

***Proyecto EA***

***Autores:***

Raúl González

Raúl Lorenzo

Jesús Tardío

Carlos Valdés

# Índice

# Introducción

Primeramente es de recibo explicar el contexto en el que se alcanza la idea de diseñar una App cómo Freepong. Dado que se nos impusieron funcionalidades del tipo localización GPS en la que interactuaran los usuarios, y tras barajar diferentes ideas que girarían en torno a ésta funcionalidad principal, llegamos al consenso de diseñar una aplicación orientada a la interactuación entre las personas con una afición en común.

Freepong da la oportunidad de poder hacer quedadas de aficionados al ping pong aprovechando el mobiliario urbano de la ciudad y poder apuntarte a partidas en tiempo real. La localización de las mesas disponibles viene dada ya por la App, ingresadas estas desde el perfil de administrador (que también hemos implementado) dando la libertad y la facilidad a los usuarios registrados de directamente comprobar en la localización de la mesa si hay partidas disponibles, o si en cambio, este mismo puede crear la suya.

Para generar más aliciente a la App en sí, hay disponible un historial de victorias de las partidas cómo desarrollaremos más adelante a lo largo de esta memoria.

Creemos que el punto fuerte de esta App es que se puede extrapolar este sistema a diferentes deportes o aficiones que se desarrollen en una localización fija, dando así una escalabilidad y probabilidad de crecimiento bastante amplia, con relativo poco esfuerzo de desarrollo.

# Objetivos del proyecto

1. **Pila MEAN (MongoDB/Mongoose, Express, Angular i NodeJS)**

Uno de los objetivos principales en el desarrollo de sobretodo la parte Backend aunque también de la Frontend, es decir la API en sí, el motor, la inteligencia de la aplicación ha sido trabajar por primera vez con la pila MEAN.

Hemos podido trabajar y desarrollar una aplicación web sólida basada en esta combinación de diferentes sistemas en los que, una vez bien coordinados y sincronizados, es muy sencillo añadir plugins y funcionalidades.

El sistema de BBDD MongoDB o su integración en la API Mongoose, facilitan mucho el desarrollo a la hora de realizar cualquier tipo de consulta ya que se basa en arquitectura JSON, de forma que la estructura de los datos es más libre y no tan estricta como en cursos anteriores en los que se usaban lenguajes como MySQL.

NodeJS y Express hacen posible la incusión de librerías y estándares desde el inicio del proyecto y permite también, unificar en caso de desarrollo diversificado en diferentes equipos, que todo el mundo trabaje con las mismas versiones. También hay que destacar la utilización del controlador de versiones Bower, que realiza funciones similares a las de Node y Express y en nuestro caso hemos usado para mantener la versión del lenguaje principal AngularJS.

Hay que mencionar también el uso de Bower que es una herramienta para la gestión de librerías. La hemos utilizado sobre todo para el mantenimiento de versiones de las dependencias de la parte FrontEnd del proyecto.

Por último, todo el diseño del FrontEnd ha sido implementado con AngularJS en el que hemos podido comprobar la evolución de JAVAScript y lo realmente escalable.

1. **Aplicaciones híbridas IONIC/CORDOVA**

Una de las particularidades principales de este proyecto es su versión de aplicación para SmartPhones. Ionic es un SDK completo de software libre basado en lenguaje AngularJS y Apache Cordova con el que se puede realizar un desarrollo usando tecnologías web tipo HTML5 y CSS.

En nuestro proyecto se ha integrado un módulo Ionic que “ataca” al servicio principal de nuestra API funcionando de forma paralela.

Éste objetivo nos parece particularmente interesante ya que acerca y simplifica el desarrollo de las aplicaciones para Smartphones con un lenguaje más cercano a nuestros conocimientos, sabiendo de la complejidad de otros lenguajes nativos de los teléfonos inteligentes tipo Android y iOS.

1. **Aplicación RESTFUL**

Como en asignaturas anteriores, en este proyecto también se ha impuesto el sistema RESTFUL que se basa en una estructura cliente/servidor sin estado es decir, que cada mensaje HTTP contiene toda la información necesaria para comprender la petición.

En las diferentes funcionalidades de nuestro proyecto se han aplicado las 4 operaciones principales de los sistemas RESTFUL como son las operaciones CRUD, principalmente los métodos POST, PUT, DELETE y GET para realizar todas las operaciones contra el servidor.

