



“SISTEMA PARA EL MONITOREO, DETECCIÓN Y ALERTA DE  
SOMNOLENCIA DEL CONDUCTOR MEDIANTE VISIÓN ARTIFICIAL,  
COMUNICACIÓN INALÁMBRICA Y GEOLOCALIZACIÓN”

---

## Primer Reporte Parcial

---

### Lista de actividades

- Definir rutas del frontend
- Diseño de rutas del backend
- Conexión Backend con Mongo DB
- Sistema de acceso con credenciales
- Creación de la base de datos no relacional
- Investigación de modelos de Redes Neuronales Convolucionales
- Diseño de una red neuronal convolucional capaz de detectar ojos cerrados y abiertos

#### *Autores:*

Alan Eduardo Gamboa Del  
Ángel  
Maite Paulette Díaz Martínez

#### *Asesores:*

M.en C. Niels Henrik Navarrete  
Manzanilla  
Dr. Rodolfo Vera Amaro

# Índice

<b>1. Definir rutas del frontend</b>	<b>4</b>
1.1. Objetivo . . . . .	4
1.2. Descripción . . . . .	4
1.3. Resultados . . . . .	4
<b>2. Definir rutas del backend</b>	<b>5</b>
2.1. Objetivo . . . . .	5
2.2. Descripción . . . . .	5
2.3. Resultados . . . . .	5
<b>3. Conexión Backend con Mongo DB</b>	<b>6</b>
3.1. Objetivo . . . . .	6
3.2. Descripción . . . . .	6
3.3. Resultados . . . . .	8
<b>4. Sistema de acceso con credenciales</b>	<b>9</b>
4.1. Objetivo . . . . .	9
4.2. Descripción . . . . .	9
4.3. Resultados . . . . .	9
<b>5. Creación de la base de datos No Relacional</b>	<b>10</b>
5.1. Objetivo . . . . .	10
5.2. Descripción . . . . .	10
5.3. Resultados . . . . .	10
<b>6. Investigación de modelos de Redes Neuronales Convolucionales</b>	<b>11</b>
6.1. Objetivo . . . . .	11
6.2. Descripción . . . . .	11
6.3. Resultados . . . . .	11
<b>7. Diseño de una red neuronal convolucional capaz de detectar ojos cerrados y abiertos</b>	<b>12</b>
7.1. Objetivo . . . . .	12
7.2. Descripción . . . . .	12
7.3. Resultados . . . . .	12
<b>8. Conclusiones</b>	<b>13</b>
<b>9. Bibliografía</b>	<b>14</b>

## Índice de figuras

1.	Creación proyecto de react . . . . .	4
2.	Creación proyecto de react . . . . .	4
3.	Página web MongoDB Atlas. . . . .	6
4.	Página Principal MongoDB . . . . .	6
5.	Página Creación de Cluster . . . . .	7
6.	Página Creación de Cluster . . . . .	7
7.	Directorio del Backend . . . . .	7
8.	Directorio del Backend . . . . .	8
9.	Directorio del Backend . . . . .	8
10.	Directorio del Backend . . . . .	8

## Índice de tablas

# 1. Definir rutas del frontend

## 1.1. Objetivo

Definir e implementar las rutas que tendrá la aplicación, así como si serán públicas o privadas y la información que se desplegará en cada una de las mismas.

## 1.2. Descripción

Como primer paso, se necesita crear un proyecto de React utilizando el siguiente comando:

```
alan@alan-Inspiron-5548:~/Documentos/eb$ npx create-react-app eb
```

Figura 1: Creación proyecto de react

Posteriormente, se realiza la instalación del moudelo *React Router Dom* utilizando el siguiente comando:

```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  GITLENS

alan@alan-Inspiron-5548:~/Documentos/eb$ npm i react-router-dom

added 3 packages, and audited 2824 packages in 13s

263 packages are looking for funding
  run `npm fund` for details

6 high severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
alan@alan-Inspiron-5548:~/Documentos/eb$
```

Figura 2: Creación proyecto de react

Dentro de nuestro proyecto creado, ingresamos al archivo *index.js*. Dentro de este archivo, importamos dos librerías ofrecidas por el módulo de *react router dom* utilizando la línea de código siguiente:

```
import {createBrowserRouter, RouterProvider} from "react-router-dom";
```

Figura 3: Creación proyecto de react

## 1.3. Resultados

## **2. Definir rutas del backend**

### **2.1. Objetivo**

Crear las rutas mediante las que el cliente realizará las peticiones y tendrá acceso a las operaciones, así como su funcionamiento en cuanto a obtención de datos y comunicación con el resto de la aplicación.

