

MEMORIA FINAL DE PROYECTO

FiestaGochi

CICLO FORMATIVO DE GRADO SUPERIOR
DESARROLLO DE APLICACIONES WEB

AUTORES

JORGE COSTA MACIÁ

TUTOR

DANIEL CALATAYUD GINER

COORDINADOR

JORGE COSTA MACIÁ

ÍNDICE

1 INTRODUCCIÓN	6
2 INTRODUCCION EN INGLÉS	7
3 ESTUDIO DE VIABILIDAD	8 - 14
3.1 Estado actual del sistema	8
3.2 Requisitos del cliente	8 - 10
3.3 Posibles soluciones	11
3.4 Solución elegida	11 - 13
3.5 Planificación temporal de las tareas del proyecto FiestaGochi	14
3.6 Planificación de los recursos a utilizar	14
4 ANÁLISIS	15 - 16
4.1 Requisitos funcionales	15
4.2 Requisitos no funcionales	16
5 DISEÑO	17 - 19
5.1 Estructura de la aplicación	17
5.2 Componentes del sistema	17
5.3 Herramientas	17
5.4 Paquetes	18 - 19
6 IMPLEMENTACIÓN	20 - 44
6.1 Entorno de implementación	20
6.2 Colecciones y APIS creadas	20 - 21
6.3 Documentos y colecciones	21 - 24
6.4 Métodos API rol USER	24 - 25
6.5 Métodos API rol ADMIN	25 - 26
6.6 Publicaciones	26 - 30
6.7 Subscripciones	30 - 33

6.8 Carga de datos	33 - 34
6.9 Ficheros de configuración servidor	34 - 36
6.10 Ficheros de configuración cliente	37 - 40
6.11 Ficheros de configuración despliegue producción	40
6.12 Configuraciones realizadas en el sistema	41 - 43
6.13 Implementaciones de código realizadas	43 - 44
7 PRUEBAS	45 - 52
7.1 Casos de pruebas	45 - 52
8 EXPLOTACIÓN	53 - 56
8.1 Planificación	53 - 54
8.2 Plan de formación	54
8.3 Implantación propiamente dicha	55 - 56
8.4 Pruebas de implantación	56
9 DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN	57 - 61
9.1 Incidencias	57 - 60
9.2 Control de versiones	60 - 61
10 CONCLUSIONES	62 - 64
11 FUENTES	65
11.1 Legislación DAW	65
11.2 Fuentes utilizadas	65
12 ANEXOS	66 - 141
12.1 Manual de usuario	66 - 121
12.1.1 Consideraciones técnicas	66 - 81
12.1.1.1 Emails y recursos corporativos	67
12.1.1.2 Actividad	67
12.1.1.3 Usuarios y permisos	68 - 69

<u>12.1.1.4 Login</u>	69 - 70
<u>12.1.1.5 Inputs</u>	70 - 74
<u>12.1.1.6 Mapa navegación</u>	75
<u>12.1.1.7 Navegadores</u>	76 - 78
<u>12.1.1.8 Orden</u>	79
<u>12.1.1.9 Buscador</u>	80
<u>12.1.1.10 Paginación</u>	80 - 81
<u>12.1.1.11 Orden, paginación y buscador al introducir datos</u>	81
<u>12.1.2 Login</u>	82 – 85
<u>12.1.3 Email verificación</u>	86
<u>12.1.4 Email bienvenida</u>	87
<u>12.1.5 Email recarga disponible</u>	88
<u>12.1.6 Email stats por debajo de 50%</u>	89
<u>12.1.7 Email stats por debajo de 25%</u>	90
<u>12.1.8 Email has perdido</u>	91
<u>12.1.9 Home</u>	92 - 95
<u>12.1.10 Foro</u>	96
<u>12.1.11 Ranking</u>	97
<u>12.1.12 Videos</u>	98 - 99
<u>12.1.13 Mi cuenta</u>	100 - 101
<u>12.1.14 Juego</u>	102 - 104
<u>12.1.15 Admin cuentas</u>	105 - 106
<u>12.1.16 Admin imágenes</u>	107
<u>12.1.17 Admin personajes</u>	108 - 109
<u>12.1.18 Admin música</u>	110 - 111
<u>12.1.19 Admin videos</u>	112 - 113

<u>12.1.20 Admin cron</u>	<u>114 - 116</u>
<u>12.1.21 Admin partida</u>	<u>117 - 118</u>
<u>12.1.22 Admin juego</u>	<u>119 - 121</u>
<u>12.2 Manual técnico</u>	<u>116 - 133</u>
<u>12.2.1 Consideraciones técnicas</u>	<u>122 - 123</u>
<u>12.2.2 Estructura template</u>	<u>123 - 124</u>
<u>12.2.3 Seguridad</u>	<u>124 - 125</u>
<u>12.2.4 Base de datos (Colecciones)</u>	<u>125 - 126</u>
<u>12.2.5 Recursos</u>	<u>126</u>
<u>12.2.6 Estructura proyecto</u>	<u>127 - 134</u>
<u>12.2.7 Paquetes</u>	<u>135 - 140</u>
<u>12.3 Posibles ampliaciones</u>	<u>141</u>

1- INTRODUCCIÓN

Este documento recoge el trabajo realizado para el módulo de Proyecto del CFGS en Desarrollo de Aplicaciones Web. Este módulo profesional complementa la formación establecida para el resto de los módulos profesionales que integran el título en las funciones de análisis del contexto, diseño del proyecto y organización de la ejecución.

Con este proyecto se pretende conseguir un juego dinámico en el que los usuarios puedan interactuar con el e ir recibiendo recompensas según vayan superando pruebas.

Además se pretende crear un punto de encuentro, donde los usuarios puedan conversar y escuchar música en directo.

El desarrollo de éste proyecto se llevará a cabo en varias fases: estudio de viabilidad, análisis, diseño, implementación en entorno local y pruebas, despliegue en entorno producción y explotación o ejecución.

A continuación se detallan las actividades/tareas/procedimientos de cada una de estas fases.

2- INTRODUCCIÓN EN INGLÉS

This document assembles the work done for the Project module of CFGS in Developing Web Applications. This professional module completes the established training for the rest of the professional modules who includes the title in the functions of the context analysis, project design and organization of the execution.

With this project, the intention is to achieve a dynamic game where the users could interact with it and receive rewards as they go by passing tests.

It is also intended to create a meeting point where the users could talk and listen to music live.

The development of this project it will be carried out in several phases: viability study, analysis, design, implementation in local environment and tests, deployment in production environment and exploitation or execution.

Below are detailed the activities, tasks and procedures of each one of this phases.

3- ESTUDIO DE VIABILIDAD

En esta fase se considera si el proyecto se puede realizar teniendo en cuenta las circunstancias internas y externas de la empresa, las diferentes soluciones posibles y los recursos de los cuales se dispone.

Para ello se hace una valoración del estado actual del sistema y de los requisitos del cliente, se presentará un estudio de soluciones alternativas y la solución elegida por el cliente.

3.1- ESTADO ACTUAL DEL SISTEMA

Actualmente la empresa cuenta con un sistema informático.

Además cuenta con conexión a internet de 50mb y una línea telefónica que utilizan para comunicarse tanto internamente como con los clientes.

3.2- REQUISITOS DEL CLIENTE

El cliente solicita una aplicación de tipo juego online.

Con esta aplicación el cliente pretende explotarla económicamente.

Quiere llevar a cabo el proyecto en 2 fases.

- **Primera fase:**

Aplicación funcionando con los módulos mínimos necesarios para el juego.

Estudio de la aceptación y números de usuarios registrados.

- **Segunda fase:**

En esta fase se ampliarán los módulos en base a la aceptación de la aplicación por parte de los usuarios y el número de los mismos.

Si fuera necesario, se incluirá unos balanceadores de carga.

Ampliar el número de mini juegos.

Explotación económica de la aplicación:

Incluir micro pagos.

Incluir anuncios publicitarios.

Incluir información de discotecas y fiestas.

Incluir tienda online de merchandising.

Incluir tienda online de discos y música.

Incluir tienda online de entradas a discotecas.

- Aplicación inicial:

La aplicación contara con usuarios que tendrán que registrarse.

Habrà usuarios con roles de user y admin.

Los usuarios con roles admin no podrán crearse ni borrarse, serán creados por el sistema.

Los usuarios con rol admin serán los encargados de gestionar la aplicación, estos no se podrán crear ni borrar.

Requiere que sea necesario un email, para poder mandarle avisos al usuario y posibles usos comerciales en la fase 2.

El email ha de ser validado.

Requiere un foro en el que los usuarios puedan hablar entre ellos.

Un apartado donde puedan verse las puntuaciones obtenidas.

Un apartado con música online en directo.

Un apartado para poder modificar su cuenta.

Un apartado con un mini juego.

Stats y objetos para interactuar con el juego principal

Un apartado con los mini juegos (Recompensas) obtenidas por el usuario.

Requiere que la aplicación sea dinámica. Habrá un apartado para que los usuarios con rol admin puedan gestionar las imágenes, usuarios, música, mini juegos y parámetros de la aplicación.

- Aplicación final:

No se especifica nada hasta ver la aceptación de la aplicación.

3.3- POSIBLES SOLUCIONES

Las dos posibles soluciones a la hora de implementarlo escogidas han sido:

- **Caso 1:**

Servidor: Apache, PHP, JS, jQuery y HTML

Cliente: JS, jQuery, Bootstrap, CSS y HTML

- **Caso 2:**

Servidor: nodeJS, mongoDB, Docker, Meteor, JS, jQuery y HTML

Cliente: mongoDB, Meteor, JS, jQuery, Bootstrap, CSS y HTML

3.4- SOLUCIÓN ELEGIDA

Se ha escogido el **caso 2** por los siguientes motivos:

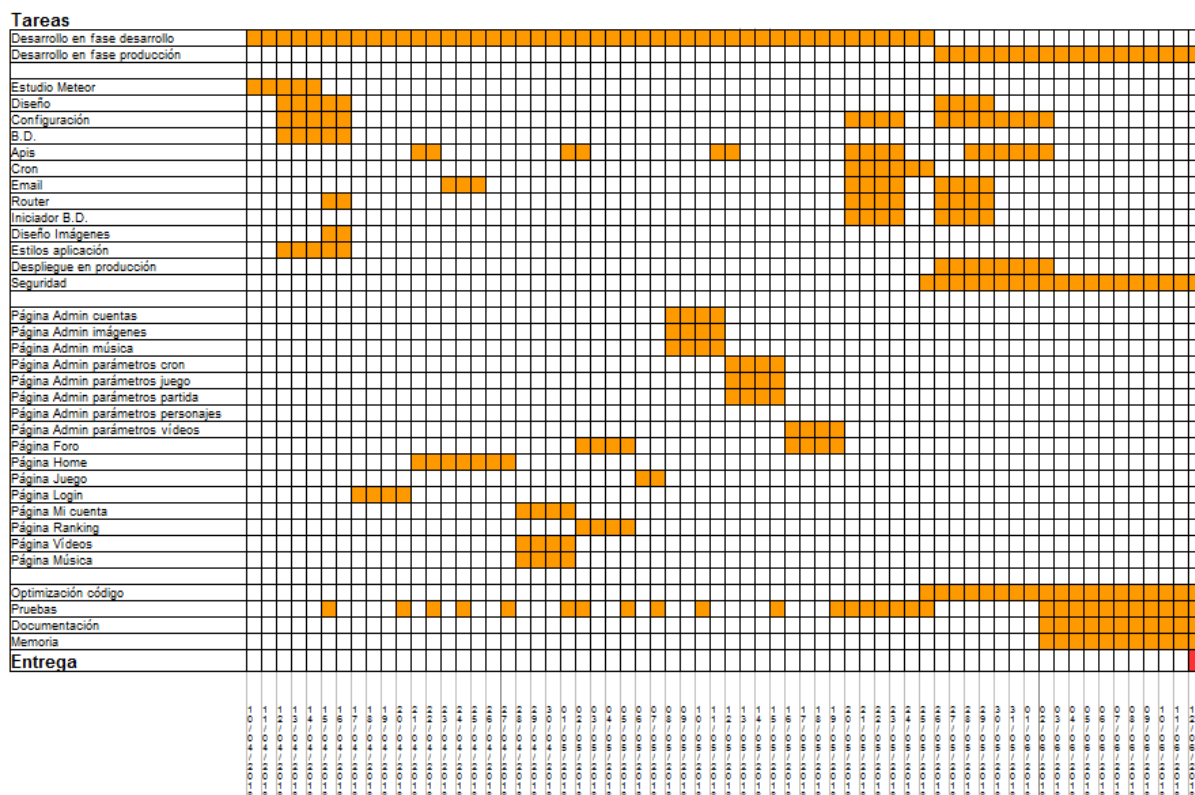
- Meteor ofrece un entorno reactivo aumentando en gran medida la experiencia de usuario comparada con otras metodologías.
- Los tiempos de carga entre páginas son mínimos.
- El tráfico de datos es mínimo una vez cargada la aplicación, ya que trabaja con datos compilados, minificados y cacheados.
- Meteor utiliza el sistema REAL de single page, crea una capa en el navegador que gestiona las peticiones. Con esto se consigue que realmente nunca salgamos de la misma página aunque accedamos a URL de nuestra aplicación desde el navegador.

Con esto se consigue reducir las peticiones al servidor y la carga de tráfico que conlleva.

- Meteor utiliza un sistema modular con componentes, por lo que simplifica el desarrollo y las posibles ampliaciones/modificaciones de la aplicación.
- Aunque Meteor soporta implementación en un servidor Apache convencional, he optado por un servidor nodeJS y Docker, por su facilidad para desplegar la aplicación en fase de producción.
- No se requiere del apagado del servidor para desplegar una actualización, por lo que la aplicación permanece online en todo momento.
- Se despliega como servicio por lo que aunque se reinicie el servidor, la aplicación siempre estará levantada.
- Al ser implementado el servidor con estos lenguajes compilados, se reducen las esperas para el usuario entre páginas y peticiones.
- Al estar implementado en su totalidad en JS se reducen los posibles fallos por incompatibilidades entre lenguajes y se consigue un código más legible y claro.
- Meteor integra las opciones de apache, así que se puede gestionar la seguridad de igual manera o quizás más eficiente que usando Apache.
- Se consigue un tiempo de desarrollo más breve, ya que cuenta con un gran número de paquetes que apenas consumen recursos en comparación con paquetes utilizados en otros frameworks.
- Meteor tiene una gestión excelente de excepciones y errores, si sucede uno la aplicación se reinicia sin dejar sin servicio al usuario.
- Se ha optado por mongoDB para la base de datos por ser una base de datos que consume poco recursos, por la existencia de servidores mongoDB con poco coste económico.
- Meteor gestiona los datos de forma muy eficiente con mongoDB.

- Meteor funciona con un sistema de api.
- Meteor funciona con bases de datos cacheadas en el cliente, siendo mongoDB la mejor opción para ahorrar tráfico y recursos.
- Meteor funciona mediante publicaciones y subscripciones, permite un gran control sobre qué datos tiene acceso cada usuario, consiguiendo un gran grado de seguridad y reduciendo el tráfico.
- Meteor cachea la base de datos en el navegador, usando un sistema de punteros, por lo que reduce en gran medida el tráfico de datos.
- Al usar una base de datos cacheada con punteros, se garantiza que siempre se obtendrán los datos actualizados.
- Se ha escogido CSS y no LESS, porque en Meteor ya se desarrolla de forma modular y devuelve los CSS compilados y minificados.
- Se ha escogido Bootstrap porque agiliza el diseño de la aplicación.
- Para desplegar la aplicación en desarrollo no se necesita acceder al servidor directamente, aumentando la seguridad y comodidad a la hora de desplegar la aplicación.

3.5- PLANIFICACIÓN TEMPORAL DE LAS TAREAS DE L PROYECTO FiestaGochi



3.6- PLANIFICACIÓN DE LOS RECURSOS A UTILIZAR

Para solventar los problemas que plantea el proyecto FiestaGochi se va a realizar una formación previa en Meteor.

Se utilizara el equipo del que ya se dispone.

Se alquilaran los servidores necesarios según los requisitos del proyecto.

Se utilizara un servidor principal para la aplicación y otro para la base datos.

En la futura amplificación se alquilara un espacio en un servidor de imágenes para externalizarlas.

4- ANÁLISIS

En esta fase se establecerán los requisitos del sistema.

4.1- REQUISITOS FUNCIONALES

Es una aplicación con un juego principal y varios mini juegos.

La aplicación debe ser capaz de dar servicio a tantos usuarios entren.

Siempre debe estar online.

Debe tener tiempos mínimos de carga entre páginas

Lograr que los usuarios se diviertan y permanezcan en la aplicación.

Dar la sensación de que la aplicación es dinámica.

Avisos por email.

Puntuaciones.

Parametrizar la aplicación para poder modificarla de forma dinámica.

4.2- REQUISITOS NO FUNCIONALES

Un foro para que los usuarios puedan hablar entre ellos.

Música online en directo.

Tiene que tener un diseño atractivo.

Estar diseñada como prioridad para móvil.

Diseño responsivo.

Estilo con ambiente fiestero o de discoteca, con diseño minimalista y colores oscuros.

Letras manteniendo el diseño de la aplicación.

Debe tener mini juegos que atraigan a usuarios.

Ha de captar el mayor número de usuarios posibles.

Externalizar el mayor número de recursos.

Preparar la aplicación para que se pueda ampliar y modificar fácilmente.

Desarrollar la aplicación para que se pueda cambiar la interfaz gráfica sin necesidad de modificar el código.

Desarrollar la aplicación para poder modificar los parámetros del juego, por parte de los administradores, sin necesidad de tocar la lógica de la aplicación.

Ejecutar toda la lógica posible en el cliente.

5- DISEÑO

En esta fase se realiza una aproximación al diseño tecnológico de la solución.

5.1- ESTRUCTURA DE LA APLICACIÓN

Se trata de una aplicación web diseñada principalmente para dispositivos móviles y secundariamente para el resto de dispositivos.

5.2- COMPONENTES DEL SISTEMA

- Servidor nodeJS.
- Aplicación desplegada en Docker.
- Aplicación desarrollada en Meteor (JS).
- Base de datos mongoDB.
- Protocolo envió HTTPS con SSL.

5.3- HERRAMIENTAS

- Meteor
- nodeJS
- Docker
- jQuery
- Bootstrap
- NPM

5.4- PAQUETES

Paquetes base utilizados por Meteor:

- meteor-base
- mobile-experience
- mongo
- blaze-html-templates

Paquetes base rendimiento utilizados por Meteor:

- standard-minifier-css
- standard-minifier-js
- es5-shim
- ecmaascript
- shell-server

Paquetes base router y renderizado:

- kadora:flow-router
- kadora:blaze-layout

Paquetes incluidos por mí:

- accounts-ui
- accounts-password
- email
- momentjs:moment
- rzymek:moment-locale-es
- mrt:moment-timezone
- mrt:cron-tick
- templating
- session
- mdg:geolocalización
- underscore

Paquetes despliegue en producción:

- mup

6- IMPLEMENTACIÓN

Partiendo del diseño, en esta fase se construye el sistema.

6.1- ENTORNO DE IMPLEMENTACIÓN

El servidor esta implementado en nodeJS, mongoDB y Docker.