1. **Geolocalización**

Este es un apartado en el que ninguno de los integrantes del equipo hemos desarrollado funcionalidades anteriormente. Dado que la funcionalidad y misión principal de nuestra App gira en torno a esta nueva posibilidad hemos querido apostar sobre seguro y hemos integrado el motor de Goggle Maps en nuestro proyecto. En el desarrollo punto por punto detallaremos la manera de hacer funcionar esta potente herramienta y cómo adaptarla a nuestra App.

Lo interesante de incluir este plugin es que ya funciona como un sistema independiente, es decir que conserva la gran mayoría de sus funcionalidades que le han hecho tan útil y popular como la Street View, por destacar un ejemplo.

1. **SCRUM**

En este proyecto otro de los objetivos ha sido el aprendizaje de uso de las herramientas de SCRUM como Pivotal Tracker a partir de la cual hemos podido ir planteando las fases de desarrollo acotadas en sprints de trabajo. Hemos podido comprobar que este tipo de herramientas son útiles para controlar en todo momento el avance y el progreso de cada una de las fases de implementación de las diferentes funcionalidades. También, por otro lado, se ha ido actualizando un blog de proyecto de donde se puede ir sacando la “foto” del avance del proyecto en cada momento, o en los eventos más destacados del mismo.

También hay que destacar la herramienta GitHub que hemos integrado totalmente en nuestro proyecto. Es básicamente un repositorio enfocado para la gestión de proyectos de desarrollo de aplicaciones. De esta forma hemos podido coordinar las aportaciones de cada uno de los miembros y actualizar en tiempo real todos los avances.

1. **Objetivos adicionales**
   1. **Login vía red social**

Tanto en la versión Web como móvil se ha incluído la posibilidad de realizar el registro automático en el sistema vía accesos de red social tipo Facebook y Twiter. De esta forma hemos querido facilitar el acceso a FreePong con una funcionalidad que nos parece interesante y actual. Este tipo de registro se incluye hoy día en la gran mayoría de las Apps sean del tipo que sean y en la nuestra no podía faltar.

Detallaremos como incluir esta funcionalidad y las librerías y módulos necesarios.

Aclarar que existe la posibilidad de crear usuarios en el sistema de forma tradicional a través del formulario de registro web a través del que se detallan todos los datos relevantes de usuario y se realiza el alta en el sistema.

* 1. **Chat**

Con los conocimientos asumidos de asignaturas anteriores, y ya que nuestro proyecto tiene el talante integrador de una red social, nos ha parecido interesante incluir esta funcionalidad, creando un chat de comunidad de FreePong basado en la estructura de sockets de conexión.

De esta forma pueden participar todos los usuarios registrados en el sistema a modo de foro en tiempo real.

En resumen los objetivos principales son los que hemos destacado a parte también mencionar, el trabajo en equipo y la repartición de tareas así como la coordinación y el seguimiento tanto para nosotros como para el profesorado a través de las entregas de sprints y la gestión de herramientas de SCRUM.

# Requerimientos

En este apartado vamos a definir todos los requerimientos y sus funcionalidades implementadas en forma de historieta, tal y como se nos ha indicado para el presente informe.

Le daremos dos puntos de vista distintos ya que existen dos perfiles de uso distintos, el de usuario registrado (Dani) y el perfil del administrador de la App (Pepe). También, a lo largo de la enumeración de las funcionalidades etiquetaremos en cuales de las vistas están disponibles tanto WEB como en la versión móvil híbrida de IONIC.

Perfil usuario registrado [WEB/IONIC]

* Dani es un gran aficionado al ping pong que tiene dificultades para encontrar diferentes personas para compartir su pasión.  Esta situación cambia cuando descubre y empieza a utilizar la aplicación **FreePong**. A partir de ese momento podrá:
  + visualizar en un mapa la lista de mesas de ping pong públicas más cercanas dentro de su perfil [WEB/IONIC]
  + comprobar seleccionando la mesa que más le convenga si existen partidas creadas por otros usuarios en el horario separado por espacios de una hora y de 8 a 20h [WEB/IONIC]
  + también, si no existe ninguna partida, Dani tiene la posibilidad de crear su propia partida y poder retar a otro usuario de FreePong en cualquiera de las franjas horarias disponibles [WEB/IONIC]
  + una vez acabada la partida, podrá registrar el resultado en la partida creada por él y después poder consultarlo en su historial paginado de partidas [WEB/IONIC]
  + crear su perfil con datos personalizados desde la web de registro o simplemente acceder vía red social [Facebook – WEB/IONIC , Twitter – IONIC]
  + Dani podrá editar los datos de su perfil en todo momento (excepto el nombre de usuario que es su identificador único en el sistema) en el apartado de edición de perfil [WEB/IONIC]
  + Si Dani decide no continuar usando FreePong, podrá eliminar su perfil en todo momento desde el menú de navegación [WEB/IONIC]
  + Dani puede comprobar su geolocalización actual para saber por qué zona buscar mesas disponibles [IONIC]
  + Dani también tiene curiosidad por ver los perfiles de los otros usuarios de FreePong y filtrar la búsqueda por cualquiera de sus datos [IONIC]
  + Dani puede chatear con la comunidad de usuarios de FreePong logueados en ese momento a través de nuestra herramienta de chat [IONIC/WEB]

