parte del propio trabajo ha sido el dominio cocinaenmarcha.com que tiene un coste de 10€/año.

El coste por horas de un programador Junior oscila entre 15€ - 20€ la hora en España. Al ser el primer proyecto el código base se computará a 15€ la hora.

El coste total por lo tanto será, 140 horas \* 15 € + 10 € = 2110€.

## Modelo de negocio

Esta aplicación no está diseñada para venderse a un cliente particular, si no para ser un modelo de suscripción a contratar por diferentes establecimientos.

El coste sería de alrededor de 150€/mes por establecimiento, y debido a la necesidad de ampliar los recursos de servidor se tendría un coste variable de 10€ por establecimiento.

Se tendrán en cuenta proposiciones de cambios o si hubiese una necesidad de una funcionalidad personalizada se presupuestaría.

Los mínimos como objetivo serían tener 5 clientes al mes y como cifra de éxito serían alrededor de 10. Estos darían un ingreso de 700€ o 1400€ respectivamente siendo el único trabajo un mantenimiento pasivo y ocasionalmente trabajos presupuestados.

# Documentación técnica

## Entidades

He definido las diferentes entidades existentes de la aplicación mediante un archivo de typescript para visualizar los “tipos” claramente. La base de datos en las que trabajaran estas entidades será MongoDB; una base de datos no relacional muy flexible.

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

## Rutas API

**Restaurantes**

**POST /restaurantes**

**GET /restaurantes/:id**

**PUT /restaurantes/:id**

**DELETE /restaurantes/:id**

**Mesas**

**GET /mesas**

**POST /mesas**

**GET /mesas/:id**

**PUT /mesas/:id**

**DELETE /mesas/:id**

**Tickets**

**GET /mesas/:mesa\_id/tickets**

**POST /mesas/:mesa\_id/tickets**

**GET /mesas/:mesa\_id/tickets/:id**

**DELETE /mesas/:mesa\_id/tickets/:id**

**Pedidos**

**GET /tickets/:ticket\_id/pedidos**

**POST /tickets/:ticket\_id/pedidos**

**GET /tickets/:ticket\_id/pedidos/:id**

**PUT /tickets/:ticket\_id/pedidos/:id**

**DELETE /tickets/:ticket\_id/pedidos/:id**

**Platos**

**GET /platos**

**POST /platos**

**GET /platos/:id**

**PUT /platos/:id**

**DELETE /platos/:id**

**Cocineros**

**GET /cocineros**

**POST /cocineros**

**GET /cocineros/:id**

**PUT /cocineros/:id**

**DELETE /cocineros/:id**

**WebSocket para Cocineros**

**WS /cocineros/:id/socket**

## Estructura del código

### Librerías

Librerías principales del proyecto en el lado del servidor:

* Express: Framework de desarrollo para facilitar la elaboración del servidor, consta de un sistema avanzado de routing y librerías para parsear las cookies u otros.
* Mongoose: Interfaz para la base de datos Mongo, permite crear Schemas (Estructuras de objetos) y muchas funciones añadidas.
* Bcrypt: Librería para encriptar las contraseñas.
* Jsonwebtokens: Librería para generar tokens con una carga que se pueden codificar, decodificar y verificar mediante una contraseña. Es muy útil su uso para el sistema de autentificación y persistencia de la sesión.

Librerías principales del proyecto en el lado del cliente:

* React: Framework de javascript creado por Facebook (actualmente META) que facilita mucho la creación de software mediante la encapsulación de código, tanto el lenguaje de marcado como las funcionalidades, para una mayor reutilización de este. Es actualmente el framework de javascript más usado en la actualidad.
* Vite: Herramienta de compilación de código ampliamente usada en la actualidad por su rapidez en tiempo de compilación o actuando como servidor de desarrollo para aplicar los cambios en el código en tiempo real cuando guardas un archivo.
* Sonner: Librería que se encarga de las notificaciones mediante cuadros de diferentes colores.
* React-router-dom: Librería hecha para react pero que no es parte de ella directamente, la uso para manejar las rutas de la aplicación y crear una SPA (Single Page Application).

## Pruebas

# Manuales de usuario

## Instalación

## Uso

# Conclusiones y ampliaciones

## Conclusiones

## Ampliaciones

## Relación de ficheros en formato digital

# Bibliografía

Modelar los datos con MongoDB:

* <https://www.youtube.com/watch?v=YsaOcUDUJKY>
* <https://www.youtube.com/watch?v=Hidk36H6hBY>

Información sobre metodología incremental:

* <https://www.reddit.com/user/BackNativos22/comments/taiqv2/describir_el_modelo_y_sus_principales/>

# Glosario

* **CDN (Content Delivery Network):** Red de servidores interconectados que contienen copias locales de contenidos para distribuirlos rápidamente desde el nodo más cercano al usuario.
* **CD (Continuos Deployment):** Es una estrategia por la cual se publican automáticamente los cambios hechos a producción (producto final).
* **SPA (Single Page Application):** Aplicación web que interactúa con el usuario repintando la aplicación. Esto consigue crear una aplicación que no cambia de página (no se refresca). Se ahorran peticiones en el servidor y se gana velocidad de interacción al utilizarlas.