La aplicación se desarrolla en Meteor.

El lenguaje utilizado es JS.

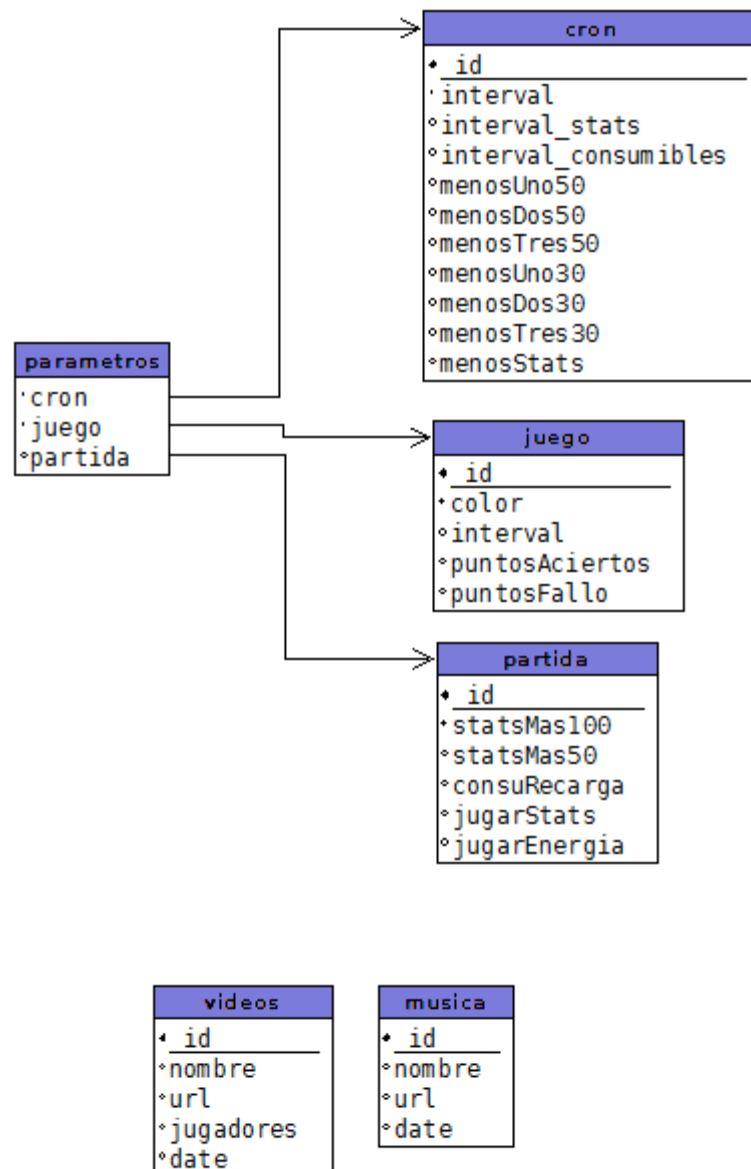
La base de datos esta implementada en mongoDB.

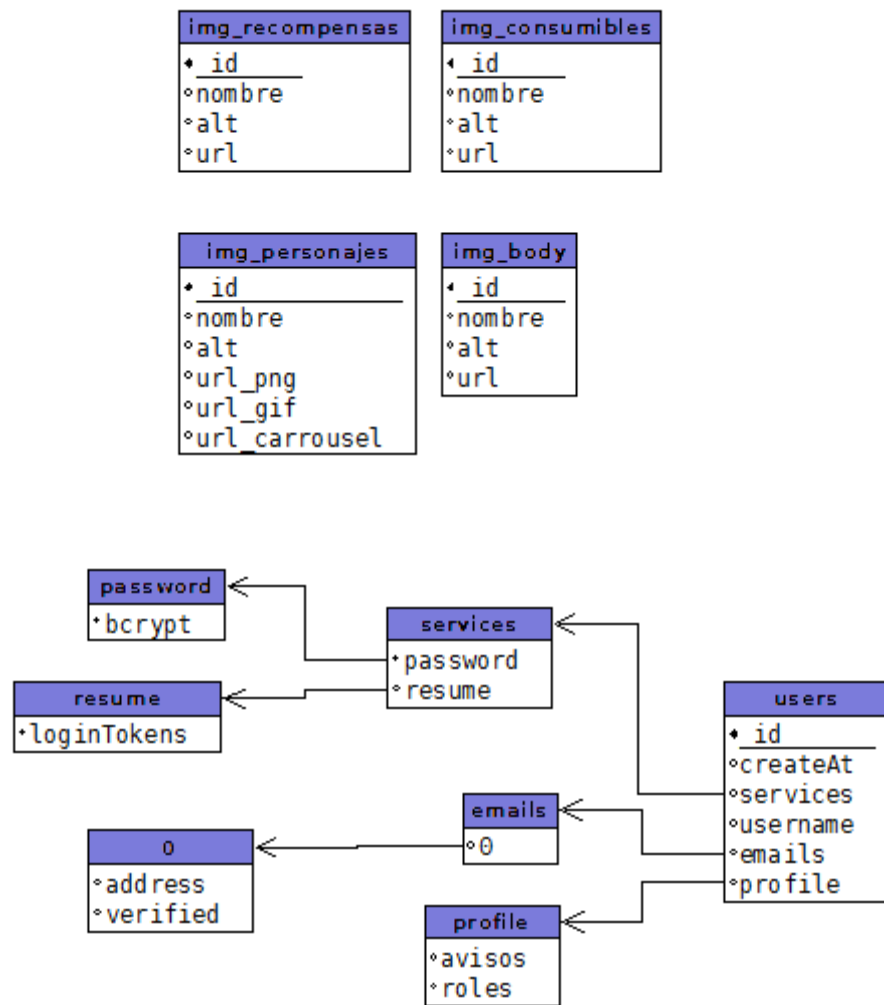
6.2- COLECCIONES Y APIS CREADAS

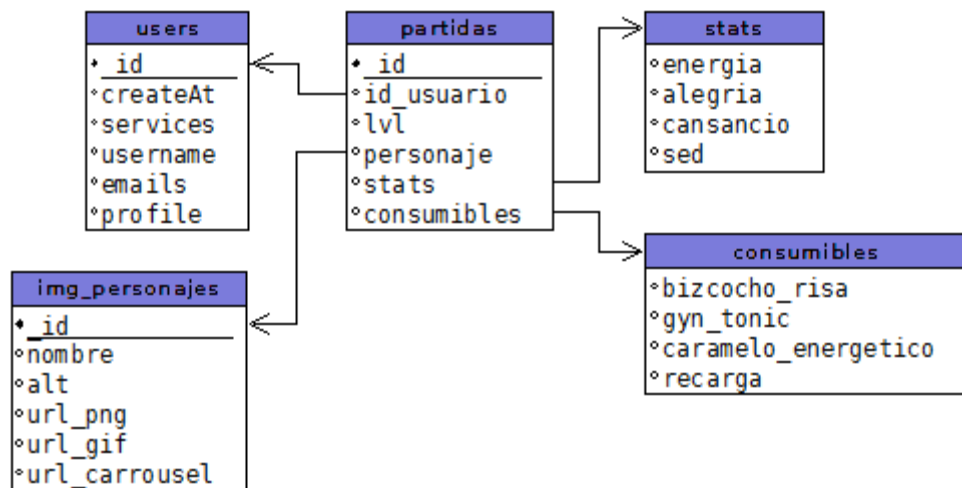
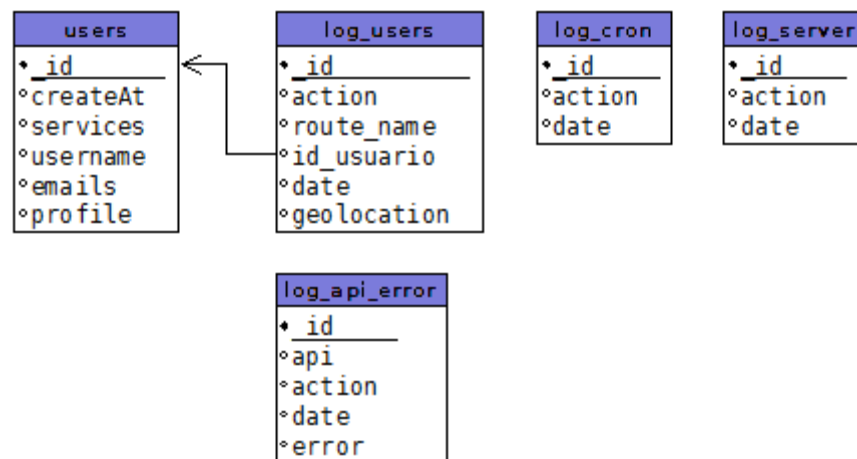
- foro
- img_body
- img_consumibles
- img_personajes
- img_recompensas
- log_api_error
- log_cron
- log_server
- log_users
- musica
- parametros
- partidas
- rankings

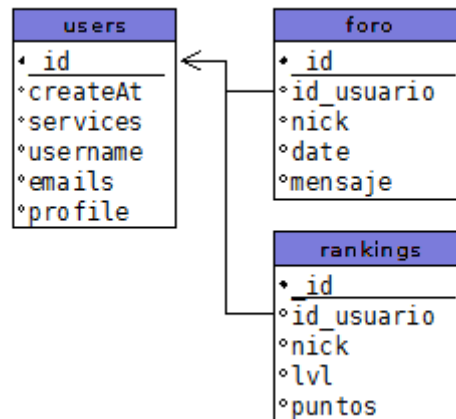
- users
- videos

6.3- DOCUMENTOS Y COLECCIONES









6.4- MÉTODOS API ROL USER

- **foro**
 - foro.insert

- **partidas**
 - partidas.crear
 - partidas.cambiarPj
 - partidas.reset
 - partidas.usarGynTonic
 - partidas.usarbizcocochoRisa
 - partidas.usarCarameloEnergetico
 - partidas.jugar
 - partidas.recarga

- **rankings**
 - ranking.crear

- **users**
 - usuarios.registrar
 - usuarios.borrar
 - usuarios.recuperar
 - usuarios.cambiarNick
 - usuarios.cambiarEmail
 - usuarios.cambiarAvisos

6.5- MÉTODOS API ROL ADMIN

- **img_body**
 - img_body.admin_update
- **img_consumibles**
 - img_gyn.admin_update
 - img_caramelo.admin_update
 - img_bizcocho.admin_update
- **img_personajes**
 - img_personaje.admin_insert
 - img_personaje.admin_remove
 - img_personaje.admin_update
- **img_recompensas**
 - img_recarga.admin_update
 - img_daleGas.admin_update

- **musica**
 - musica.admin_insert
 - musica.admin_remove
 - musica.admin_update

- **parametros**
 - parametros.admin_cron_update
 - parametros.admin_juego_update
 - parametros.admin_partida_update

- **users**
 - users.admin_insert
 - users.admin_remove
 - users.admin_update

- **videos**
 - videos.admin_insert
 - videos.admin_remove
 - videos.admin_update

6.6- PUBLICACIONES

- **foro**
 - foro
 - Todos los documentos
 - Todos los campos

- **img_body**
 - img.body
 - Todos los documentos
 - Todos los campos

- **img_personajes**
 - img.personajes
 - Todos los documentos
 - Todos los campos

- **img_recompensas**
 - img.recompensas
 - Todos los documentos
 - Todos los campos

- **img_recompensas**
 - img.recompensas
 - Todos los documentos
 - Todos los campos

- **musica**
 - musica
 - Todos los documentos
 - Todos los campos

- **musica**

- o musica

- Todos los documentos

- Todos los campos

- **parametros**

- o parametros.cron

- Todos los documentos con _id cron

- Todos los campos

- o parametros.juego

- Todos los documentos con _id juego

- Todos los campos

- o parametros.partida

- Todos los documentos con _id partida

- Todos los campos

- **partidas**

- o partidas

- Todos los documentos con _id del usuario logeado

- Todos los campos

- o partidas.admin

- Todos los documentos

- Todos los campos

- rankings

○ rankings

Todos los documentos

Todos los campos

- users

○ default

Todos los documentos con _id del usuario logeado

Campos username, emails y profile

○ Usuarios.admin

Todos los documentos con roles user

Campos username, emails, profile y createdAa

- videos

○ videos.jug1

Se publica la cantidad de documentos con jugador 1 igual al LVL del jugador logeado

Todos los campos

○ videos.jug2

Se publica la cantidad de documentos con jugador 2 igual al LVL del jugador logeado

Todos los campos

○ videos.jug3

Se publica la cantidad de documentos con jugador 3 igual al LVL del jugador logeado

Todos los campos

- videos.jug4
Se publica la cantidad de documentos con jugador 4 igual al LVL del jugador logeado
Todos los campos
- videos.admin_jug1
Se publican todos los documentos con jugador 1
Todos los campos
- videos. admin_jug2
Se publican todos los documentos con jugador
Todos los campos
- videos. admin_jug3
Se publican todos los documentos con jugador 3
Todos los campos
- videos. admin_jug4
Se publican todos los documentos con jugador 4
Todos los campos

6.7- SUBSCRIPCIONES

- **Página admin_cuentas**
 - musica
 - usuarios.admin
- **Página admin_imagenes**
 - musica
 - partidas

- img.consumibles
- img.recompensas
- img.personajes
- img.body

- **Página admin_musica**

- musica

- **Página admin_parametros_cron**

- musica
- parametros.cron

- **Página admin_parametros_juego**

- musica
- parametros.juego

- **Página admin_parametros_partidas**

- musica
- parametros.partida

- **Página admin_personajes**

- musica
- img.personajes

- **Página admin_videos**
 - musica
 - videos.admin_jug1
 - videos.admin_jug2
 - videos.admin_jug3
 - videos.admin_jug4

- **Página foro**
 - musica
 - foro

- **Página home**
 - musica
 - partidas
 - img.consumibles
 - img.compensas
 - img.personajes

- **Página juego**
 - musica
 - partidas
 - img.personajes
 - parametros.juego

- **Página login**
 - música

- **Página mi_cuenta**
 - musica
 - partidas
 - img.personajes

- **Página ranking**
 - musica
 - rankings

- **Página videos**
 - musica
 - videos.jug1
 - videos.jug2
 - videos.jug3
 - videos.jug4

6.8- Carga de datos

Se realizan mediante la carpeta startup, es lo primero en ejecutarse al levantarse el servidor.

La carga de datos se realiza mediante los scripts
/imports/startup/server/ini_database.js

Si no hay documentos, crea en cada colección los documentos necesarios para el desarrollo de la aplicación.

En este punto la inicialización de colecciones más significativas serian:

- **Parametros**

Necesaria para ejecutar la aplicación.

- **Users**

Crea los usuarios admin, estos no son posibles crear ni borrar a través de la aplicación.

El esto de inicializaciones son importantes, pero la aplicación podría desplegarse y utilizarse sin problema.

6.9- FICHEROS DE CONFIGURACIÓN SERVIDOR

- **moment**

/imports/startup/server/config.js

Asigna locale es y timezone Europe/Madrid

- **Accounts**

/imports/startup/server/config.js

Configura la duración de la sesión

Activa forbiddenPass

- **Email**

/imports/startup/server/config.js

Se configure la variable de Sistema que enlace con Gmail vía SMTPS.

En ella se configura la cuenta, contraseña y puerto por el que conecta a Gmail.

Configura el siteName de las templates de email.

Configura el from de las templates de email.

Configura subject y text de las templetas de verifyEmail y resetPassword.

- **Cron_clean_db**

/imports/startup/server/cron_clean_db.js

Inicializa el cron encargado de las tareas de mantenimiento y limpieza en la base de datos.

- **Cron_recarga**

/imports/startup/server/cron_recarga.js

Inicializa el cron encargado de actualizar el atributo recarga de los documentos de la colección partidas.

Se inicializa con los parámetros guardados en la base de datos.

- **Cron_stats**

/imports/startup/server/cron_stats.js

Inicializa el cron encargado de actualizar los stats de los documentos de la colección partidas.

Se inicializa con los parámetros guardados en la base de datos

- **Cron_users**

/imports/startup/server/cron_stats.js

Inicializa el cron encargado de comprobar si los emails están verificados.

Inicializa el cron encargado de volver a enviar los emails de verificación.

- **Triggers**

/imports/startup/server/triggers.js

- o **Triger onCreateUser**

Cuando se crea un usuario, crea una partida asociada a él.

Si ese usuario tiene el rol admin, le valida el email.

6.10- FICHEROS DE CONFIGURACIÓN CLIENTE

- **VerificationLink**

/imports/startup/client/config.js

Elimina evento para crearlo con otros parámetros.

- **Menú contextual**

/imports/startup/client/config.js

Desactiva menú contextual y muestra mensaje advertencia al intentar abrirlo.

- **moment**

/imports/startup/client/config.js

Asigna locale es y timezone Europe/Madrid

- **Rutas comunes**

/imports/startup/client/routes_common.js

En este fichero es donde se configuran las rutas utilizadas tanto para usuarios con rol user como admin.

Se utiliza un sistema de templates dinámicas.

El encargado de renderizadas es BlazeLayout.

FlowRouter recibe la URL y da la orden a BlazeLayout de que template ha de renderizar en la layout body.

- Rutas admin

/imports/startup/client/routes_admin.js

En este fichero es donde se configuran las rutas utilizadas tanto para usuarios con rol admin.

Se utiliza un sistema de templates dinámicas.

El encargado de renderizadas es BlazeLayout.

FlowRouter recibe la URL y da la orden a BlazeLayout de que template ha de renderizar en la layout body.

- Triggers

/imports/startup/client/routes_triggers.js

○ Trigger onLogin

Cuando un usuario logea, desloguea el resto de cuentas activas con el mismo id.

Valida que el email este verificado.

Si no lo está cierra la sesión y muestra advertencia.

Si lo está lleva a la página home.

○ Trigger onLogout

Cuando un usuario desloguea, se carga la página login.

- Trigger onEmailVerificationLink

Cuando un usuario hace click en el enlace enviado al email para verificar el email.

Se verifica esa dirección y se manda un email de bienvenida.

- Trigger exit

Cuando un usuario sale de una página.

Se guarda un log.

- Trigger enter

Cuando un usuario entra de una página.

Se guarda un log.

Se inicializan las variables de control de las subscripciones.

Se valida si esta logeado, si no lo esta se redirige a login.

Se valida que su email este verificado, si no lo esta se desloguea.

Si la página es juego, se valida que su energía sea superior a 80, sino se redirige a home.

Si la página es para usuarios con roles admin, se valida que el usuario tenga dicho rol, si no lo tiene se redirige a home.

- Trigger notFound

Si la página no existe, se redirige a home.

6.11- FICHEROS DE CONFIGURACIÓN DESPLIEGUE PRODUCCIÓN

`/.deploy/mup.js`

En este fichero se configura el entorno en producción de la aplicación.

Apartados:

- server

Contiene el host, user y pass del servidor.

- app

Contiene el nombre y ruta local de la aplicación.

Contiene en que servidor se va a desplegar.

Si admite opciones.

Las variables de entorno, donde especificaremos la dirección del servidor con la que queremos que sirva nuestra aplicación, y en mi caso la URL al servidor mongoDB con los datos de conexión.

- Docker:

El paquete y la versión con la que vamos a desplegar la aplicación.