Perfil administrador [WEB]

* Pepe acaba de lanzar con su equipo de desarrolladores la aplicación FreePong App en la que han puesto mucho empeño e ilusión. Para dar el mejor servicio y mantenimiento a los usuarios de la aplicación tanto Pepe como su equipo han habilitado las siguientes funcionalidades en su perfil de administrador:
  + Visualizar y gestionar todas las partidas creadas por los usuarios de la aplicación en ese momento con la posibilidad de filtrar por diferentes campos para facilitar su búsqueda
  + Tanto Pepe como todos los integrantes del equipo de desarrollo, tienen la posibilidad de gestionar las mesas existentes y disponibles para los usuarios de la App
  + Pueden también añadir más mesas disponibles con su imagen identificadora al sistema, para que luego aparezcan en el mapa y se puedan seleccionar
  + Pepe quiere controlar en todo momento la cantidad y los datos de los usuarios que están registrados en el sistema. En el perfil de administrador tiene la opción de visualizarlos todos en una lista paginada, filtrar para la búsqueda por cualquiera de los campos y eliminar perfiles en caso de ser necesario

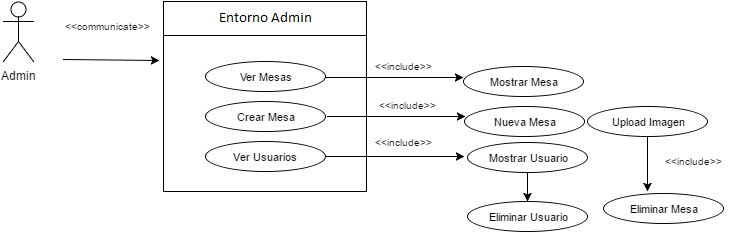
Requerimientos futuribles

* Es el primer gran proyecto de Pepe y su equipo y valorando la aceptación del público FreePong puede albergar requerimientos y funcionalidades del tipo:
  + Creación de torneos entre usuarios registrados y ordenados por ranking
  + Funcionamiento de red social con posibilidad de tener grupo de “Amigos” integrantes del sistema
  + Visualización de partidas o vídeos por streaming
  + Pepe y su equipo, dependiendo del éxito del funcionamiento de este primer proyecto, pueden extrapolar y adaptar este sistema a cualquier otro deporte en el que se puedan reaprovechar material urbano
  + Incuir saldo para igreso en actividades especiales

# Análisis de requerimientos

En este apartado vamos a basarnos en el diagrama UML de casos de uso para poder mostrar de forma gráfica la forma en la que interactúa el usuario, tanto el perfil de usuario registrado como el de administración, con nuestro sistema.

Ilustración - Diagrama de casos de uso admin



En este diagrama se pueden observar todas las opciones que tiene disponible el administrador desde su perfil.

Estas funciones están centradas en el mantenimiento y supervisión básicos de la aplicación, como comprobación de datos de usuario, gestión de la mesas y de las partidas que han tenido lugar entre los diferentes jugadores.

Casos de uso user.pngAquí se pueden ver las funcionalidades de las que pueden disfrutar los usuarios de FreePong así como la interactuación con otros usuarios en determinados apartados como son, el chat o a la hora de formar una partida conjunta con otro jugador.

Ilustración - Diagrama de casos de uso usuario

# Diseño

A continuación mostraremos la estructura de nuestra aplicación a través del diagrama de clases. Se pueden identificar todos los atributos y métodos utilizados y disponibles en cada una de las clases.

También se incluye la relación de dependencia entre cada una de ellas y las clases que utilizan a otras para poder asignar datos a sus atributos propios.

Después incluiremos un diagrama de los componentes de la aplicación, para poder ver cómo interactúan las clases previamente definidas entre sí.