### **2.2. Descripción**

### **2.3. Resultados**

## 3. Conexión Backend con Mongo DB

### 3.1. Objetivo

Realizar la conexión de NodeJs con la base de datos MongoDB.

### 3.2. Descripción

Como primer paso, se debe ingresar a la página web <https://mongodb.com> y registrarse.

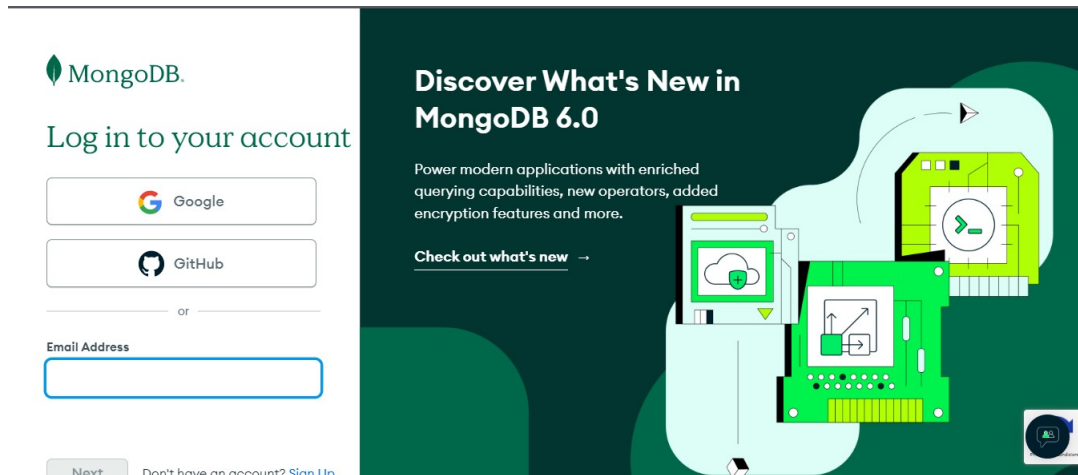


Figura 4: Página web MongoDB Atlas.

Después de haber iniciado sesión, daremos click al botón *Create* para crear un cluster al cuál podremos conectarnos para utilizar los servicios de Mongo Atlas.

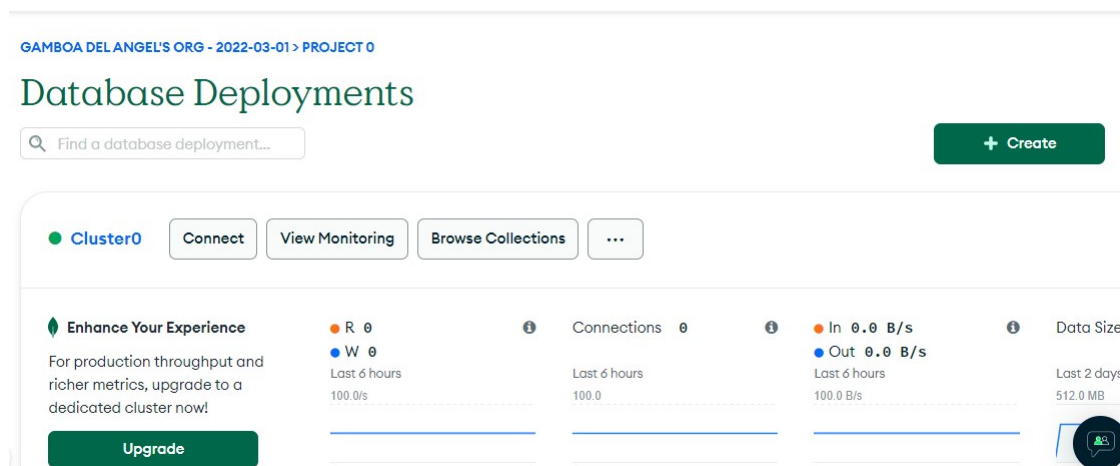


Figura 5: Página Principal MongoDB

Posteriormente, se seleccionarán los ajustes del cluster a crear, será un cluster compartido ya que esta opción es gratis.

CLUSTERS &gt; CREATE A SHARED CLUSTER

## Create a Shared Cluster

Serverless

Dedicated

Shared

For learning and exploring MongoDB in a sandbox environment. Basic configuration controls.  
No credit card required to start. Upgrade to dedicated clusters for full functionality.  
Explore with sample datasets. Limit of one free cluster per project.