- proxy

Contiene el dominio para el SSL, el SSL con el email al que va asociado y la opción para que utilice siempre HTTPS.

6.12- CONFIGURACIONES REALIZADAS EN EL SISTEMA

Desarrollo local:

- Instalación nodeJS.
- Instalación NPM.
- Instalación Meteor.
- Crear proyecto y quitar paquetes innecesarios.
- Quitar paquete autopublish.
- Agregar a NPM los paquetes instalados en Meteor.
- Instalar bcrypt a través de NPM.
- Instalar NPM moment
- Instalar NPM moment-timezone
- Crear proyecto privado en github con la aplicación.

En este punto para desarrollo en local no es necesario configurar el apartado de .deploy.

No se necesita configurar el apartado de base de datos, Meteor en local crea una base de datos local.

Desarrollo producción:

- En servidor aplicación:
 - Registro en DigitalOcean.
 - Crear un droplet (servidor) en DigitalOcean.
 - Enlazar el dominio utilizado con el servidor creado.
 - Crear usuario.
 - Hacer un update y upgrade del sistema.
 - Configurar timezone servidor Europa/Madrid.
 - Configurar locale es.
 - Instalar nodeJS.
 - Instalar NPM.
 - Instalar NPM moment
 - Instalar NPM moment-timezone
 - Instalar Meteor.
 - Instalar el paquete mup a través de NPM.

- En servidor base datos:
 - Registro en mLab.
 - Crear base de datos.
 - Crear usuario.

- En local:
 - Instalar el paquete mup a través de NPM
 - Configurar los ficheros de configuración de la aplicación.
 - Configurar los ficheros de configuración para el despliegue en producción.

- Setup de nuestra aplicación.
- Si el setup ha sido correcto, se hará un deploy, con esto se compilará y subirá el contenedor Docker con la aplicación al servidor.
- Si en un futuro se quiere actualizar la aplicación, solo hará falta setup y deploy. De esta forma se actualizará la aplicación sin tener que dejar de dar servicio.

6.13- IMPLEMENTACIONES DE CÓDIGO REALIZADAS

Sin lugar a duda los aspectos más destacables que tiene una aplicación desarrollada en Meteor podrían ser:

- Como cachea la aplicación en el navegador.
- Como cachea la base de datos en el navegador, haciendo uso de punteros.
- El entorno reactivo con el que trabaja Meteor.

Estos puntos fuertes podemos verlos en los objetos Session y en los helpers de las templates.

Donde vemos que sin hacer uso de llamadas Ajax siempre obtiene los datos actualizados.

Un ejemplo sería:

En la template Status_bar_left

El helper stats

```
Template.Status_bar_left.helpers({
  Stats(){
    If(Session.get('partidasSub') === "ready" && Meteor.user()){
      return Partidas.findOne({_id: Meteor.userId()}).stats;
    }
  }
});
```

Esta función realiza lo siguiente:

Primero evalúa si la suscripción esta lista.

Cuando esta lista devuelve el documento stats de la partida del usuario logeado.

Al trabajar en un entorno reactivo con datos cacheados.

Este helper siempre devolverá los datos actualizados.

Como se observa, se reduce el tráfico, ya que no hay que estar haciendo constantes llamadas de Ajax para mantener la vista actualizada.

Reduce la carga en el servidor, al no estar recibiendo peticiones constantemente.

El código es mucho más claro y legible,

7- PRUEBAS

Son muchas las pruebas que pueden realizarse en un proyecto, para eliminar los posibles errores y garantizar su correcto funcionamiento. Los casos de prueba establecen las condiciones/variables que permitirán determinar si los requisitos establecidos se cumplen o no.

A continuación se detallan algunos de los casos de prueba que se ejecutarán para comprobar la correcta construcción de este proyecto.

7.1- CASOS DE PRUEBAS

- 13/04/2018

- Pruebo que se ejecuta la aplicación en local con una estructura vacía.
- **Incidencias:**
 - Ninguna

- 20/04/2018

- Pruebo el componente login
 - Compruebo las siguientes funcionalidades:
 - Login
 - Registro
 - Recuperar contraseña
 - Realizo las siguientes pruebas:
 - No deja logear con un usuario que no existe.

- No deja registrarse con un Nick o email que ya está registrado.
 - Solo devuelve la contraseña del email introducido y al email introducido.
 - Hago una primera prueba de permisos, para comprobar que solo se puede escribir en la base de datos a través del servidor.
 - Hago una primera prueba sobre las publicaciones/subscripciones para ver si está filtrando bien los datos que están accesibles desde el cliente.
 - Hago una primera prueba del enrutador y los triggers.
- Pruebo si inicializa correctamente la base de datos.
 - **Incidencias:**
 - Dejaba escribir desde el lado del cliente.
 - No salta el trigger si no hay usuario.
 - **Soluciones:**
 - Se configura para que no se pueda escribir desde el cliente.
 - Se retrasa la función que evalúa si hay usuario, ya que la colección de usuarios conectados tarda un poco en estar lista.
- **22/04/2018**
- Pruebo los componentes del navegador
 - Compruebo que los enlaces funcionan correctamente.

- Compruebo el componente música, compruebo si enlaza con la api, si recibe música, si se puede escuchar correctamente y si se puede cambiar de emisora o parar/play la música.
- Compruebo el componente usuario, si puedo desbloquear, si se actualiza el Nick al cambiarlo.

- **Incidencias:**

- Ninguna

- **24/04/2018**

- Pruebo los componentes del navegador
 - Compruebo si se muestran los personajes.
- Pruebo los componentes de home
 - Compruebo si se muestran los stats.
 - Compruebo si se muestran los consumibles.
 - Compruebo si se muestra la recarga y recompensa.
 - Compruebo si se actualiza la vista cuando cambio la base de datos.
- **Incidencias:**
 - Muestra errores al iniciar la aplicación en los helpers con findOne.
- **Soluciones:**
 - Se agregan variables de control para controlar cuando están listas las subscripciones.

- 27/04/2018

- Pruebo los componentes del navegador
 - Vuelvo a comprobar si hace los enlaces a las rutas con las nuevas configuraciones del router.
 - Compruebo si el router captura bien los eventos.
 - Compruebo si cambia de navegador de la página login al resto de páginas.
 - Vuelvo a comprobar si actualiza los campos probados el 24/04/2018.
 - Compruebo el diseño, si queda conforme al diseño y si cambia según el tamaño.
- **Incidencias:**
 - Ninguna

- 01/05/2018 a 02/05/2018

- Pruebo los componentes de la página videos.
 - Compruebo que muestra los videos.
 - Compruebo que solo están accesibles los videos correspondientes a su LVL.
 - Compruebo que los se muestran bien los videos.
 - Compruebo si cambian al cambiar la base de datos.
 - Compruebo el diseño, si queda conforme al diseño y si cambia según el tamaño.
- Pruebo los componentes de la página mi_cuenta.
 - Compruebo que muestra los datos del usuario logeado.

- Compruebo si solo están accesibles sus datos.
- Compruebo que campos están accesibles.
- Compruebo que solo se puede escribir desde el servidor.
- Compruebo si cambian al cambiar la base de datos.
- Compruebo el diseño, si queda conforme al diseño y si cambia según el tamaño.

- **Incidencias:**

- Ninguna.

- **05/05/2018**

- Pruebo los componentes de la página foro.
 - Compruebo que muestra los mensajes.
 - Compruebo que los se muestran bien los mensajes.
 - Compruebo si cambian al cambiar la base de datos.
 - Compruebo el diseño, si queda conforme al diseño y si cambia según el tamaño.
- Pruebo los componentes de la página ranking.
 - Compruebo que muestra las puntuaciones.
 - Compruebo que los se muestran bien los videos.
 - Compruebo si cambian al cambiar la base de datos.
 - Compruebo el diseño, si queda conforme al diseño y si cambia según el tamaño.

- **Incidencias:**

- Ninguna

- 10/05/2018

- Pruebo los componentes buscador y paginación de los componentes ya desarrollados.
 - Compruebo que hacen los filtrados y ordenados correctamente.
 - Compruebo el diseño, si queda conforme al diseño y si cambia según el tamaño.
- Pruebo las APIS.
 - Pruebo enviado datos erróneos a mano, para comprobar que hacen los filtrados correctamente.
 - Compruebo que capturan bien las excepciones.
- **Incidencias:**
 - No ordena ni filtra bien.
- **Soluciones:**
 - Reescribir el método del helper encargado de devolver los datos.
 - Añadir variables de control a session.

- 15/05/2018

- Pruebo los componentes de las paginas admin desarrolladas hasta el momento.
- Compruebo que cambian los datos correctamente, que solo son accesibles para usuarios con rol admin
- Que los datos se actualizan de forma correcta.
- Solo se puede escribir desde el servidor.

- **Incidencias:**
 - Ninguna.

- **20/05/2018 a 25/05/2018**

- Pruebo todo el proyecto en local.
- Compruebo las rutas, que enrutan y capturan bien los eventos.
- Compruebo que filtra correctamente los datos a los que se tiene acceso.
- Compruebo que solo se puede escribir desde el servidor.
- Compruebo que solo se tiene acceso a los datos definidos.
- Compruebo el diseño general de la página, que todos los componentes están en “armonía”.
- Compruebo los crons.
- Compruebo los envíos de emails, si los envía correctamente y si el diseño es correcto.
- Reviso todas las rutas, para comprobar que no hay ninguna caída.
- Reviso bien las colecciones de log, para comprobar que todo ha funcionado correctamente y no ha habido ningún error.
- Compruebo el juego:
 - Si solo se puede jugar si se cumplen los requisitos
 - Si captura bien los eventos
 - Si suma o resta bien los puntos
 - El aspecto del juego
 - Si se ejecuta correctamente

- Modifico los parámetros de la aplicación, para ver si se actualiza correctamente la aplicación.

- **Incidencias:**

- Ninguna.

- **02/06/2018 a 12/06/2018**

- Compruebo toda la aplicación al completo en el servidor.
- Compruebo si se reinicia al reiniciar el servidor.
- Compruebo si se pierde el servicio al desplegarla de nuevo.
- Compruebo que enlaza correctamente con el servidor de mLab para la base de datos.

- **Incidencias:**

- Ninguna.

8- EXPLOTACIÓN

La implantación es la fase más crítica del proyecto ya que el sistema entra en producción, es decir opera en un entorno real, con usuarios reales.

8.1- PLANIFICACIÓN

1- Antes de desplegar en producción:

- Se eliminaran paquetes y recursos innecesarios.
- Se externalizaran todos los paquetes y recursos posibles.
- Se revisara el código.
 - Se dejara lo más limpio y claro posible.
 - Se eliminaran fragmentos innecesarios ej `console.log()`.
 - Se eliminaran cuentas y contraseñas.
- Se actualizara Meteor y los paquetes instalados.

2- Registrar y crear emails.

3- La aplicación se desplegara inicialmente en un único servidor

4- DigitalOcean con la base de datos en mLab.

5- Registrar y crear servidor en DigitalOcean, el escogido es Ubuntu 18.04 con 1gb de RAM y 25gb de disco duro.

6- Actualizar servidor.

7- Crear usuario en servidor.

8- Instalar nodeJS.

9- Instalar NPM.

10-Instalar Meteor.

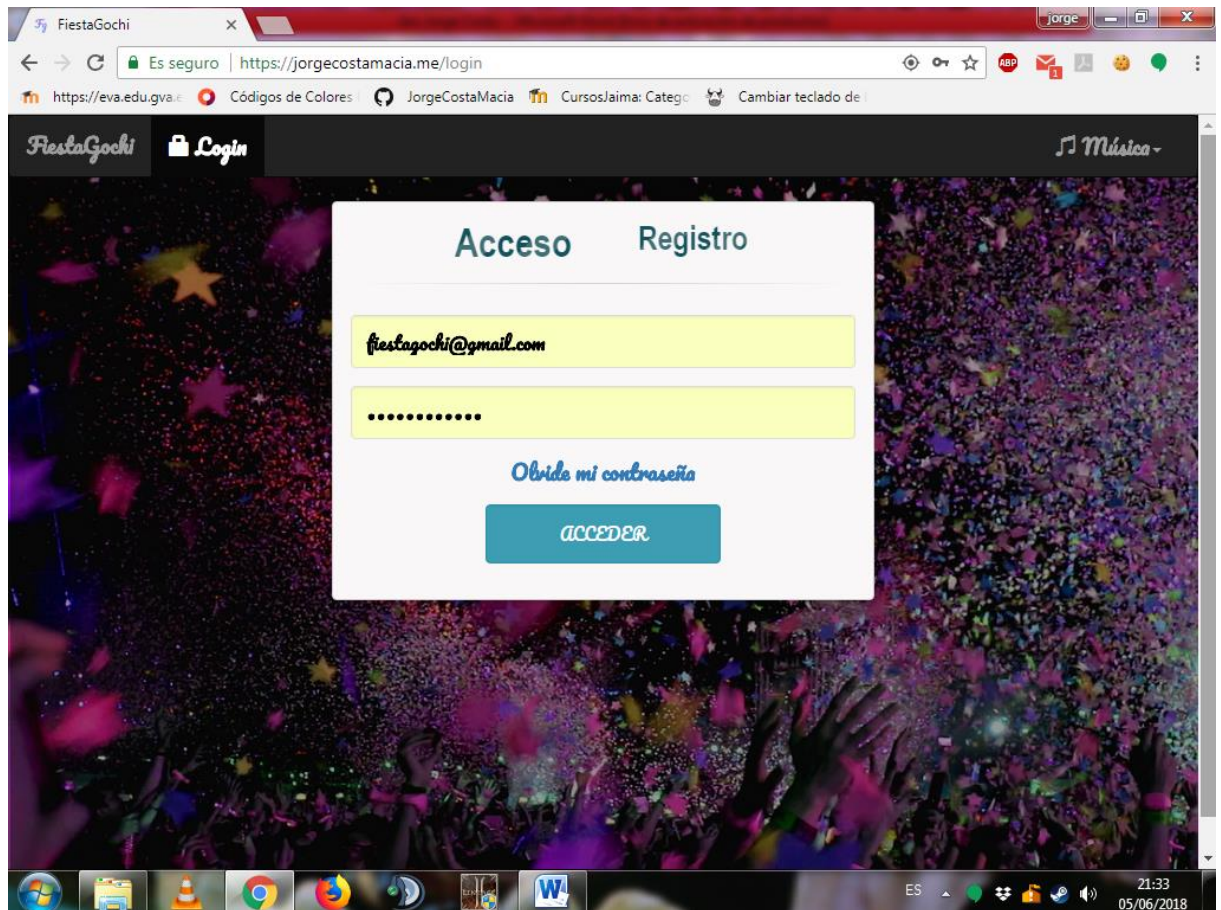
- 11-Instalar mongoDB.
- 12-Registrar y crear la base de datos en mLab.
- 13-Crear usuario para la base datos.
- 14-Configurar el apartado deploy.
- 15-Desplegar la aplicación.
- 16-Comprobar la estabilidad y seguridad.
- 17-Si el servidor DigitalOcean no fuera suficiente. Se estudiaría la ampliación más apropiada.
Se estudiaría si crear otro servidor espejo y balancear la carga con NGNIX o aumentar el servidor.
- 18-Si la base de datos en mLab no fuera suficiente. Se estudiaría la ampliación más apropiada.
Se estudiaría si aumentar la base datos contratada, contratar un nuevo servidor y separar las colecciones, o disminuir la cantidad tope de documentos que no elimina el cron clean_db.

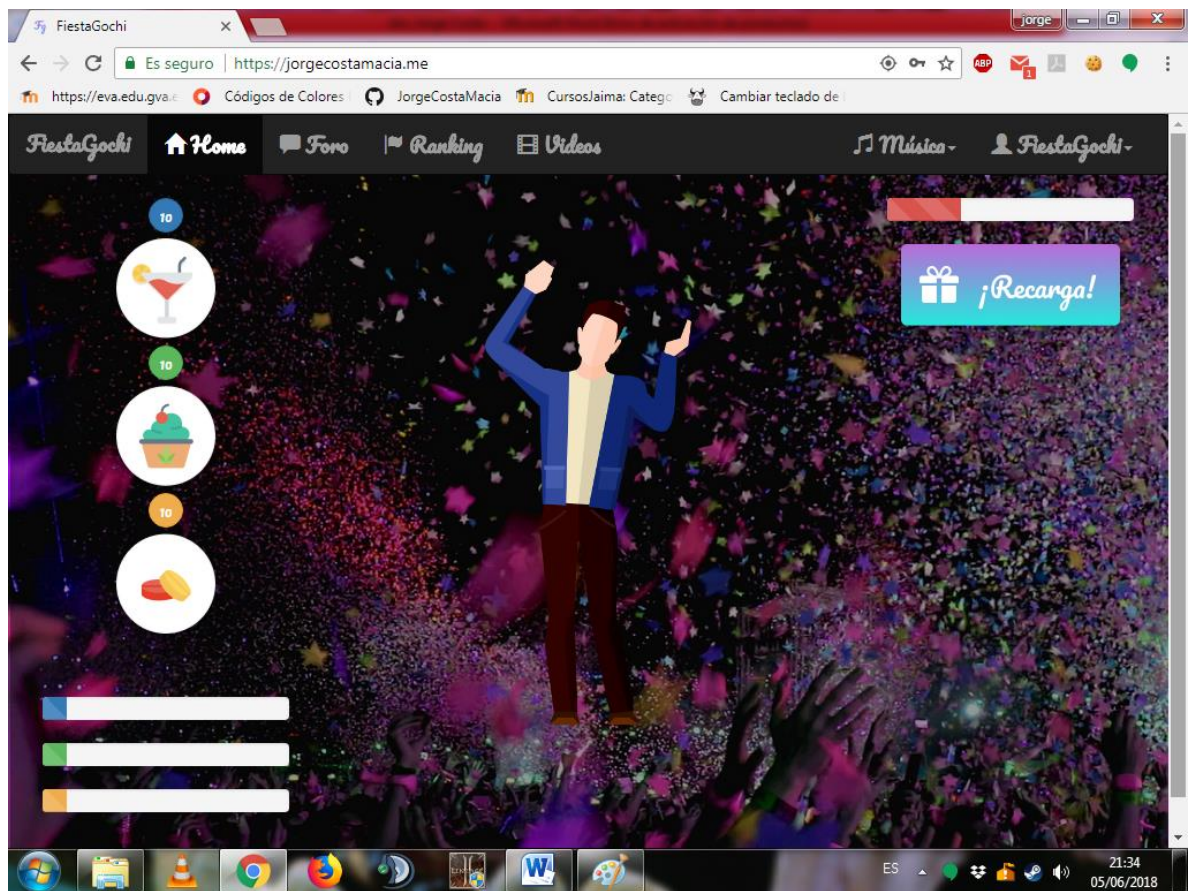
8.2- PLAN DE FORMACIÓN

Al verificar el email, se manda un email con las instrucciones de la aplicación. Existen varios emails corporativos donde los usuarios pueden exponer sus dudas.

Habrà un manual de usuario con las instrucciones, dicho manual estará disponible para los administradores.

8.3- IMPLANTACIÓN PROPIAMENTE DICHA





8.4- PRUEBAS DE IMPLANTACIÓN

Se repitieron todas las pruebas sin incidencias.

9- DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN

A lo largo del ciclo de vida del proyecto se producirán cambios e incidencias que deberán controlarse y registrarse.

9.1- Incidencias

Se ha utilizado un registro de log de toda la actividad de la aplicación.

En el caso de que suceda un error se crea el siguiente documento:

- El api donde se produce
- La acción que lo produjo
- Fecha y hora
- Error

Este documento se guarda en log_api_error.

Además se manda el mismo doc por a suportfiestagochi@gmail.com

El encargado de mantenimiento podrá ver y analizar el error, o bien a través de la base de datos o a través del email.

Al solucionarla solamente tendrá que volverá desplegar la aplicación.

Estas incidencias son visibles a través de la web que proporciona el proveedor de mLab o del email suportfiestagochi@gmail.com

La base de datos también es accesible través de un gestor mongoDB, del propio Shell o a través de un IDE.

mLab

```
{
  "_id": "5zXQprtAXKkehk3Wn",
  "api": "none",
  "action": "start_server",
  "date": "2018-06-04, 21:24:53",
  "error": "none"
}

{
  "_id": "eCMQjXnoTuEYPMBjr",
  "api": "usuarios",
  "action": "registrar",
  "date": "2018-06-05, 19:41:30",
  "error": {
    "isClientSafe": true,
    "error": 403,
    "reason": "Username already exists.",
    "message": "Username already exists. [403]",
    "errorType": "Meteor.Error"
  }
}
```

mongoDB Compass

fiestagochi.log_api_error

DOCUMENTS 2 TOTAL SIZE 736B AVG. SIZE 368B INDEXES 1 TOTAL SIZE 8.0KB AVG. SIZE 8.0KB

Documents Schema Explain Plan Indexes Validation

FILTER { field: 'value' } OPTIONS FIND

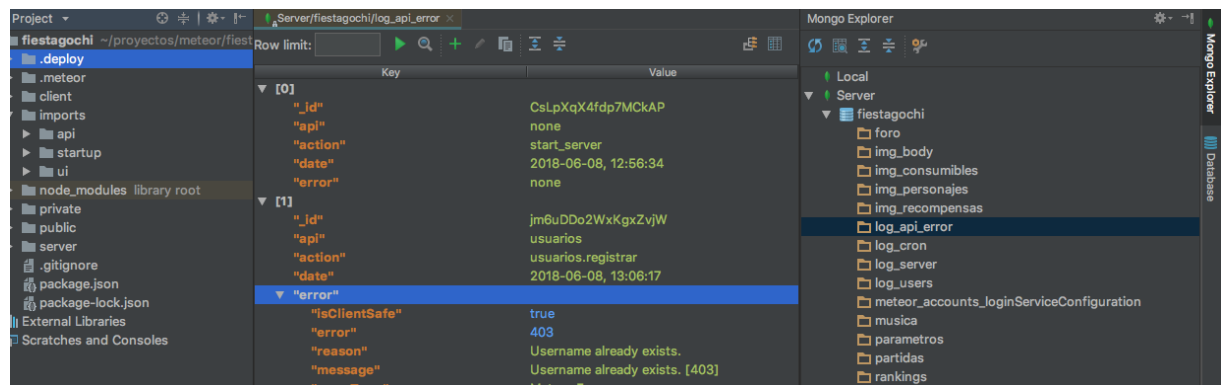
INSERT DOCUMENT VIEW LIST TABLE

Displaying documents 1 - 2 of 2

```
{
  "_id": "CsLpXqX4fdp7MckAP",
  "api": "none",
  "action": "start_server",
  "date": "2018-06-08, 12:56:34",
  "error": "none"
}

{
  "_id": "jm6u0D0zWxKgxZvjW",
  "api": "usuarios",
  "action": "usuarios.registrar",
  "date": "2018-06-08, 13:06:17",
  "error": {
    "isClientSafe": true,
    "error": 403,
    "reason": "Username already exists.",
    "message": "Username already exists. [403]",
    "errorType": "Meteor.Error"
  }
}
```

PhpStorm



Email

[FiestaGochi](#)

Se ha producido el error con `_id`
`M6XySajatrKK3ACdS`

En la api usuarios

En la fecha 2018-06-05, 23:10:48

Error:

isClientSafe

true

error

403

reason

Username already exists.

details

undefined

message

Username already exists. [403]

errorType

Meteor.Error

9.2- Control versiones

La aplicación ha sido desarrollada en github, en un repositorio privado.

Actualmente solo ha sido desarrollada por Jorge Costa Maciá por lo que ha sido desarrollada en su totalidad en la rama master.

Al final el proyecto se creara un repositorio nuevo con el proyecto.

En este nuevo proyecto no se incluirán los ficheros de cuentas ni deploy.

Estos ficheros han sido incluidos en fase de desarrollo por incluir las cuentas y pruebas iniciales.

Si fuera necesario, se daría permisos al repositorio para modificaciones.

Solo tendría acceso a las cuentas y al apartado deploy la persona responsable de dicho proyecto.

88 commits 1 branch 0 releases 0 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

jorgecostamacia Update trigger create user Latest commit 511526b an hour ago

.deploy	add ssl, https y geolocalizacion	6 days ago
.meteor	Update version	an hour ago
client	update	12 days ago
imports	Update trigger create user	an hour ago
private	final updates	a month ago
public/img	finish project	27 days ago
server	Initial commit	a month ago
.gitignore	Update	5 days ago
package-lock.json	fix error moment and add msj success	21 hours ago
package.json	fix error moment and add msj success	21 hours ago

Add a README with an overview of your project. Add a README

Version Control: Local Changes Log Console

Branch: All User: All Date: All Paths: All

Commit Message	Author	Date
Update trigger create user	jorgecostamacia	8/6/18 15:18
Update version	jorgecostamacia	8/6/18 14:58
fix error moment	jorgecostamacia	8/6/18 12:31
fix error moment	jorgecostamacia	8/6/18 12:05
fix error moment and add msj success	jorgecostamacia	7/6/18 19:35
fix error crons stats recarga	jorgecostamacia	7/6/18 16:34
update colores	jorgecostamacia	6/6/18 0:03
control errores	jorgecostamacia	5/6/18 23:42
control errores	jorgecostamacia	5/6/18 23:31
control errores	jorgecostamacia	5/6/18 23:25

10- CONCLUSIONES

Ha sido un proyecto desarrollado en Meteor.

El entorno reactivo hace de este framework la mejor opción para aplicaciones que requieran mostrar datos actualizados de la base de datos.

Otra de las ventajas de este framework, tiene unos tiempos de carga mínimos, aumentando en gran medida la experiencia de usuario.

Se han cumplido todos los objetivos planteados en la propuesta, incluido las opcionales.

Se han agregado funcionalidades no planteadas en la propuesta, como:

- Docker
- Bcrypt
- SSL y HTTPS en Docker.
- Se ha desarrollado un router en JS.
- Uso de templates.
- Envío de emails
- Templates personalizadas para emails
- Servidor mongoDB
- Verificación de emails.
- Recuperación de contraseña.
- 4 Crons
- Geolocalización
- Diferencias colecciones log para monitorizar la aplicación.

- Envíos de emails con los errores.
- Roles en los usuarios,
- Música online en directo.
- La aplicación esta parametrizada lo máximo posible, es posible cambiar las imágenes, colores, agregar música... desde la propia aplicación.
- Se ha externalizado lo máximo posible, para reducir la cara del servidor.

Ejemplos:

- o La música la obtiene el cliente a través de la api interna de locaFM, no se almacena.
- o Los videos de recompensas los obtiene el cliente a través de un iframe, aprovechando el sistema que tiene Youtube para este uso.
- o Las imágenes aunque en principio las obtiene del servidor, las obtiene a través de un GET, está desarrollado de esta forma para en un futuro migrar las imágenes a un servidor tipo Amazon.

En un principio los estilos fueron diseñados en LESS, pero se refactorizo a CSS.

Se cambió a CSS porque la aplicación no tenía grandes stylesheets, CSS era la mejor opción para reducir el uso de recursos.

Además Meteor compila toda la aplicación y la aplicación integra un minificador de CSS por lo que no era necesario usar LESS.

El uso de este framework me ha servido personalmente para:

- Conocer bien las nuevas funciones de ES6 ya que Meteor trabaja con esa versión de JS.
- Conocer bien como trabajar con una base de datos mongo.
- Cambiar mi forma de ver y pensar los objetos en JS.
- Aprender a desplegar en Docker y como configurar el servidor.
- Aprender a iniciar y configurar un servidor en nodeJS con NPM.
- Aprender Meteor.

11- FUENTES

11.1- Legislación DAW:

Enseñanzas mínimas: **Real Decreto 686/2010, de 20 de mayo (BOE 12/06/2010)**

http://pdf/IFCS03/titulo/RD20100686_TS_Desarrollo_Aplicaciones_Web.pdf

Currículo: **D. 1/2011, de 13 de enero (BOCM 31/01/2011)**

http://pdf/IFCS03/curriculo/D20110001_TS_Desarrollo_Aplicaciones_Web.pdf

11.2- Fuentes utilizadas

- <https://www.meteor.com/>
- <http://meteor-up.com/docs.html>
- <https://atmospherejs.com>
- <https://bootsnipp.com/>
- <https://www.digitalocean.com/community/tutorials/configuracion-inicial-del-servidor-en-ubuntu-16-04-es>
- <https://mlab.com/welcome/>
- <https://nodejs.org/es/>
- <https://github.com>

12- ANEXOS

12.1- MANUAL DE USUARIO

12.1.1- CONSIDERACIONES TÉCNICAS

La aplicación consiste en un juego con pruebas y recompensas.
También incluye un foro, ranking y música.

Cada partida contara con un personaje, estados y consumibles.

Con el paso del tiempo los estados irán disminuyendo, se tendrá que consumir los objetos para incrementar los estados.

Si el estado subidón llega a 0 habrá perdido y se reiniciara la partida.

Cada cierto tiempo el sistema generara una recarga que se podrá utilizar para conseguir consumibles.

Cada cierto tiempo el sistema actualizara los estados.

(Parece complicado, pero en el apartado de home viene explicado junto a imágenes)

El aspecto puede variar en función del dispositivo que se utilice.

El manual explica el funcionamiento en un ordenador, por ser disponer de una pantalla mayor y ser fácil de identificar los apartados.

Durante el manual se emitirán cifras, ya que estas pueden ser variadas por los administradores.

Ejemplo:

La cantidad que disminuyen los estados al consumir los objetos.

12.1.1.1- EMAILS Y RECURSOS CORPORATIVOS

La aplicación cuenta con los siguientes emails:

- fiestagochi@gmail.com
- suportfiestagochi@gmail.com
- stafffiestagochi@gmail.com

Todos ellos contarán con una cuenta creada en la aplicación con roles admin.

El email suportfiestagochi@gmail.com será utilizado para las incidencias, en el caso de haber algún error el sistema mandará un email a dicha dirección con los datos del error sucedido.

Contará con una cuenta y servidor en DigitalOcean y mLab.

12.1.1.2- ACTIVIDAD

La actividad de la aplicación será monitorizada y guardada en las colecciones log, las cuales serán accesibles a través de medios externos por los usuarios autorizados.

12.1.1.3- USUARIOS Y PERMISOS

Dicho juego cuenta con usuarios de 2 tipos con unos permisos determinados.

- **Usuarios sin registrar:**
 - Escuchar música.
 - Registrarse.

- **Usuarios rol user podrán:**
 - Jugar al juego principal.
 - Conversar en el foro.
 - Ver el ranking de puntuaciones.
 - Escuchar música.
 - Gestionar su cuenta de usuario.
 - Jugar a los mini juegos.

- **Usuarios rol admin podrán:**
 - Realizar las funciones de usuarios rol user.
 - Gestionar las cuentas de usuarios.
 - Gestionar los parámetros de:
 - Cron.
 - Juego.
 - Partida.
 - Gestionar los recursos de:
 - Imágenes principales.
 - Personajes.

- Música.
- Videos.

12.1.1.4- LOGIN

Para acceder a la aplicación se deberá poseer una cuenta de usuario con email verificado.

Si no se dispone de una se deberá registrar.

Este registro solo permite registrarse con rol user.

Se requiere de un nick, email y pass para el registro.

Al registrarse el sistema:

- Mandara un email para verificar el email introducido.
- Creará una partida con nivel 1 y un personaje seleccionado aleatoriamente.
- Creará una cuenta con los datos del usuario y la opción de recibir avisos seleccionados.

El registro no estará completo hasta haber verificado el email.

El sistema vuelve a mandar el email de verificación cada hora.

Si se verifica el email y no se logea pasadas 24h el sistema borrara los datos de dicha cuenta.

El sistema no tendrá en cuenta las partidas con emails no verificados, no actualizara stats ni consumibles.

Si se cambia el email se deberá de volver a verificar.

En este caso se dispondrá de 48h para verificarlo antes de que el sistema borre los datos.

Las cuentas con rol admin no podrán registrarse ni borrarse, si no existen, las creara el sistema automáticamente con email verificado.

Si se ha modificado el email se enviara el email de validación cada hora, pero no serán borrados los datos hasta pasadas 48h.

Si se ha olvidado la contraseña, se podrá recuperar mediante Olvide mi contraseña. Esta opción solo estará disponible para emails verificados.

La aplicación contara con sesiones con una duración de medio día.

Si se cierra el navegador y se vuelve a entrar antes de la duración de las sesión (Sin haber seleccionado Salir), no se necesitara volverá a logear.

El sistema solo admite una cuenta logeada al mismo tiempo.

Al logear se desconectan al resto de usuarios con la misma cuenta.

Si se selecciona Salir, cerraremos la sesión.

12.1.1.5- INPUTS

- Nick:

Tamaño mínimo 1 carácter.

Tamaño máximo 15 caracteres.

Admite cualquier carácter.

Ha de ser único.

- **Email:**

Tamaño máximo 40 caracteres.

Solo admite emails válidos.

Ha de ser único.

- **Contraseña:**

Tamaño mínimo 8 carácter.

Tamaño máximo 15 caracteres.

Admite cualquier carácter.

- **Nombre:**

Tamaño mínimo 1 carácter.

Tamaño máximo 30 caracteres.

Admite cualquier carácter.

- **Alt:**

Tamaño mínimo 1 carácter.

Tamaño máximo 30 caracteres.

Admite cualquier carácter.

- **Url:**

Tamaño máximo 100 caracteres.

Solo admite URL válidas.

Aunque soporta formato http, es recomendable utilizar solo recursos con https por cuestiones de seguridad.

- **Cron interval:**

Valor mínimo 1

Valor máximo 1000000000000

- **Cron interval stats:**

Valor mínimo 1

Valor máximo 100

- **Cron interval consumibles:**

Valor mínimo 1

Valor máximo 100

- **Cron menos 50 y 30:**

Valor mínimo 0

Valor máximo 100

- **Cron menos stats:**

Valor mínimo 0

Valor máximo 100

- **Partida stats más 100:**

Valor mínimo 0

Valor máximo 100

- **Partida stats más 50:**
Valor mínimo 0
Valor máximo 100

- **Partida consumibles recarga:**
Valor mínimo 0
Valor máximo 100

- **Partida jugar stats:**
Valor mínimo 0
Valor máximo 100

- **Partida jugar energía:**
Valor mínimo 0
Valor máximo 100

- **Juego interval:**
Valor mínimo 0
Valor máximo 4999

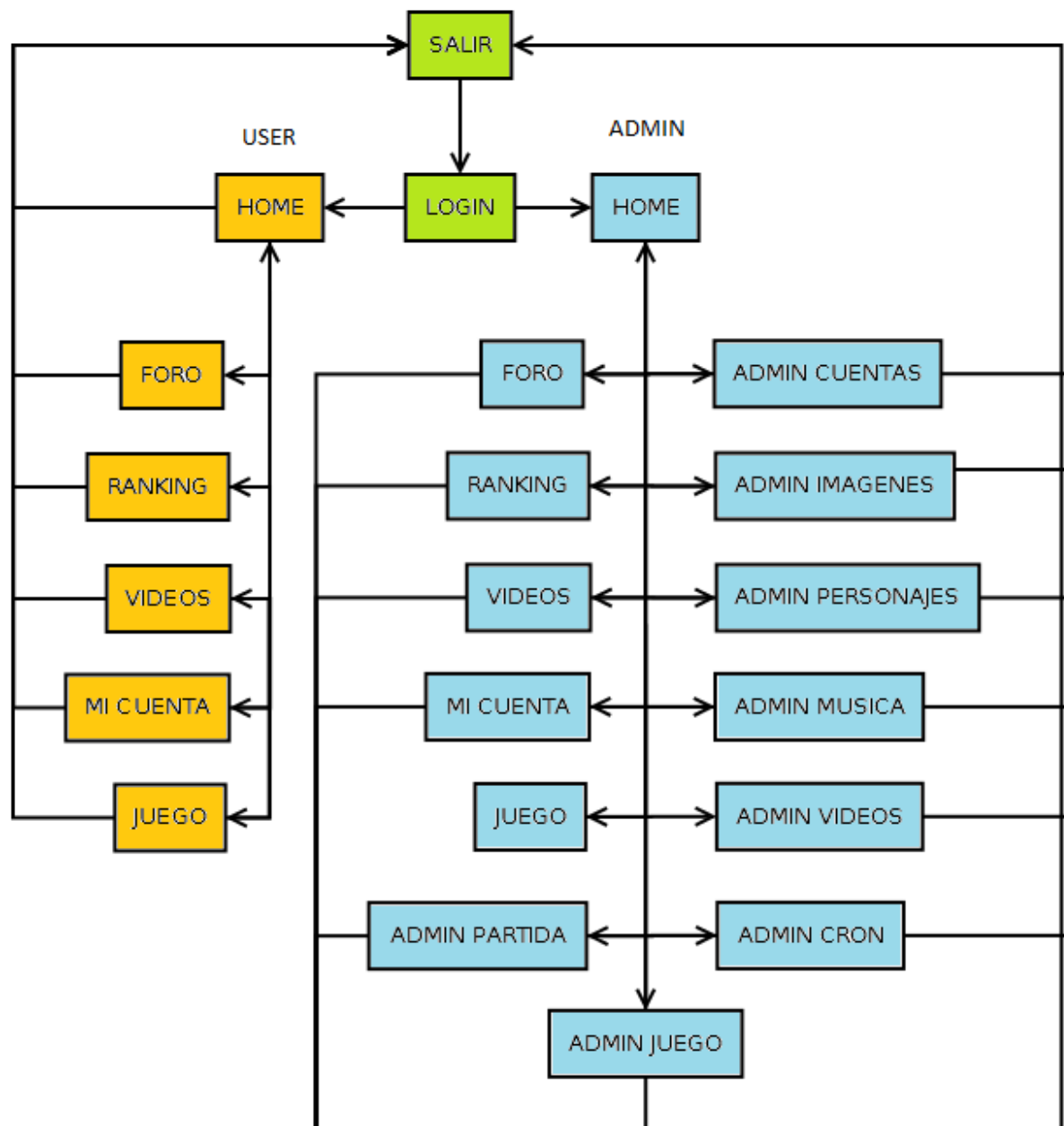
- **Juego puntos aciertos**
Valor mínimo 0
Valor máximo 1000000000000

- **Juego puntos fallo**

Valor mínimo 0

Valor máximo 1000000000000

12.1.1.6- MAPA NAVEGACIÓN



12.1.1.7- NAVEGADORES

A través del navegador se podrá desplazar por las distintas páginas de la aplicación.