Estos diagramas forman parte de la documentación estándar de hoy día de cualquier proyecto de esta índole, y de esta manera queremos ofrecer y equipararnos a proyectos profesionales.

Diagrama de clases.png

Ilustración - Diagrama de clases

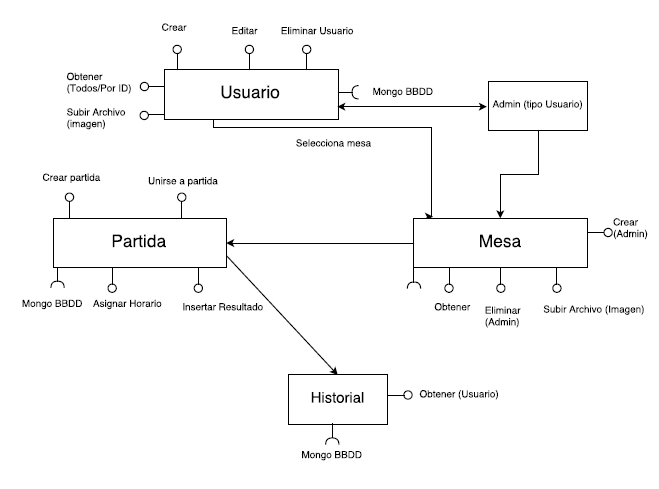


Ilustración - Diagrama de componentes

# Implementación: Descripción de las librerías, frameworks y herramientas utilizadas

Primeramente adjuntamos el contenido de los archivos “package.json” que son los que contienen el listado de dependencias y librerías que se han de cargar para poder desarrollar nuestro proyecto y sus versiones. Haremos un breve comentario de la utilidad de las más destacables.

Gracias a Node Express y la herramienta Bower que hemos mencionado anteriormente, hemos podido mantener todos los miembros del grupo al día todas las funcionalidades. Simplemente con actualizar la versión a través de GitHub en local y luego con las instrucciones “npm install” y “bower install” (esta última en el directorio de la parte frontend-cliente) podemos descargarnos e instalar todas estas librerías.

**"body-parser"**: **"~1.13.2"** -> Manejo de entornos con Body múltiple  
**"cookie-parser"**: **"~1.3.5"** -> Habilita uso de coockies en entorno web  
**"cors"**: **"^2.7.1"** -> Gestión de conexión dinámica   
**"debug"**: **"~2.2.0"**  
**"express"**: **"~4.13.1"**  
**"express-session"**: **"^1.13.0"**  
**"formidable"**: **"^1.0.17"** -> Módulo de gestión de archivos  
**"jade"**: **"~1.11.0"**  
**"method-override"**: **"^2.3.5" ->** Habilita PUT y DELETE en cliente  
**"mongoose"**: **"^4.4.10" ->** Módulo de gestión de Mongo BBDD **"mongoose-middleware"**: **"^0.2.17"**  
**"morgan"**: **"~1.6.1"**  
**"passport"**: **"^0.3.2"**  
**"passport-facebook"**: **"^2.1.0"** -> Intercambio de datos con Facebook  
**"serve-favicon"**: **"~2.3.0"**  
**"socket.io"**: **"^1.4.6" ->** Dependencia de gestión de sockets

**"dependencies"**: {  
 **"gulp"**: **"^3.5.6"**  
 **"gulp-sass"**: **"^2.0.4"**  
 **"gulp-concat"**: **"^2.2.0"**  
 **"gulp-minify-css"**: **"^0.3.0"**  
 **"gulp-rename"**: **"^1.2.0"**},  
**"devDependencies"**: {  
 **"bower"**: **"^1.3.3" ->** Gestión de librerías Bower  
 **"gulp-util"**: **"^2.2.14"**  
 **"shelljs"**: **"^0.3.0"**},  
**"cordovaPlugins"**: [  
 **"cordova-plugin-device"**  
 **"cordova-plugin-console"**  
 **"cordova-plugin-whitelist"**  
 **"cordova-plugin-splashscreen"**  
 **"cordova-plugin-statusbar"**  
 **"ionic-plugin-keyboard"**  
 **"org.apache.cordova.file"** -> Módulo de gestión de archivos  
 **"org.apache.cordova.file-transfer"** -> Módulo de gestión de archivos  
 **"cordova-plugin-inappbrowser"**]

Los entornos o frameworks y herramientas utilizados por los miembros del equipo son los siguientes:

* WebStorm
* Google Chrome
* Git Bash
* Notepad ++
* GitHub
* Putty
* Postman

# Conclusiones

# Referencias

<https://www.npmjs.com>