**Script del Sistema de Encuestas de Opinión con Visualización Gráfica**

**1. Introducción**

Este sistema es una aplicación web interactiva que permite realizar encuestas de opinión, almacenar las respuestas y visualizar los resultados mediante gráficos dinámicos. Está construida sobre el stack tecnológico **MEAN**, que incluye:

* **MongoDB**: Base de datos NoSQL utilizada para almacenar las encuestas y respuestas.
* **Express.js**: Framework de backend para la gestión de rutas y lógica de servidor.
* **Angular**: Framework frontend para crear una interfaz de usuario dinámica y modular.
* **Node.js**: Entorno de ejecución de JavaScript en el servidor.

**2. Arquitectura del Sistema**

La aplicación sigue una arquitectura **cliente-servidor** con una clara separación entre el frontend y el backend.

**2.1 Backend (Node.js + Express + MongoDB)**

Estructura de archivos:

* server.js: Punto de entrada del servidor Express.
* config/db.js: Configuración de conexión a MongoDB.
* routes/: Rutas REST para gestionar encuestas y respuestas.
* controllers/: Lógica de negocio.
* models/: Esquemas de datos de Mongoose.

**2.2 Frontend (Angular)**

Estructura de componentes:

* src/app/components/:
  + survey-form/: Formulario para enviar respuestas.
  + survey-results/: Visualización de resultados con gráficos.
* src/app/services/: Servicios para comunicación HTTP con el backend.
* src/app/models/: Interfaces de datos y clases auxiliares.

**3. Flujo de Trabajo**

1. El usuario completa una encuesta a través de Angular.
2. Angular envía los datos al backend usando HTTP.
3. El backend almacena la información en MongoDB.
4. Angular recupera los datos procesados del backend.
5. Se muestran resultados en gráficos usando **Chart.js**.

**4. Configuración de Base de Datos**

La conexión con MongoDB se realiza mediante Mongoose. Se puede usar tanto una instancia local como una instancia en la nube a través de **MongoDB Atlas**.

// backend/config/db.js

const mongoose = require('mongoose');

const connectDB = async () => {

try {

const mongoURI = process.env.MONGO\_URI || process.env.MONGODB\_URI || 'mongodb+srv://[credenciales]';

const conn = await mongoose.connect(mongoURI);

console.log(`MongoDB Connected: ${conn.connection.host}`);

} catch (error) {

console.error(`Error: ${error.message}`);

process.exit(1);

}

};

module.exports = connectDB;

**5. Componentes Principales**

**5.1 API RESTful**

El backend expone endpoints para:

* Crear nuevas encuestas.
* Guardar respuestas de los usuarios.
* Consultar resultados agregados.

**5.2 Visualización de Resultados**

El componente SurveyResultsComponent muestra los resultados de cada pregunta en formato gráfico:

// survey-results.component.ts (fragmento clave)

this.surveyService.getResults().subscribe({

next: (data) => {

this.results = data;

this.isLoading = false;

setTimeout(() => this.createCharts(), 0);

},

error: (error) => {

console.error('Error loading results:', error);

this.isLoading = false;

}

});

Este componente emplea **Chart.js** para renderizar gráficos circulares con colores dinámicos, leyendas e interactividad.

**6. Estilo y Diseño**

La interfaz está diseñada para ser limpia, moderna y adaptativa a dispositivos móviles:

.results-container {

max-width: 1200px;

margin: 0 auto;

padding: 2rem;

}

.charts-grid {

display: grid;

grid-template-columns: repeat(auto-fill, minmax(500px, 1fr));

gap: 2rem;

}

.chart-container {

background-color: #fff;

border-radius: 8px;

box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

}

**7. Variables de Entorno**

La configuración se controla mediante variables definidas en un archivo .env, que está excluido del repositorio por razones de seguridad:

NODE\_ENV=development

PORT=5000

MONGO\_URI=mongodb+srv://usuario:contraseña@cluster.mongodb.net/...

**8. Seguridad**

* Las credenciales sensibles están almacenadas en variables de entorno.
* El archivo .env está listado en .gitignore.
* La conexión a MongoDB se realiza mediante credenciales y URI seguros con autenticación encriptada.

**9. Conclusión**

Este sistema de encuestas proporciona una solución integral para la recolección y análisis visual de datos de opinión. Su diseño modular, arquitectura escalable y uso de tecnologías modernas permiten mantener, escalar y mejorar la aplicación con facilidad.

Además, la integración de **Chart.js** permite ofrecer visualizaciones atractivas y comprensibles para los usuarios finales, mejorando la experiencia e interpretación de resultados.