En función del rol del usuario logeado, se tendrá acceso a más o menos páginas.

Se mostrara resaltada la página en la que se encontré.

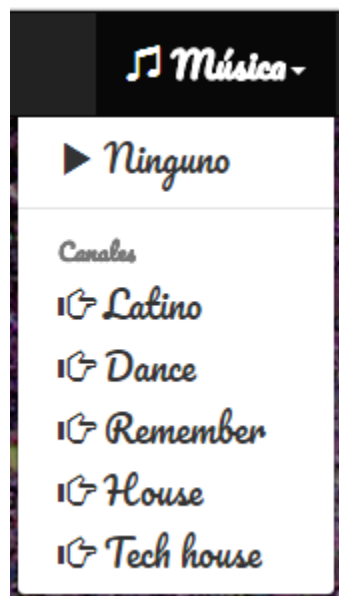
Desde aquí también se podrá escuchar música o desloguear.

La página juego solo estará disponible a través de home cuando la energía de la partida supere el 80%.

Si está disponible aparecerá el siguiente icono:



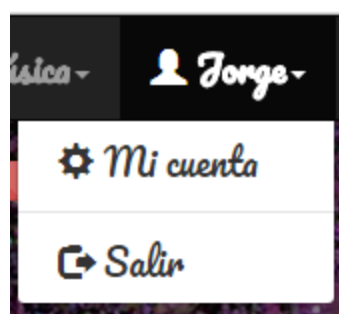
- Música



- Login



- User



El nombre mostrado es el del user logeado.

- Admin

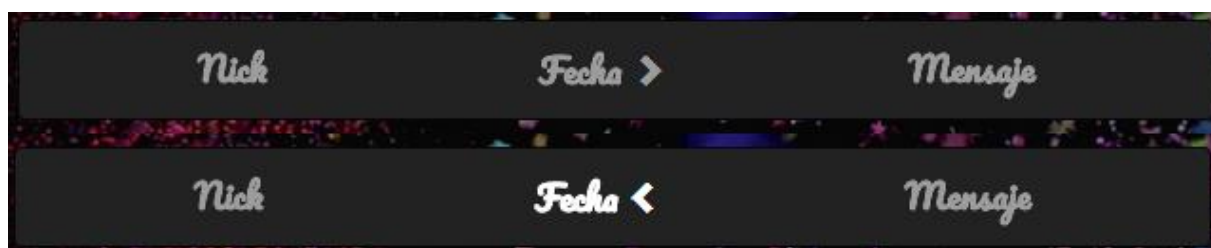


El nombre mostrado es el del user logeado.

12.1.1.8- ORDEN

Los datos mostrados serán ordenados por defecto por fecha.

En aquellas páginas que dispongan de menú de ordenación aparecerá el siguiente menú:



El campo actual por el que está siendo ordenado aparecerá junto a > o <.

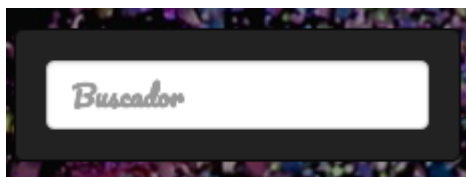
Si aparece > los datos serán ordenados de mayor a menor.

Si aparece < los datos serán ordenados de menor a mayor.

Al hacer click sobre una de las opciones se cambiara el campo y orden por el que ordenar.

12.1.1.9- BUSCADOR

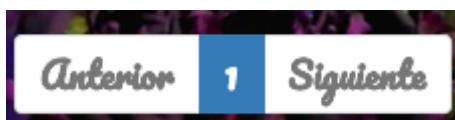
En aquellas páginas que dispongan de menú de búsqueda aparecerá el siguiente menú en la parte derecha:



Este menú complementa al menú de ordenación, ya que busca en el campo de orden seleccionado, aquellos que contengan el texto del menú buscador.

12.1.1.10- PAGINACIÓN

En aquellas páginas que dispongan de menú de paginación aparecerá el siguiente menú en la parte inferior derecha:



Este menú complementa al menú de ordenación y búsqueda, ya que la paginar tiene en cuenta el orden y campo a buscar.

Cada página contendrá 10 documentos como máximo.

Muestra la página actual.

Si está en la primera página, deshabilita el botón anterior.

Si no existen más documentos en las siguientes páginas, deshabilita el botón siguiente.

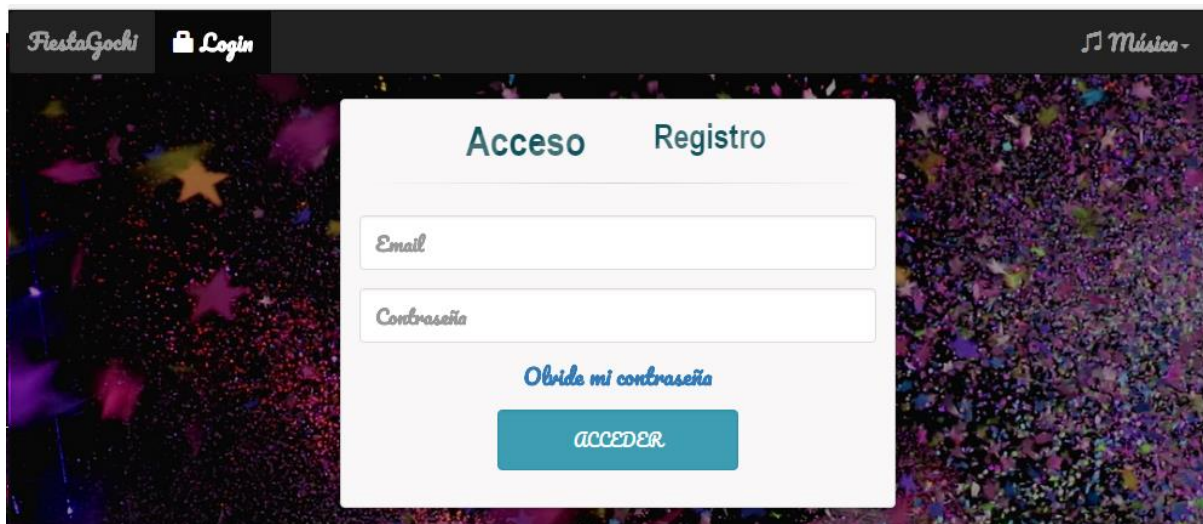
12.1.1.11- ORDEN, PAGINACIÓN Y BUSCADOR AL INTRODUCCIÓN DATOS

Al insertar un nuevo documento, automáticamente se elimina el texto buscado en el buscador, cambia a la primera página y ordena de mayor a menor por fecha de creación.

De esta forma siempre veremos primero el documento insertado.

12.1.2- LOGIN

Se mostrara la siguiente página:



Desde aquí se podrá logear, registrar o recuperar la contraseña.

Al hacer click en acceso, registro u olvide mi contraseña aparecerá su respectivo formulario.

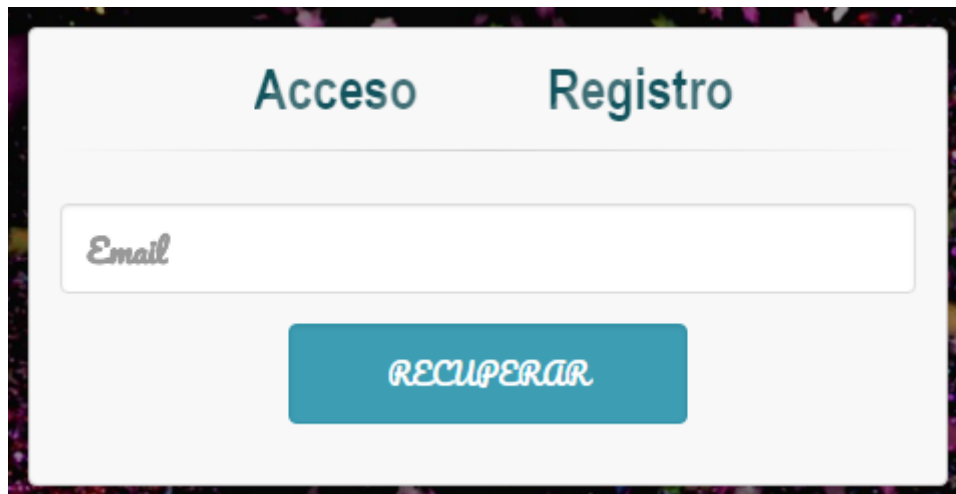
- Acceso

Al introducir los datos y seleccionar acceder pueden suceder los siguientes casos:

- Si los datos introducidos no son correctos aparecerá un mensaje de advertencia.
- Si el email no existe aparecerá un mensaje de advertencia.
- Si el email no ha sido verificado aparecerá un mensaje de advertencia.

- Si los datos introducidos no coinciden con los registrados aparecerá un mensaje de advertencia.
- Si los datos son correctos inicia la sesión y redirige a la página home.

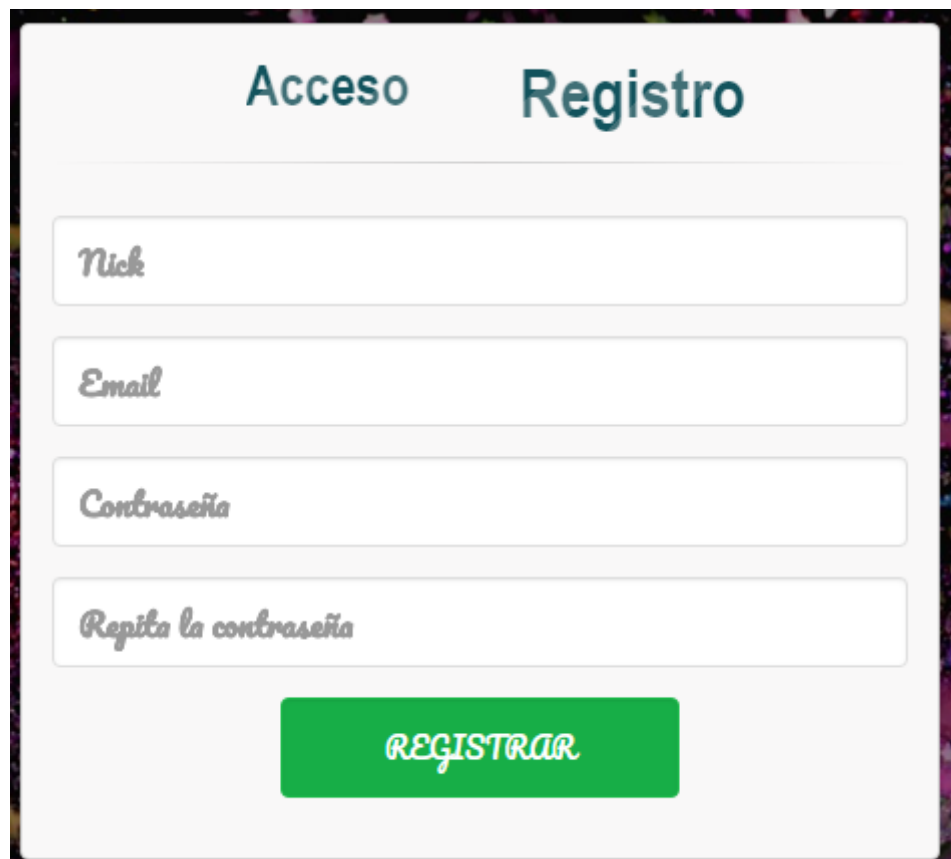
- Olvide mi contraseña

El formulario de recuperación de contraseña tiene un fondo blanco con un borde decorativo de confeti multicolor. En la parte superior, hay dos pestañas: 'Acceso' y 'Registro', ambas en un color verde azulado. Debajo de las pestañas, hay un campo de entrada de texto con el placeholder 'Email' en un color gris. En el centro del formulario, hay un botón rectangular de color verde azulado con el texto 'RECUPERAR' en un color blanco.

Al introducir los datos y seleccionar recuperar contraseña pueden suceder los siguientes casos:

- Si los datos introducidos no son correctos aparecerá un mensaje de advertencia.
- Si el email no existe aparecerá un mensaje de advertencia.
- Si el email no ha sido verificado aparecerá un mensaje de advertencia.
- Si el email existe y ha sido verificado aparecerá un mensaje de advertencia y el sistema mandará el email para la recuperación de la contraseña.

- Registro



Acceso Registro

Nick

Email

Contraseña

Repita la contraseña

REGISTRAR

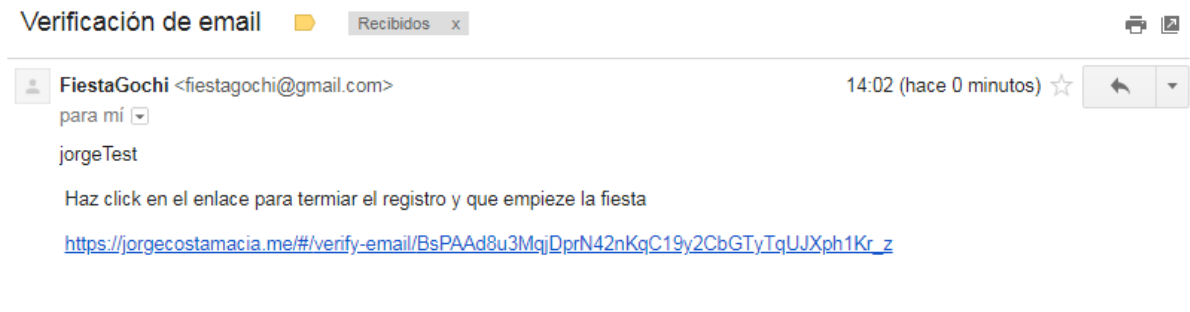
Al introducir los datos y seleccionar registrar pueden suceder los siguientes casos:

- Si los datos introducidos no son correctos aparecerá un mensaje de advertencia.
- Si el email ya existe aparecerá un mensaje de advertencia.
- Si el Nick ya existe aparecerá un mensaje de advertencia.
- Si las contraseñas no coinciden aparecerá un mensaje de advertencia.

- Si no existen ni el Nick ni el email y las contraseñas son correctas se creara la cuenta, se mandara el email de verificación y aparecerá un mensaje de advertencia.

12.1.3- EMAIL VERIFICACIÓN

Al registrarse o cambiar el email se manda el siguiente email:



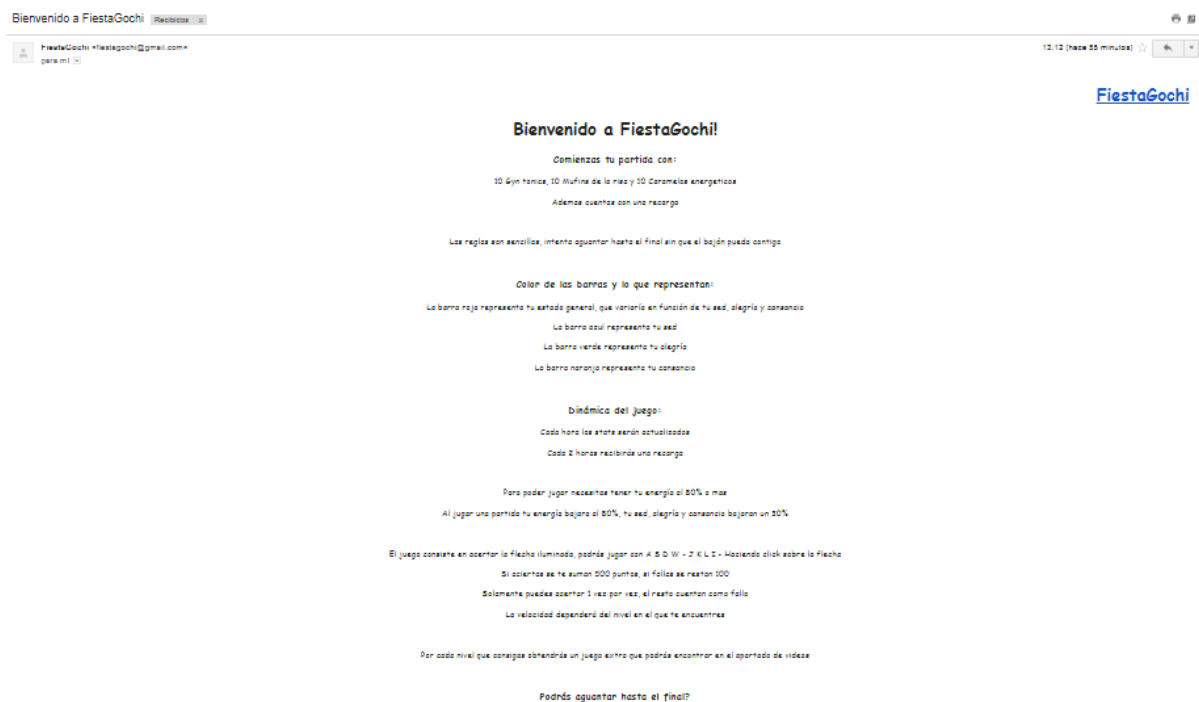
Si se hace click sobre el enlace, se abrirá una nueva ventana.

Se mostrara una advertencia de que el email ha sido verificado.

Verificara nuestro email y ya podremos acceder.

12.1.4- EMAIL BIENVENIDA

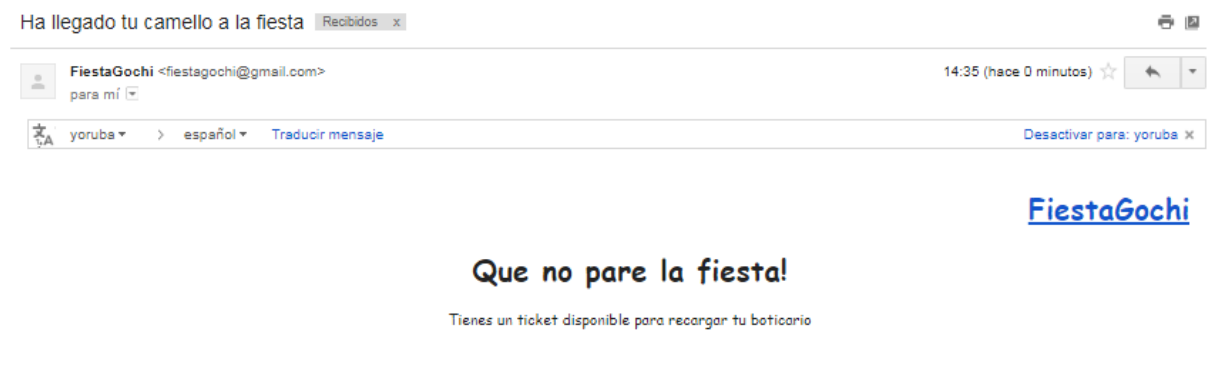
Al verificar el email se manda el siguiente email:



Este email contiene las instrucciones para jugar.

