Nombre: Jose Miguel Gómez Lozano
Correo: josemygel@gmail.com || jmgl18@alu.ua.es
DNI: 48833504L
Fecha: 3/11/2018

# CyberApp

Este proyecto consistirá en una aplicación para dispositivos móviles con una parte de página web.

*CyberApp* consiste en una herramienta para aprendizaje sobre seguridad informática, desde el nivel más básico (la composición de las computadoras) hasta niveles más avanzados (tipos de encriptación, vulnerabilidades en sistemas, pentesting, etc).

En un principio, lo que haremos para la primera práctica será un servicio de Rest API en el que se pueda tener una cuenta, registrarse, iniciar sesion, cambiar la imagen de perfil, ver nivel en las materias (sesiones superadas * su valoración / el total de la suma de las valoraciones), etc.

Para comenzar, crearemos un proyecto con NodeJS e iniciaremos el NPM.
Como base de datos hemos seleccionado MongoDB, por varias razones.
La principal es que MongoDB trabaja con JSON, a parte de ser una base de datos basada en NoSQL.

Próximamente: En la página web se publicarán juegos a resolver, cuya contestación deberá realizarse a través de la APP para ganar experiencia y subir de nivel.

Nombre: Jose Miguel Gómez Lozano
Correo: josemygel@gmail.com || jmgl18@alu.ua.es
DNI: 48833504L
Fecha: 3/11/2018

# Composición del proyecto:

**BASE DE DATOS: MONGODB** **PRUEBAS UNITARIAS: POSTMAN**

==========

Las pruebas se han realizado con POSTMAN, y la base de datos a utilizar es MONGODB. La base de datos se inicializará automáticamente y las pruebas crean automáticamente los datos para trabajar con ellos.

Las pruebas estarán en el directorio /tests/.
La base de datos habrá que instalarla, se usa la configuración por defecto.

**CARPETAS DEL PROYECTO**

==========

**/test/** En esta carpeta hemos metido un archivo .json de POSTMAN en el que hemos metido todas las pruebas (las finales fallan por estar en pleno desarrollo).

**/CyberApp/** En esta carpeta hemos metido todo el proyecto, y dentro de este tenemos:

**/CyberApp.js** Este archivo es el ejecutable principal del proyecto.

**/app/** Contiene toda la información del proyecto modularizado por carpetas, por un lado las rutas, por otro los middlewares, por otro los controladores y por otro los schemas de mongodb, para tener un código repartido de forma más eficiente.

**/config.js** Contiene el token secreto.

**/config/** Contiene la configuración de conexión con la base de datos.

Nombre: Jose Miguel Gómez Lozano
Correo: josemygel@gmail.com || jmgl18@alu.ua.es
DNI: 48833504L
Fecha: 3/11/2018

# Entradas a la REST API:

| **ENTRADA** | **MÉTODO:USO** |
|---|---|
| / | GET: TODO el mundo puede acceder a la página inicial. |
| /login/ | POST: Solo usuario anónimo puede iniciar sesión. |
| /register/ | POST: Solo el usuario anónimo puede registrarse. |
| /users/ | GET: Solo los administradores pueden ver la lista de usuarios. |
| /me/ | GET: Ver la información del usuario logueado.<br>PUT: Actualizar la información del usuario logueado.<br>DELETE: Eliminar cuenta de usuario. |
| /threads/ | GET: Todos los usuarios pueden ver los hilos.<br>POST: Solo los usuarios logueados pueden crear hilos. |
| /threads/:idThread | GET: Todos los usuarios pueden ver un hilo concreto.<br>POST: Solo los registrados pueden **COMENTAR** en los hilos.<br>PUT y DELETE: Solo el autor puede editar y borrar el hilo. |
| /subjects/ | GET: Todos pueden ver los temas (pero solo los de principiantes).<br>POST, PUT y DELETE: Solo administradores. |
| /subjects/:idSubject | GET: Para ver temas altos deben superar los bajos y estar logueados.<br>POST: Se usará para contestar a las preguntas de un tema.<br>PUT y DELETE: Solo administradores.<br>(Contestación de temario no implementada todavía) |

Debido a la implementación adicional, quizás *subject* o *threads* presenten algunos errores y las pruebas no se realicen TODAS correctamente.

Nombre: Jose Miguel Gómez Lozano
Correo: josemygel@gmail.com || jmgl18@alu.ua.es
DNI: 48833504L
Fecha: 3/11/2018

# AUTENTICACIÓN:

Como pone en la documentación, la autenticación de usuarios se realizará con jwt-simple, haciendo uso de la siguiente estructura:

```
payload  = {
        //username contiene el nombre de usuario con el que identificarle
        username: "NOMBRE DE USUARIO",
        //expire contiene el tiempo en el que es válido el token
        expire: moment.add(2,'days'').valueOf()
}
```

Haremos uso de username para saber que usuario es el logueado, y de ahí obtener si es administrador, su id, etc.

Nombre: Jose Miguel Gómez Lozano
Correo: josemygel@gmail.com || jmgl18@alu.ua.es
DNI: 48833504L
Fecha: 3/11/2018

# EXÁMENES:

# (FUTURAS MEJORAS)

Ideas: varios modos y varios juegos

Modos:

**-Selection**: Seleccionar que partes de un código están mal escritas, se mandará un array con las líneas seleccionadas {“solution”:”*[0,7,9,15]”}* en modo texto y se evaluará comparando la solución guardada en la base de datos.

**-Input**: Se rellenan los espacios restantes y se mete las respuestas en un array, para mayor simplificación serán de una línea *{“solution”:”[“#include<string.h>”,”String var;”,”var = 5;”]”}*, luego en el servidor, se trabajarán las soluciones para comprobar si están bien escritas, por ejemplo *String var;* necesita el espacio, pero *var = 5;* no, por lo que para evitar errores a *var = 5;* y a *#include <string.h>* se le eliminarían los espacios en el backend para su verificación.

Para acceder a un examen se haría con *exam/:idPart* y se elegiría uno de los exámenes que haya disponible para esa parte de forma aleatoria.