Cloud Provider & Region

AWS, N. Virginia (us-east-1)

aws

Google Cloud

Azure

★ Recommended region

🔒 Dedicated tier region

Figura 6: Página Creación de Cluster

De igual manera, se mantendrán los valores por default en cuanto al almacenamiento y la versión de Mongo a utilizar.

Cluster Tier

M0 Sandbox (Shared RAM, 512 MB Storage)  
Encrypted

Additional Settings

MongoDB 5.0, No Backup

Cluster Name

Cluster1

Figura 7: Página Creación de Cluster

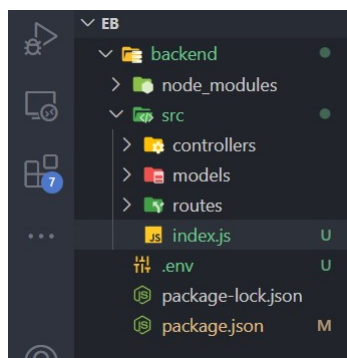


Figura 8: Directorio del Backend



```
backend > src > .\index.js > ...
1  const express = require("express");
2  const mongoose = require("mongoose");
3  require('dotenv').config({ path: 'env' });
4
5  const app = express();
6  const port = process.env.PORT || 9000;
7  const URI = 'mongodb+srv://root:250997@cluster0.zrhyx.mongodb.net/?retry
8
9  //iniciar servidor
10 app.get("/", (req,res) => {
11     res.send("Servidor Listo")
12 });
13
14
```

Figura 9: Directorio del Backend

```
15 //conexion mongo
16 mongoose.set("strictQuery", true);
17 mongoose.connect(URI, {
18     useNewUrlParser: true,
19     useUnifiedTopology: true
20 }).then(() => console.log("Conexión a mongo exitosa"))
21 .catch((error) => console.log(error));
22
23 //levantar servidor
24 app.listen(port,() => console.log("servidor escuchando en el puerto", port));
25
```

Figura 10: Directorio del Backend

### 3.3. Resultados

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER  COMMENTS  node

C:\Users\alang\Documents\TT\eb\backend>npm run start

> backend@1.0.0 start
> nodemon src/index.js

[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node src/index.js`
servidor escuchando en el puerto 9000
Conexión a mongo exitosa
```

Figura 11: Directorio del Backend

## **4. Sistema de acceso con credenciales**

### **4.1. Objetivo**

Establecer los roles de cada tipo de usuario con sus respectivos permisos de acceso a la aplicación web utilizando Amazon Cognito

### **4.2. Descripción**

### **4.3. Resultados**

## **5. Creación de la base de datos No Relacional**

### **5.1. Objetivo**

Crear la base de datos en MongoDB.

### **5.2. Descripción**

### **5.3. Resultados**

## **6. Investigación de modelos de Redes Neuronales Convolutionales**

### **6.1. Objetivo**

Determinar distintos modelos de redes neuronales convolucionales que ofrezcan mejor eficiencia al clasificar imágenes.

### **6.2. Descripción**

### **6.3. Resultados**

## **7. Diseño de una red neuronal convolucional capaz de detectar ojos cerrados y abiertos**

### **7.1. Objetivo**

Diseñar y realizar pruebas de los modelos de redes neuronales convolucionales previamente investigados para determinar el rendimiento y la precisión de cada uno.

### **7.2. Descripción**

### **7.3. Resultados**

## 8. Conclusiones

Para el desarrollo de la Estación base, se decidió utilizar la suite de herramientas de Amazon Amplify. Se hará uso de Amplify Hosting, el cuál tiene una integración directa con Github, esto quiere decir, que los cambios que se realicen en el repositorio se reflejarán de manera automática en la aplicación web. Además, AWS Amplify ofrece su servicio de almacenamiento en la nube S3, este será de gran ayuda para almacenar contenido multimedia, en este caso los videos de incidencia de los conductores. Para el manejo de credenciales, se utilizará Amazon Cognito, que se encargará de administrar las credenciales de acceso a la aplicación. Para el manejo de datos, se estará utilizando MongoDB, un manejador NoSQL que trabaja con documentos. Para el desarrollo de la aplicación, se decidió utilizar el lenguaje de programación Javascript, junto con NodeJs que nos ayudará a manejar varias peticiones al mismo tiempo. Finalmente el análisis del sistema de comunicaciones, en un principio sólo se había contemplado el análisis de telemetría pero al ir realizando las actividades del tercer reporte e ir profundizando en algunos temas de comunicaciones, se decidió que también que se necesitaba el análisis de la cobertura y de los datos, es decir, de la transmisión de los fotogramas por lo que se incluyeron también en este reporte.

## 9. Bibliografia

### Referencias