12.1.5- EMAIL RECARGA DISPONIBLE

Cuando el sistema agrega recargas a las partidas se manda el siguiente email:



12.1.6- EMAIL STATS POR DEBAJO 50%

Cuando el sistema actualiza los stats, si la energía o subidon de la partida es igual o inferior al 50% se manda el siguiente email:



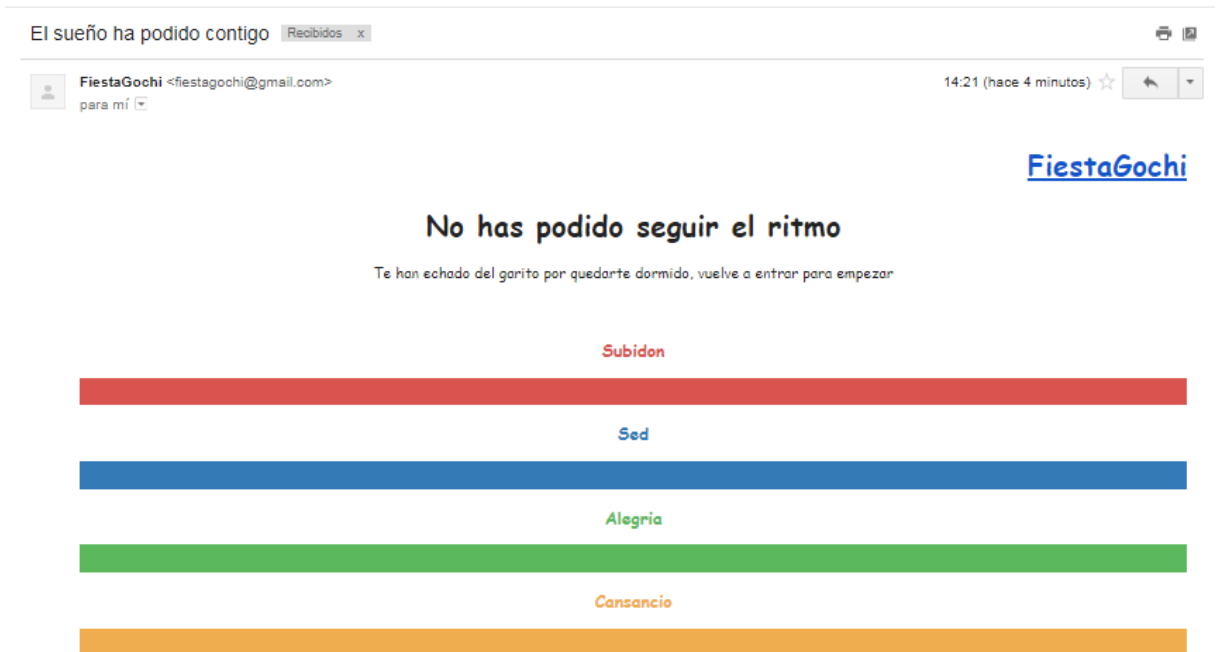
12.1.7- EMAIL STATS POR DEBAJO DEL 25%

Cuando el sistema actualiza los stats, si la energía o subidon de la partida es igual o inferior al 25% se manda el siguiente email:



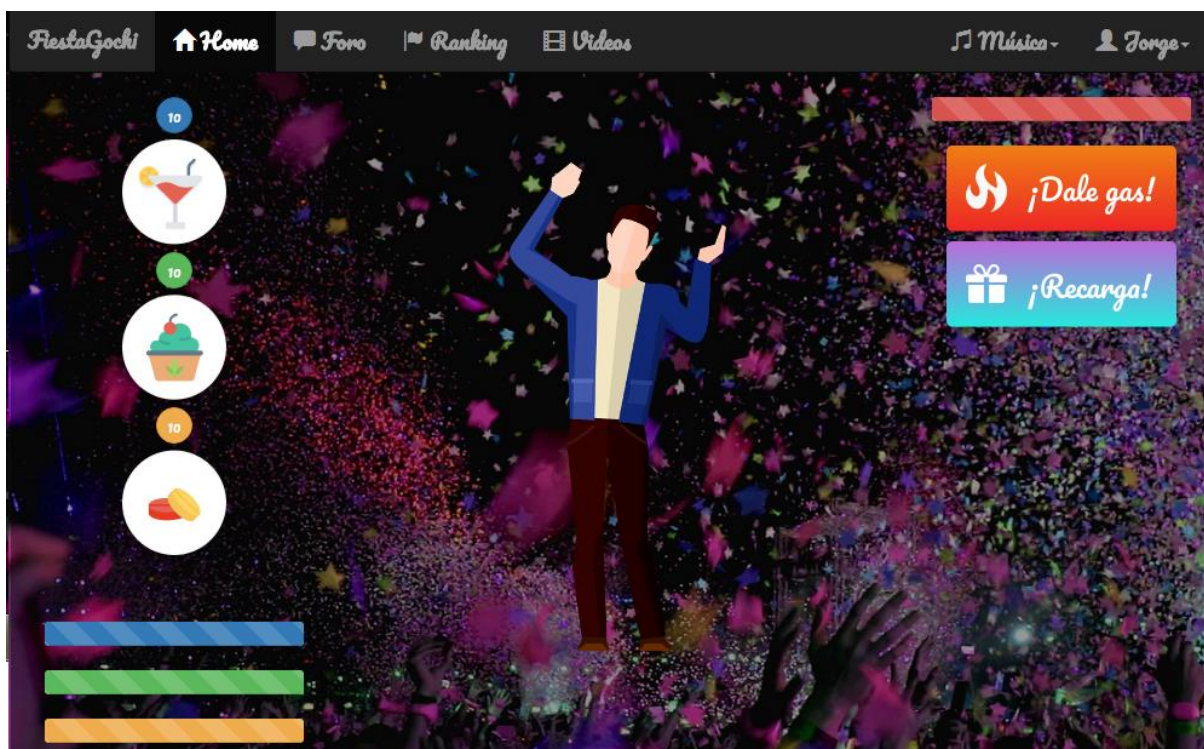
12.1.8- EMAIL HAS PERDIDO

Cuando el sistema actualiza los stats, si la energía llega a 0, se reinicia la partida y se manda el siguiente email:



12.1.9- HOME

Se mostrara la siguiente página:



Los iconos de la izquierda representan los consumibles que se podrán usar.
El número encima de cada consumible representa la cantidad disponible.

Las barras de la parte inferior representan cada uno de nuestros estados.

Cuanto más llenas estén las barras en mejor estado estará el personaje.

Precaución!!! Porque puede que no le siente del todo bien el consumir un objeto con su estado lleno.

Se pueden asociar a través de los colores.

- La barra azul representa la sed.

Si el círculo azul es superior a 0, significara que se dispone de gyn tonics.

Al hacer click sobre el gyn tonic o su cantidad estos giraran, si se tenía gyn tonics disponibles su contador disminuirá en 1.

Si la barra de sed no está llena, ese estado mejorara al consumir el gyn tonic.

Dependiendo del estado que representan las 2 barras cercanas, también mejorara la energía o subidón, representada por la barra roja.

Si se tenía el estado de sed lleno le dará un coma etílico y disminuirá tanto la energía como la sed.

- La barra verde representa la alegría.

Si el círculo verde es superior a 0, significara que se dispone de muffins de la risa.

Al hacer click sobre el muffin o su cantidad estos girara, si se tenía muffins disponibles su contador disminuirá en 1.

Si la barra de alegría no está llena, ese estado mejorara al consumir el muffin.

Dependiendo del estado que representan las 2 barras cercanas, también mejorara la energía, representada por la barra roja.

Si se tenía el estado de alegría lleno le dará amarillo y disminuirá tanto la energía como la alegría.

- La barra naranja representa el cansancio o sueño.
Si el círculo naranja es superior a 0, significara que se dispone de caramelos energéticos.
Al hacer click sobre el caramelo o su cantidad estos giraran, si se tenían caramelos disponibles su contador disminuirá en 1.
Si la barra de cansancio no está llena, ese estado mejorara al consumir el caramelo.
Dependiendo del estado que representan las 2 barras cercanas, también mejorara la energía, representada por la barra roja.
Si se tenía el estado de cansancio lleno le dará un jamacuco y disminuirá tanto la energía como el cansancio.

En el centro aparece el personaje con el que se esté jugando.

La barra de la parte derecha representa la energía o subidón, es el estado más importante durante el juego.

Si este llega a 0 se habrá perdido y se reiniciara la partida.

Este estado se actualizara automáticamente por el sistema.

Para mantenerlo en un estado óptimo se deberá seguir los pasos antes explicados.

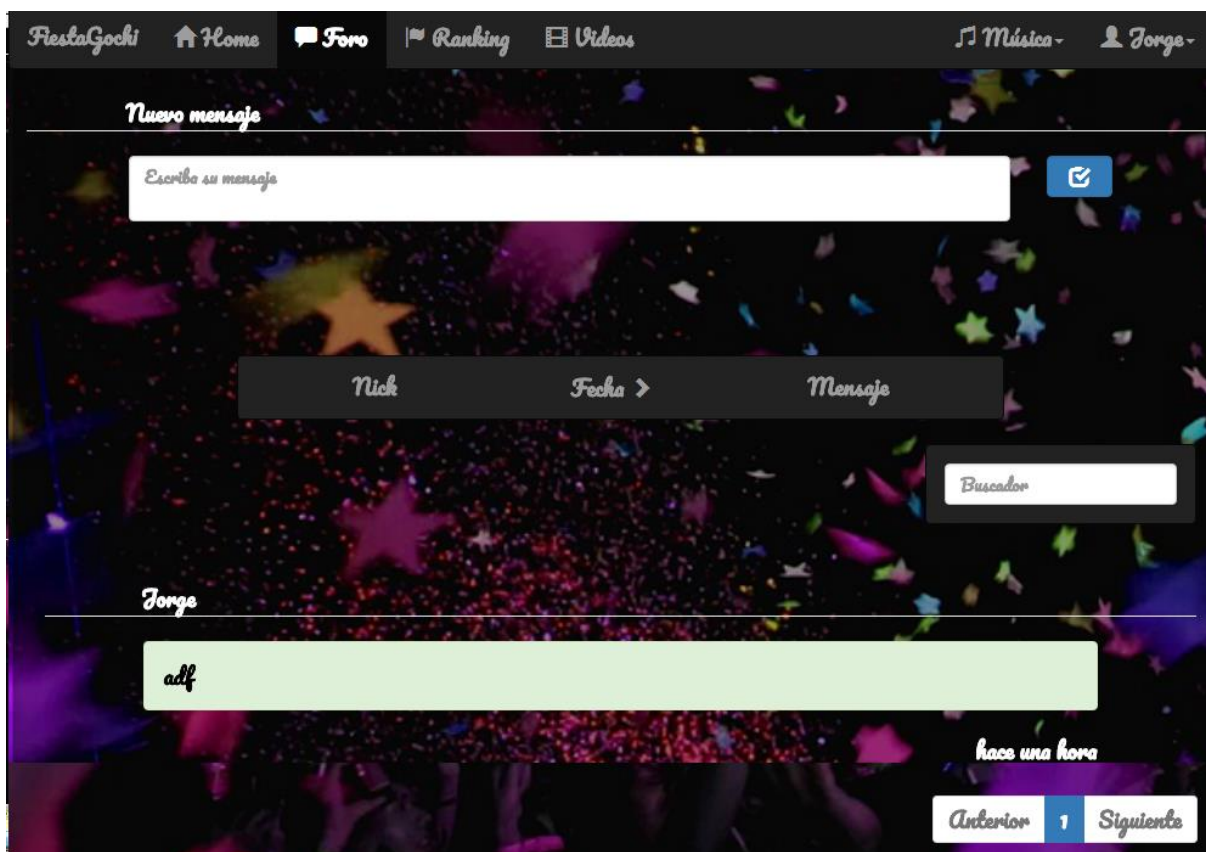
Los iconos Dale gas y Recarga no estarán siempre disponibles.

- El icono Dale gas aparecerá cuando la energía supere el 80%, al hacerle click se redirigirá a la página juego donde se tendrá que demostrar las dotes de baile.
Al hacer click los estados disminuirán.

- El icono Recarga aparecerá cuando esté disponible una recarga.
Al hacer click sobre él, se recargaran los objetos y desaparecerá el icono.
Al recargar se verá cómo giran los consumibles y su cantidad sube.
Solo se podrá tener una recarga disponible, no se acumulan.

12.1.10- FORO

Se mostrara la siguiente página:



Se dispone de una paginación, orden y buscador antes explicados.

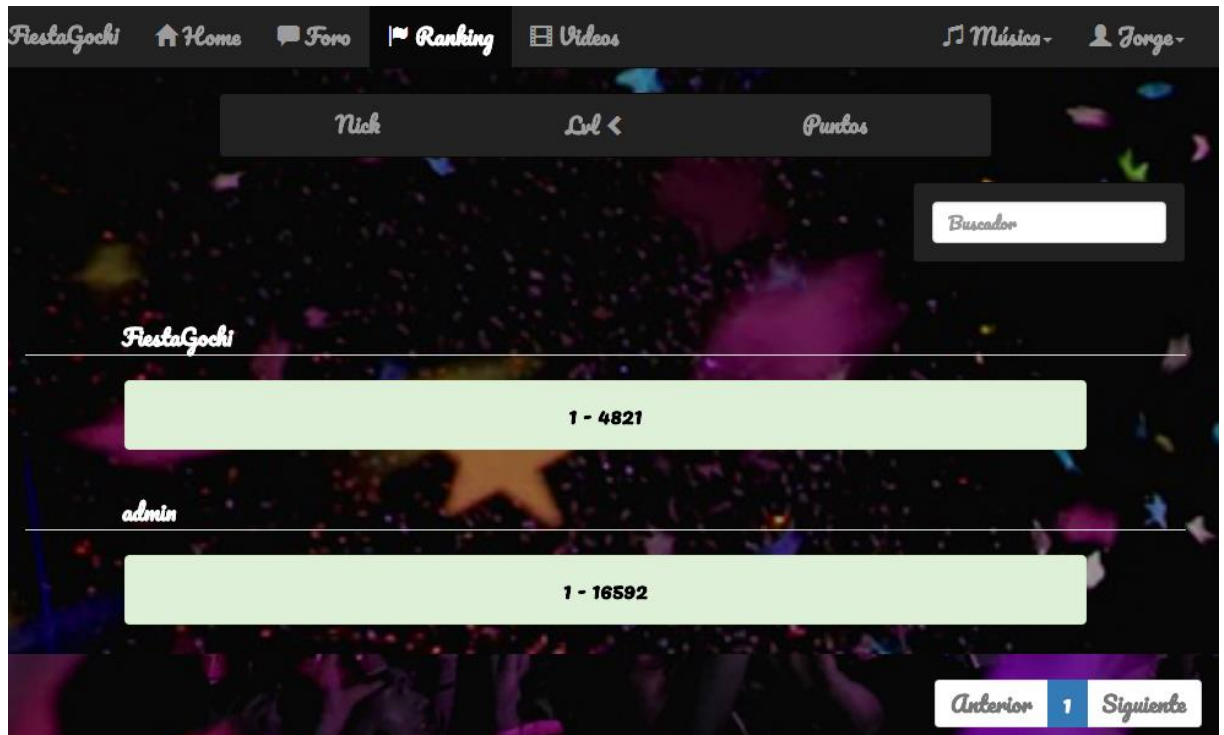
En la parte superior se tendrá un formulario para escribir en el foro.

Al escribir un mensaje aparecerá el primero.

Aquí se podrá ver, quien escribió el mensaje, el contenido y el tiempo transcurrido desde que fue escrito.

12.1.11- RANKING

Se mostrara la siguiente página:

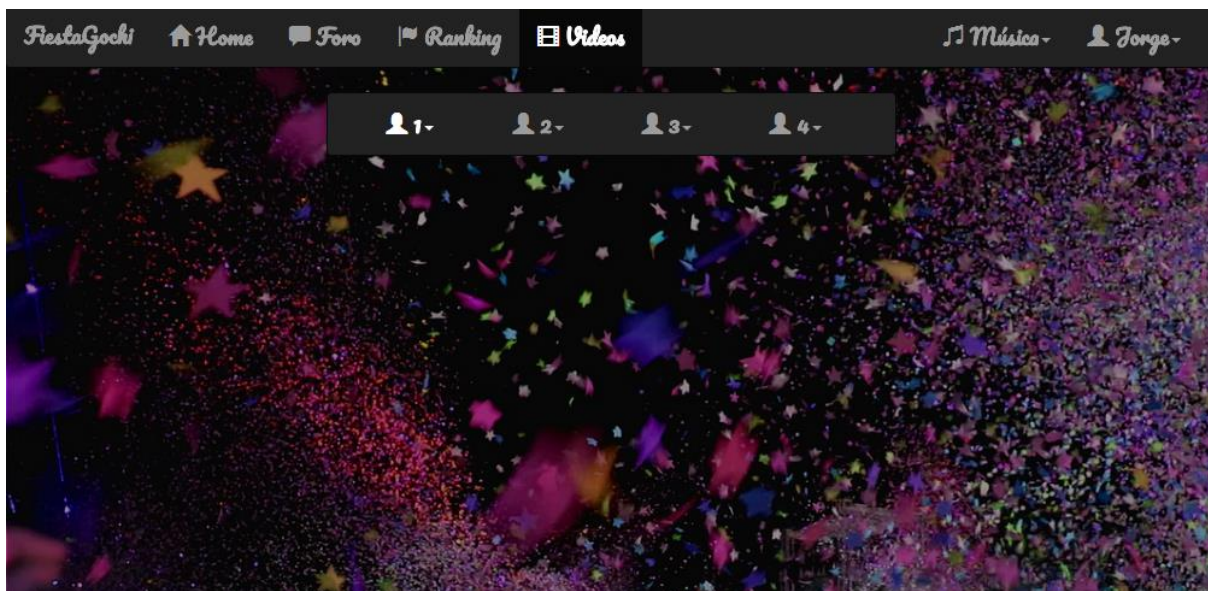


Se dispone de una paginación, orden y buscador antes explicados.

Desde aquí podremos ver quien obtuvo esa puntuación, en qué nivel al obtuvo y cuál fue su puntuación.

12.1.12- VIDEOS

Se mostrara la siguiente página:



Desde aquí se podrá acceder a mini juegos en forma de videos.

En función del nivel en el que se esté, se dispondremos de esa cantidad de mini juegos.

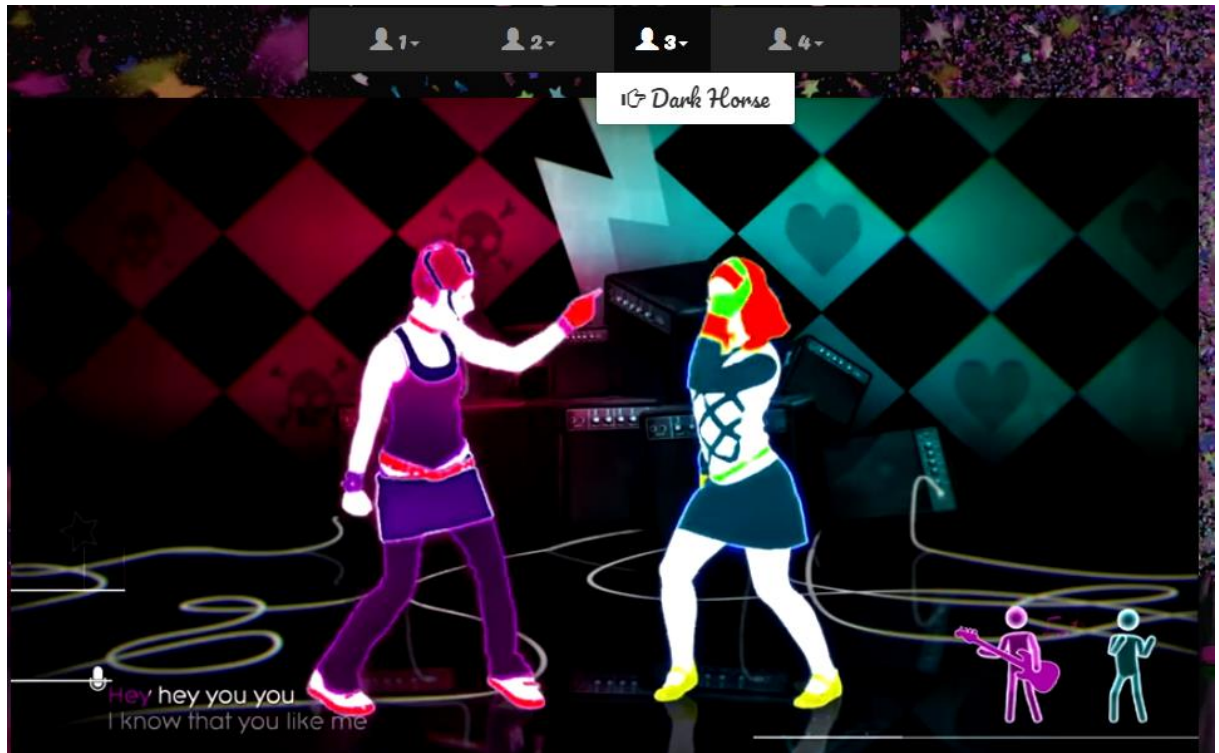
Se tiene un mini juego para 1 2 3 y 4 por cada nivel obtenido.

Estos mini juegos se podrán jugar tantas veces como se desee.

Si se pierde o se reinicia la partida, se pierden los mini juegos obtenidos.

Los números representan la cantidad de jugadores que pueden jugar a cada mini juego.

Al seleccionar un video nos aparece el siguiente mini juego:



12.1.13- MI CUENTA

Se mostrara la siguiente página:

The screenshot shows the 'MI CUENTA' page with the following details:

- Nick:** Jorge
- Email:** jorgamovilistico@gmail.com
- Avisos:** No recibir
- Contraseña:** (masked with asterisks)
- Personaje:** El Tony
- Nivel:** 1

Desde aquí se podrá ver y modificar los datos de la cuenta logeada.

Los campos siguen los mismos criterios antes explicados.

Si se selecciona la cruz roja inferior se reinicia la partida.

Si se seleccionan los botones naranjas se despliega un formulario con los campos necesarios para modificar dicho campo.



Desde estos formularios se podrá modificar los datos haciendo click sobre el icono rojo, si es correcto se cambiara, mostrara una advertencia y esconderá el formulario.

En caso de ser incorrecto se mostrara una advertencia y esconderá el formulario.

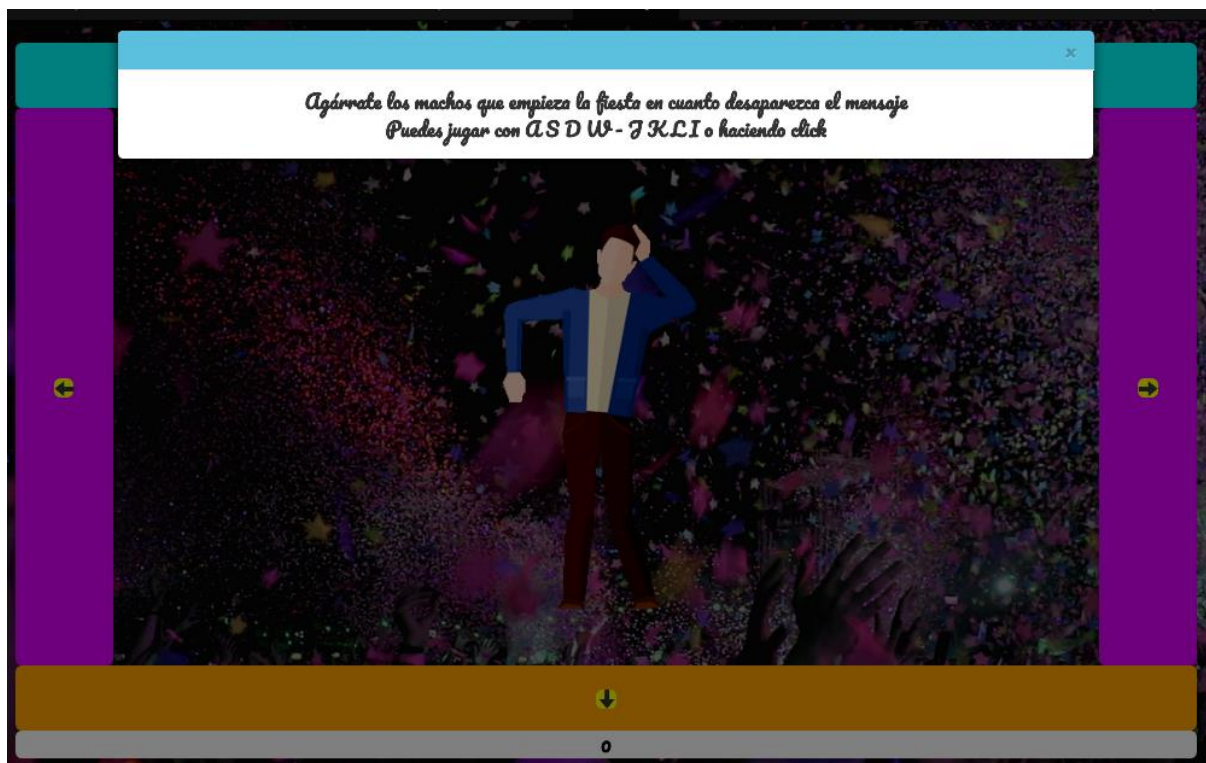
También se podrá volver a hacer click sobre el mismo icono naranja para esconder el menú.

Solo puede haber un menú desplegado, por lo que al desplegar un menú se esconden los anteriores.

Al cambiar el email se deslogueara y no se podrá acceder hasta que haya sido verificado.

12.1.14- JUEGO

Se mostrara la siguiente página:



Las cuatro flechas que cambian de color representan las posiciones que más tarde deberemos acertar.

El número en la parte inferior representa la puntuación actual, esta cambiara tanto de cantidad como de color al acertar o fallar la posición.

La página inicia con una demo y un mensaje de advertencia con las instrucciones.

Al terminar la demo se quitara el mensaje (sino se ha cerrado antes) y se quitara el color de los cuatro lados.

Solo podrá haber un lado activo por momento.

Al activarse ese lado, ese cambiara de color y la barra de puntuación se volverá blanca.

Se tendrá que acertar el lado activo o bien haciendo click sobre el lado o apretando la tecla que lo representa.

- Arriba: W I
- Abajo: S K
- Izquierda: A J
- Derecha D L

Si se acierta, la puntuación se volverá verde y subirá de valor.

Si se falla, la puntuación se volverá roja y bajara de valor.

Solo se permite un acierto por activación, una vez acertado el resto cuentan como fallos.

Los fallos no tienen límite por lado.

Al terminar el juego, se mostrara una advertencia y se redirigirá a la página home.

Si se ha superado el nivel, el nivel se incrementara en 1 y se registrara la puntuación en el ranking.

12.1.15- ADMIN CUENTAS

Se mostrara la siguiente página:

Nuevo usuario

fiestagochi@yna

Email

Nick Email Creación >

Buscador

Id: H5eWLo8R0X8eR7e2

forja

forjanoviliteco@gmail.com

Verificado

Recibir

11-08-2018 14:44:00

Anterior 1 Siguiendo

Se dispone de una paginación, orden y buscador antes explicados.

Los campos siguen los mismos criterios antes explicados.

Por seguridad la contraseña y roles no son modificables.

La partida tampoco es modificable para evitar posibles trampas o ayudas por partes de los administradores.

En la parte superior habrá un formulario para registrar nuevos usuarios.

Al registrar un nuevo usuario aparecerá el primero.

Si no es correcto se mostrara una a advertencia.

En la parte central aparecen los usuarios registrados.

Desde aquí se podrá ver el id y los datos de los usuarios registrados.

Si se hace click en el icono naranja se actualizarán los datos que hayan sido modificados de dicho usuario.

Si se introduce un nuevo email, se mandara el email de verificación.

El usuario dueño de dicho email será deslogueado antes de hacer una acción.

Si se hace click en el icono rojo, se eliminaran todos los registros asociados a dicho jugador.

12.1.16- ADMIN IMÁGENES

Se mostrara la siguiente página:

Section	Field 1	Field 2	Field 3	Buttons
Imágenes Body	Background home	Imágenes background home	http://orguendelamaria.es/img/background_body.png	Orange, Red
Imágenes Gps Tonic	Gps tonic	Imágenes gps tonic	http://orguendelamaria.es/img/gps_tonic.png	Orange, Red
Imágenes Escudo de la vida	Escudo de la vida	Imágenes escudo de la vida	http://orguendelamaria.es/img/escudo_vida.png	Orange, Red
Imágenes Coramulo anagelita	Coramulo anagelita	Imágenes coramulo anagelita	http://orguendelamaria.es/img/coramulo_anagelita.png	Orange, Red
Imágenes Dado Gps	Dado gps	Imágenes dado gps	http://orguendelamaria.es/img/dado_gps.png	Orange, Red
Imágenes Recarga	Recarga	Imágenes recarga	http://orguendelamaria.es/img/recarga.png	Orange, Red

Desde aquí se podrá cambiar tanto el nombre, ALT como la URL de las imágenes principales.

Los campos siguen los mismos criterios antes explicados.

Si se hace click sobre el icono naranja, se abrirá una nueva ventana con la imagen de la URL.

Si se hace click sobre se icono rojo, si los datos son correcto se actualizan los datos de dicha imagen y mostrara advertencia.

Si los datos son incorrectos se mostrara advertencia.

12.1.17- ADMIN PERSONAJES

Se mostrara la siguiente página:

Se dispone de una paginación, orden y buscador antes explicados.

Los campos siguen los mismos criterios antes explicados.

En la parte superior habrá un formulario para el registro de nuevos personajes.

Al hacer click sobre uno de los iconos naranjas de dicho formulario, se abrirá una nueva ventana con la URL.

Si se hace click sobre el icono rojo y los datos son correctos, se guardara y aparecerá el personaje en primera posición.

Si los datos son incorrectos se mostrara advertencia.

En la parte central aparecen los personajes registrados.

Si se hace click sobre el icono naranja y los datos son correctos, se actualiza y aparece una advertencia.

Si los datos son incorrectos se mostrara una advertencia.

Si se hace click en el icono rojo se elimina dicho personaje.

Se mostrara un personaje por defecto a los usuarios que tuvieran dicho personaje seleccionado.

12.1.18- ADMIN MÚSICA

Se mostrara la siguiente página:

The screenshot shows the 'ADMIN MÚSICA' interface. At the top, a navigation bar includes links for 'Home', 'Foro', 'Ranking', 'Videos', and 'Música'. The 'Música' section is currently selected. Below the navigation bar, the page title is 'Nueva emisora'. There is a form to add a new radio station with fields for 'Nombre' (containing 'http://127.0.0.1') and 'URL'. To the right of the 'Nombre' field is an orange button with a plus icon. Below the form, there is a table listing existing radio stations. Each row in the table has a 'Nombre' field, a 'URL' field, and an orange button with a plus icon. The table contains six rows, all with the same URL 'http://127.0.0.1'. At the bottom right of the page, there are two buttons: 'Cerrar' and 'Signado'.

Se dispone de una paginación, orden y buscador antes explicados.

Los campos siguen los mismos criterios antes explicados.

En la parte superior habrá un formulario para el registro de nuevas emisoras de música.

Al hacer click sobre uno de los iconos naranjas de dicho formulario, se abrirá una nueva ventana con la URL.

Si se hace click sobre el icono rojo y los datos son correctos, se guardara y aparecerá la emisora en primera posición.

Si los datos son incorrectos se mostrara advertencia.

En la parte central aparecen las emisoras registradas.

Si se hace click sobre el icono naranja y los datos son correctos, se actualiza y aparece una advertencia.

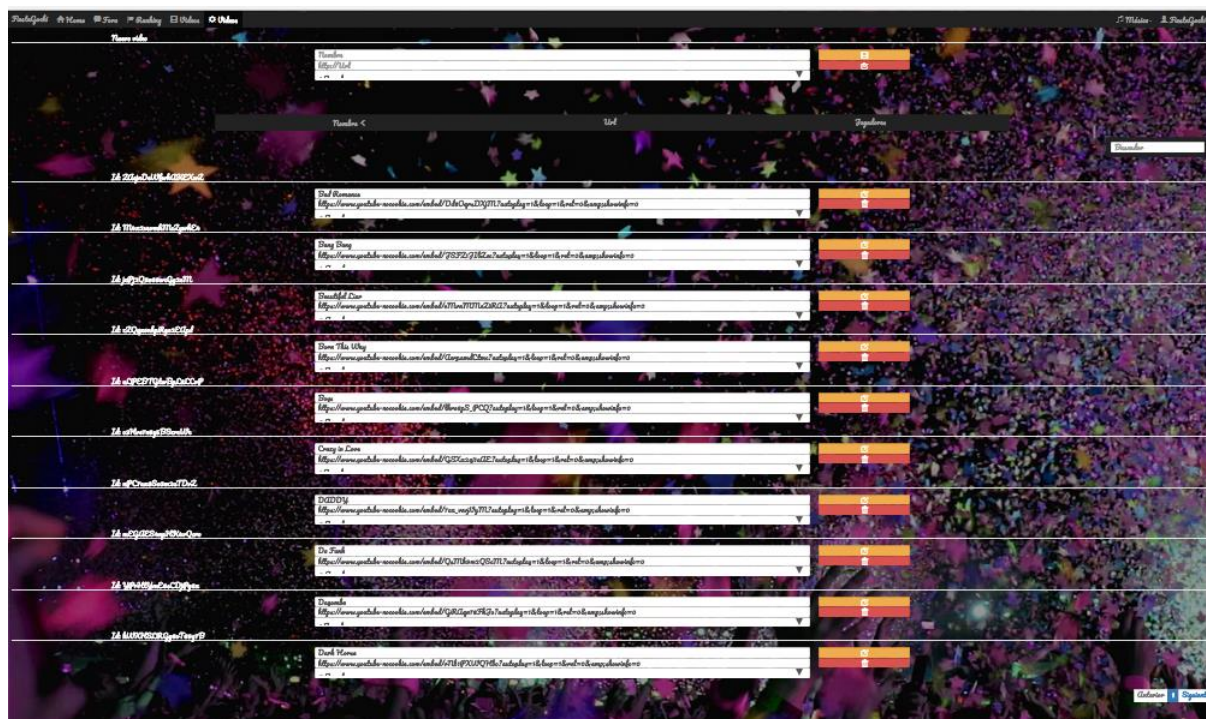
Si los datos son incorrectos se mostrara una advertencia.

Si se hace click en el icono rojo se elimina dicha emisora.

Este cambio no surtirá efecto en los usuarios hasta que cierren y vuelvan a abrir el navegador.

12.1.19- ADMIN VIDEOS

Se mostrara la siguiente página:



Se dispone de una paginación, orden y buscador antes explicados.

Los campos siguen los mismos criterios antes explicados.

En la parte superior habrá un formulario para el registro de nuevos mini juegos.

Al hacer click sobre uno de los iconos naranjas de dicho formulario, se abrirá una nueva ventana con la URL.

Si se hace click sobre el icono rojo y los datos son correctos, se guardara y aparecerá el video en primera posición.

Si los datos son incorrectos se mostrara advertencia.

En la parte central aparecen los videos registrados.

Si se hace click sobre el icono naranja y los datos son correctos, se actualiza y aparece una advertencia.

Si los datos son incorrectos se mostrara una advertencia.

Si se hace click en el icono rojo se elimina dicho video.

12.1.20- ADMIN CRON

Se mostrara la siguiente página:

Label	Value	Action
Interval	3000000	[Checkmark]
Interval slots	1	
Interval consumables	2	
Mamas 50 Uno	30	
Mamas 50 Dos	15	
Mamas 50 Tres	20	
Mamas 30 Uno	5	
Mamas 30 Dos	10	
Mamas 30 Tres	15	
Mamas slots	10	

Desde aquí se podrá configurar los parámetros de los crons.

Los cambios no surtirán efecto hasta reiniciar el servidor o volver a desplegar la aplicación.

Los campos siguen los mismos criterios antes explicados.

Al hacer click sobre el icono rojo, si los datos son correctos, se actualizan y muestra mensaje de advertencia.

Si son incorrectos se muestra mensaje de advertencia.

Significado y representación de los campos:

- **Interval:**

Tiempo en milisegundos que transcurre entre cada ciclo del cron.

- **Interval stats**

Número de ciclos que transcurren hasta su ejecución.
Cron encargado de actualizar los stats de las partidas.
En este ejemplo se ejecutaría cada interval.

- **Interval consumibles**

Número de ciclos que transcurren hasta su ejecución.
Cron encargado de actualizar las recargas de las partidas.
En este ejemplo se ejecutaría cada 2 interval, es decir se ejecutaría una vez, la siguiente no y la siguiente sí.

- **Menos 50 Uno**

Cantidad de energía que disminuye si solo hay un stat secundario por debajo de 50.

- **Menos 50 Dos**

Cantidad de energía que disminuye si hay 2 stats secundario por debajo de 50.

- **Menos 50 Tres**

Cantidad de energía que disminuye si hay 3 stats secundario por debajo de 50.

- **Menos 30 Uno**

Cantidad de energía que disminuye si solo hay un stat secundario por debajo de 30.

- **Menos 30 Dos**

Cantidad de energía que disminuye si hay 2 stats secundario por debajo de 30.

- **Menos 30 Tres**

Cantidad de energía que disminuye si hay 3 stats secundario por debajo de 30.

- **Menos stats**

Cantidad que disminuye de los 3 stats secundarios.

12.1.21- ADMIN PARTIDA

Se mostrara la siguiente página:

Configuración	Valor
Stats más 100	20
Stats más 50	5
Consumibles recarga	5
Jugar stats	30
Jugar energía	50

Desde aquí se podrá configurar los parámetros de la partida.

Los cambios surtirán efecto en el momento del cambio.

Los campos siguen los mismos criterios antes explicados.

Al hacer click sobre el icono rojo, si los datos son correctos, se actualizan y muestra mensaje de advertencia.

Si son incorrectos se muestra mensaje de advertencia.

Significado y representación de los campos:

- **Stats más 100:**

Cantidad que disminuye el stat secundario y la energía al consumir el consumible asociado estando ese estado lleno.

- **Stats más 50**

Cantidad que incrementa el stat secundario al consumir el consumible asociado.

Cantidad que incrementa la energía cuando los 3 stats secundarios están por encima del 50%

- **Consumibles recarga**

Cantidad por consumible que se incrementa al usar una recarga

- **Jugar stats**

Cantidad de los stats secundarios que disminuye al jugar una partida.

- **Jugar energía**

Cantidad de energía que disminuye al jugar una partida.

12.1.22- ADMIN JUEGO

Se mostrara la siguiente página:

The screenshot shows the 'Juego' (Game) administration page. The background is a festive, dark-themed image with falling confetti and stars. The top navigation bar is dark with white text and icons for 'FiestaGochi', 'Home', 'Foro', 'Ranking', 'Videos', 'Juego' (highlighted), 'Música', and a user profile. Below the navigation bar, there are several configuration fields:

- Color 1**: A horizontal bar with a cyan color. To its right is a red button with a white edit icon.
- Color 2**: A horizontal bar with a green color.
- Color 3**: A horizontal bar with an orange color.
- Color 4**: A horizontal bar with a purple color.
- Interval**: A horizontal bar with the value '499' displayed inside.
- Puntos acierto**: A horizontal bar with the value '500' displayed inside.
- Puntos fallo**: A horizontal bar with the value '100' displayed inside.

Desde aquí se podrá configurar los parámetros del juego.

Los cambios surtirán efecto en el momento que un usuario inicie un juego.

Los campos siguen los mismos criterios antes explicados.

Al hacer click sobre el icono rojo, si los datos son correctos, se actualizan y muestra mensaje de advertencia.

Si son incorrectos se muestra mensaje de advertencia.

Significado y representación de los campos:

- **Color 1:**

Color número 1 utilizado en el juego.

- **Color 2:**

Color número 2 utilizado en el juego.

- **Color 3:**

Color número 3 utilizado en el juego.

- **Color 4:**

Color número 4 utilizado en el juego.

- **Interval**

Tiempo en milisegundos que transcurre entre cada activación de los lados del juego.

El tiempo entre activaciones se calcula restando a 5000 el interval por el nivel del usuario.

Si es por debajo de 0 se asigna 200.

- **Puntos acierto**

Cantidad de puntos que se suman a la puntuación al acertar un lado activo en el juego.

- Puntos fallo

Cantidad de puntos que se restan a la puntuación al acertar un lado activo en el juego.

12.2- MANUAL TÉCNICO

12.2.1- CONSIDERACIONES TÉCNICAS

La aplicación ha sido desarrollada utilizando el patrón **MVVM** de Meteor.
En este patrón los datos y las plantillas actúan de controlador.

Esta desarrollada íntegramente en JS.

Se han utilizado, siempre que fuera posible, las funciones nativas de Meteor.

En las funciones que Meteor no integra se ha utilizado jQuery para el apartado de DOM y underscore para el manejo de datos.

Meteor se despliega compilado en un contenedor Docker.

No es necesario tirar el servidor ni la aplicación para actualizar el contenedor y si el servidor se reinicia, siempre arranca con la aplicación como servicio.

La **ejecución** de la aplicación se separa en 2 apartados:

- **Cliente:**

- Se ejecuta en el cliente

- Todo lo que este dentro de la/las carpeta client

- También se puede gestionar con isClient

- **Server**

- Se ejecuta en el servidor

- Todo lo que este dentro de la/las carpeta server

- También se puede gestionar con isServer

Meteor utiliza singlepage REAL, al entrar en la aplicación se sirve todo lo necesario comprimido con GZIP, el resto de la navegación y acciones se realizan utilizando la cache del navegador.

En el cliente, Meteor utiliza la cache del navegador para funcionar.

Los mensajes de advertencia se muestran mediante modales.

12.2.2- ESTRUCTURA TEMPLATE

Para el pintado del HTML se utilizan templates, cada template cuenta con su JS y CSS.

- HTML´

- Importar una template:
`{{> Nombre_template }}`
- Importar una traducción
`{{_ Nombre_traduccion}}`
- Datos de un helper
`{{ Nombre_helper }}`
- Se puede utilizar lógica en las templates.

- JS

- helper :
Se utilizan para integrar datos a la template.
Todo lo creado dentro tiene un entorno reactivo.
- events:
Eventos capturados dentro de la template
- onCreated:
Se ejecuta crearse la template.
- onRendered
Se ejecuta al renderizarse por completo la template.

12.2.3- SEGURIDAD

Se ha integrado el protocolo HTTPS para mayor seguridad.

Para esta tarea se ha utilizado NGINX.

Se ha configurado NGINX para que fuerce el protocolo HTTPS, redirige siempre a este protocolo.

Se ha utilizado NGINX por integrar estas configuraciones y balanceadores de carga.

Dejando la aplicación preparada para posibles futuras aplicaciones.

Por motivos de seguridad se ha desactivado el menú contextual.

Aunque Meteor cuenta con métodos similares a los de APACHE, no se han editado por tener una buena configuración por defecto.

Desde el navegador solo se puede acceder a lo alojado en la carpeta public.

Se ha utilizado como control de errores en los inputs las funciones que integra HTML, ya que tanto Meteor como mongo escapan caracteres.

12.2.4- BASE DE DATOS (COLECCIONES)

La **base de datos** ha sido desarrollada en mongoDB.

Se distribuye en colecciones.

Cada colección sería el equivalente a una tabla en MYSQL.

Las colecciones utilizan como **PK** el campo `_id`.

Al ingresar un nuevo documento este ha de contener un `_id`, si no ha sido definido el sistema lo genera automáticamente.

Ya que mongo no tiene **relaciones**, han sido definidas las siguientes:

- Users y partidas: Se relacionan a través del `_id`.
- Users y resto de documentos: `Users._id` con `document.id_usuario`.

Cada colección cuenta con una API.

La API es la encargada de manejar los datos de dicha colección.

La escritura solo se puede hacer a través del servidor, los métodos utilizados por el cliente para la escritura están alojados en los ficheros `methods` y son llamados desde el cliente a través de `Meteor.call`.

Los datos a los que el cliente tiene acceso el cliente se controlan a través de:

- El servidor a través del fichero `publications` con el método `Meteor.publish` el cual publica los datos que queremos hacer accesibles.

- El cliente a través del método Meteor.subscribe el cual guarda en la cache del navegador los datos de la publicación a la que nos hemos suscrito del servidor, haciendo estos accesibles.

El servidor tiene acceso a todos los datos.

Existe una colección sesión creada en el cliente.

Esta colección se guarda en el navegador y es accesible y modificable por el cliente.

Se utiliza para las variables de control y estado.

12.2.5- RECURSOS

Las imágenes inicialmente son servidas a través del servidor mediante GET, se ha hecho de esta forma para dejar preparado el sistema para en un futuro externalizarlas.

La música y videos se obtienen por el cliente a través de APIS de terceros.

En el servidor solo se almacenan las URL.

Los recursos y paquetes solo utilizados por el cliente son cargados por cliente mediante un GET a la página oficial.

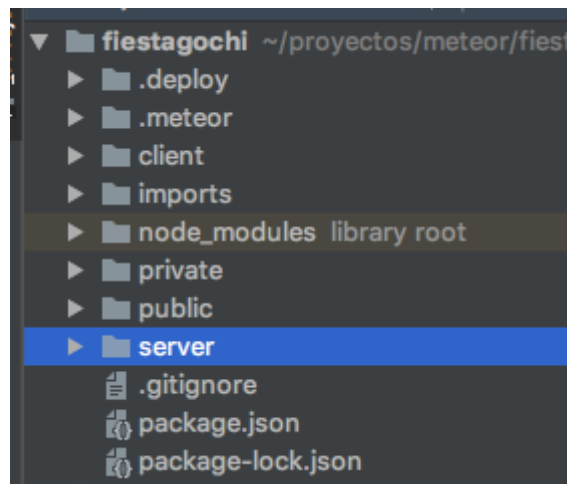
Bootstrap, fuentes,

12.2.6- ESTRUCTURA PROYECTO

Meteor cuenta con una estructura predefinida con la que se crea el proyecto, aunque existen multitud de estructuras.

Se ha utilizado esta estructura por los siguientes motivos:

- Me pareció una estructura bien organizada y clara.
- La documentación oficial utiliza esta estructura.
- Los paquetes oficiales (Página atmosphere) utilizan esta estructura.



- **.deploy**

Archivos para el despliegue en producción de la app.

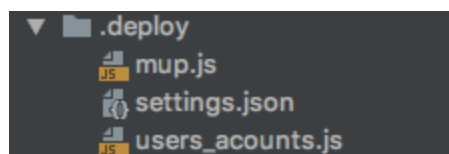
- **.meteor**

Archivos internos de Meteor

- **client**

Archivos que funcionan como punto de entrada para el cliente.

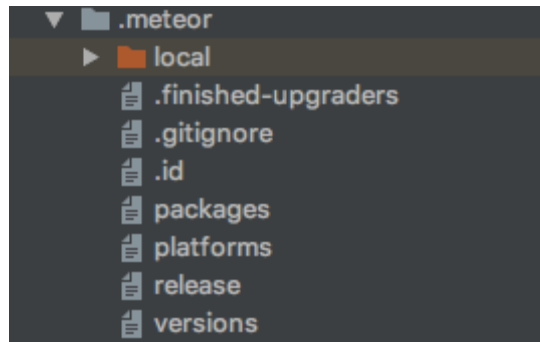
- **Imports**
Archivos con las APIS, configuraciones y componentes de la aplicación.
- **node_modules**
Módulos de nodeJS.
- **private**
Archivos accesibles solo desde el servidor.
- **public**
Archivos accesibles externamente.
Aquí irán las imágenes, fuentes...etc.
- **server**
Archivos que funcionan como punto de entrada para el servidor.
- **.gitignore**
Archivos excluidos del repositorio
- **package.json**
Paquetes y configuraciones.
- **package-lock.json**
Paquetes y configuraciones.



- **mup.js**
Configuración para el despliegue de la aplicación.
- **settings.json**
Configuraciones opciones del servidor.

- **users_accounts.js**

Cuentas de usuarios por defecto.



- **local**

Archivos creados durante la ejecución en local.

Entre ellos se incluyen las colecciones.

- **.finished-upgraders**

Configuraciones para la actualización de Meteor y packages.

- **.gitignore**

Archivos excluidos del repositorio.

- **.id**

Id generado para nuestro proyecto.

- **packages**

Paquetes instalados junto a su versión

- **platforms**

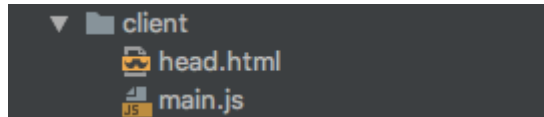
Plataformas en las que se ejecuta

- **release**

Datos de interés

- **versions**

Versiones



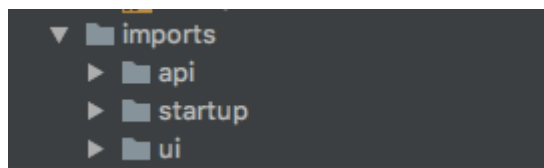
- **head.html**

Contenido de etiqueta head en el HTML.

- **main.js**

Primer fichero que se carga en el cliente al entrar.

Desde aquí se llama e incluye la estructura necesaria para el cliente.



- **api**

APIS de la aplicación.

Publicaciones y métodos.

- **startup**

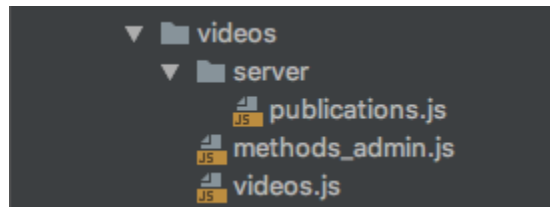
Lo primero en ejecutarse en el cliente al entrar.

Lo primero en ejecutarse por el servidor al arrancar.

Contiene triggers, configuraciones, rutas, inicialización de BD, crons...

- **ui**

Componentes de la web.



- **videos**

API videos

- **methods_admin.js**

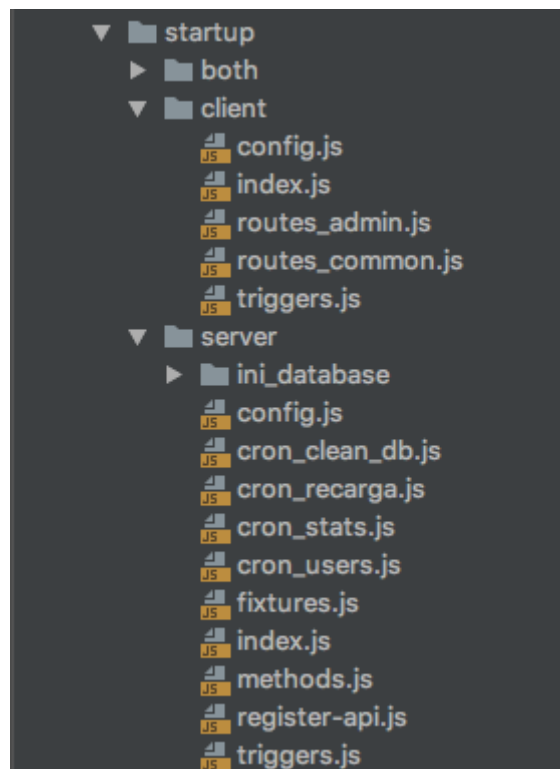
Métodos llamados por el cliente para modificar datos en la colección.

- **videos.js**

Definición de la colección.

- **Server/publications.js**

Publicaciones de esa colección.



- **both**
Archivos ejecutados en el servidor y cliente.
- **client**
Archivos ejecutados por el cliente
- **server**
Archivos ejecutados por el servidor
- **client/config.js**
Configuración del entorno del cliente
- **client/index.js**
Punto de entrada al directorio, es el primer fichero en ser llamado.
Desde este fichero se llama al resto de ficheros del directorio.

- **routes_admin.js**
Rutas para usuarios con rol admin
- **routes_common.js**
Rutas comunes
- **client/triggers.js**
Triggers del usuario.
- **ini_database.js**
Ficheros con los datos usados para la carga inicial de datos.
- **server/config.js**
Configuración del entorno del servidor.
- **cron_clean_db.js**
Configuración del cron encargado de las tareas de mantenimiento de la base de datos.
- **cron_stats.js**
Configuración del cron encargado de las tareas de actualización de los stats de las partidas.
- **cron_recarga.js**
Configuración del cron encargado de las tareas de actualización de las recargas de las partidas.
- **fixtures**
Punto de entrada donde se llama a los archivos de ini_database.
- **Index.js**
Punto de entrada al directorio, es el primer fichero en ser llamado.
Desde este fichero se llama al resto de ficheros del directorio.

- **methods.js**

Métodos llamados por el cliente para realizar funciones que no pertenecen a ningún api y que no pueden ser ejecutados en entorno cliente.

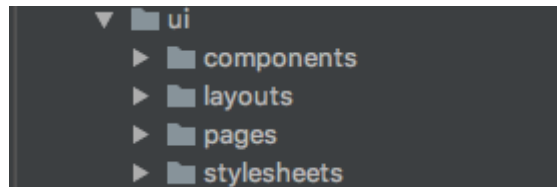
- **register-api.js**

Fichero donde se incluyen las APIS.

Se incluyen la API, publicaciones y métodos.

- **server/triggers.js**

Triggers del servidor.



- **components**

Componentes de la aplicación utilizados en las pages.

- **layouts**

Layouts de la aplicación, donde se renderizan las pages.

- **pages**

Página por ruta.

Cada página incluye los componentes necesarios.

En cada página se definen las subscripciones.

- **stylesheets**

Estilos y JS necesarios para la aplicación.

12.2.7- PAQUETES

- **meteor-base**

<https://atmospherejs.com/meteor/meteor-base>

Core de Meteor.

- **mobile-experience**

<https://atmospherejs.com/meteor/mobile-experience>

Paquete para optimizar el aspecto en móviles.

- **blaze-html-templates**

<https://atmospherejs.com/meteor/blaze-html-templates>

Paquete para la compilación de HTML. con Blaze.

- **estándar-minifier-css**

<https://atmospherejs.com/meteor/standard-minifier-css>

Paquete para minificar css

- **estándar-minifier-JS**

<https://atmospherejs.com/meteor/standard-minifier-js>

Paquete para minificar css

- **es5-shim**

<https://atmospherejs.com/meteor/es5-shim>

Paquete para la compatibilidad con versiones ES5 de JS en los navegadores que no soporten ES6.

- **ecmascript**

<https://atmospherejs.com/meteor/ecmascript>

Paquete con las funciones de la última versión de JS.

- **shell-server**

<https://atmospherejs.com/meteor/shell-server>

Paquete para optimizar la línea de comandos.

- **kadira:flow-router**

<https://atmospherejs.com/kadira/flow-router>

Paquete para definir el router

- **kadira:blaze-layout**

<https://atmospherejs.com/kadira/blaze-layout>

Paquete para renderizar dinámicamente las templates y configurar el árbol principal del DOM del que cuelgan las templates.

- **accounts-ui**

<https://atmospherejs.com/meteor/accounts-ui>

Paquete para la gestión de cuentas y usuarios.

- **accounts-password**

<https://atmospherejs.com/meteor/accounts-password>

Paquete para agregar el servicio de login con contraseña a accounts-ui

- **email**

<https://atmospherejs.com/meteor/email>

Paquete para el envío de emails.

- **momentjs:moment**

<https://atmospherejs.com/momentjs/moment>

Paquete para la gestión de fechas y tiempos.

Paquete de nodeJS para la gestión de fechas y tiempos.

- **mrt:momento-timezone**

<https://atmospherejs.com/mrt/moment-timezone>

Paquete para cambiar el timezone del paquete moment.

Paquete de nodeJS para cambiar el timezone del paquete moment.

- **templating**

<https://atmospherejs.com/meteor/templating>

Paquete para compilar templates y crear nodos head y body.

- **mrt:cron-tick**

<https://atmospherejs.com/mrt/cron-tick>

Paquete para crear y configurar crons.

- **session**

<https://atmospherejs.com/meteor/session>

Paquete para crear y gestionar la colección session en el cliente.

- **mdg:geolocation**

<https://atmospherejs.com/mdg/geolocation>

Paquete para capturar la geolocalización del cliente.

- **rzymek:moment-locale-es**

<https://atmospherejs.com/rzymek/moment-locale-es>

Paquete para agregar la locale ES al paquete moment

- **mongo**

<https://atmospherejs.com/meteor/mongo>

Paquete para la gestión de la base de datos.

Incluye la configuración del entorno.

- **underscore**

<https://atmospherejs.com/meteor/underscore>

Paquete utilizado para la gestión de datos, similar a jQuery.

- **mup**

<http://meteor-up.com/getting-started.html>

Paquete de nodeJS para el despliegue en producción.

- **bcrypt**

<https://www.npmjs.com/package/bcrypt>

Paquete de nodeJS para el cifrado de datos.

- **bootstrap**

<https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css>

Estilos CSS.

<https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js>

Estilos JS.

- **Fuentes**

<https://fonts.googleapis.com/css?family=Pacifico>

Fuente pacifico.

12.3- POSIBLES AMPLIACIONES

- Externalizar imágenes.
- Mejorar el aspecto de la página, agregando animaciones y otros eventos.
- Agregar balanceador de carga.
- Agregar traducciones e idiomas.
- Agregar SEO.
- Agregar más logs y control sobre la aplicación.
- Agregar micro pagos.
- Agregar publicidad.
- Agregar tienda online de entradas a eventos musicales.
- Agregar tienda online de merchandaising.
- Agregar tienda online relacionado con la música.