

**2024**

implementación de aplicación web que funciona el método Montessori con una plataforma web

**MANUAL TÉCNICO**

|  |
| --- |
| **QUETZALTENANGO** |
| **V.0.1** |

Para más información, por favor contactase

Allan Lizandro López López

(502) 3343 0250

alopezl62@miumg.edu.gt

Contenido

[Acerca del proyecto 4](#_Toc178806593)

[Requisitos del sistema 4](#_Toc178806594)

[Instalación y Configuración del Front-End y Back-End 6](#_Toc178806595)

[Front-End 6](#_Toc178806596)

[Back-End 7](#_Toc178806597)

[Estructura del Proyecto 9](#_Toc178806598)

[Frontend 9](#_Toc178806599)

[Backend: 9](#_Toc178806600)

[Base de datos (MongoDB): 10](#_Toc178806601)

[Funcionamiento General del Sistema 10](#_Toc178806602)

[Autenticación JWT: 10](#_Toc178806603)

[CRUD de Alumnos: 10](#_Toc178806604)

[Actividades Interactivas: 10](#_Toc178806605)

[Envío de Correos Electrónicos (Nodemailer): 10](#_Toc178806606)

[Diagramas UML 11](#_Toc178806607)

[Diagrama de arquitectura del sistema 11](#_Toc178806608)

[Diagrama de secuencia 12](#_Toc178806609)

[Diagrama de componentes 13](#_Toc178806610)

[Diagrama de actividades 14](#_Toc178806611)

[Diagramas de estados 15](#_Toc178806612)

[Diagrama de base de datos 16](#_Toc178806613)

[Diagrama de caso de uso 17](#_Toc178806614)

# 

# Acerca del proyecto

El proyecto proporciona una serie de actividades digitalizadas inspiradas en el método Montessori que su objetivo es motivar a los niños a seguir aprendiendo de la educación, pero de una manera más dinámica y atractiva a través de una plataforma web.

Este manual está dirigido a los desarrolladores y administradores del sistema, proporcionando instrucciones detalladas para la instalación, configuración y mantenimiento del sistema Montessori.

## Requisitos del sistema

1. Requisitos Mínimos:

Estos son los requisitos necesarios para que la aplicación funcione adecuadamente, pero con un rendimiento básico:

* CPU: Intel i3 (dual-core) a 2.0 GHz o equivalente.

Aunque la plataforma maneja animaciones (por ejemplo, LottieFiles) y actividades interactivas, no requiere un procesamiento intensivo. Un procesador de gama media puede manejar la carga básica.

* RAM: 4 GB.

Las actividades interactivas, aunque visualmente ricas, no consumen mucha memoria. Sin embargo, 4 GB es el mínimo para garantizar que la aplicación funcione sin cuellos de botella, especialmente si el sistema operativo también está utilizando RAM.

* Espacio en Disco: 5 GB.

Explicación: Los datos almacenados en MongoDB (alojado en la nube) no ocupan espacio local. El espacio en disco se utiliza principalmente para el almacenamiento de archivos temporales y las dependencias del sistema.

* Gráficos: GPU integrada con soporte para aceleración básica de gráficos.

Las animaciones Lottie y las actividades interactivas requieren algo de procesamiento gráfico, pero una GPU básica integrada es suficiente para el renderizado.

2. Requisitos Recomendados:

Estos requisitos ofrecen un rendimiento óptimo, permitiendo que la plataforma funcione sin interrupciones, incluso con múltiples usuarios o actividades simultáneas.

* CPU: Intel i5 o Ryzen 5 (quad-core) a 2.5 GHz o superior.

Un procesador de cuatro núcleos ayudará a manejar múltiples solicitudes simultáneamente, sobre todo si hay varios usuarios interactuando con la plataforma al mismo tiempo.

* RAM: 8 GB.

Aunque 4 GB son suficientes para operaciones básicas, 8 GB garantizan un rendimiento fluido incluso cuando hay múltiples pestañas abiertas o varios usuarios interactuando con actividades ricas en gráficos.

* Espacio en Disco: 10 GB o más.

Esto es ideal para manejar las dependencias del sistema y cualquier archivo temporal que se pueda generar al cargar o descargar contenido educativo.

* Gráficos: GPU dedicada o integrada con soporte para aceleración gráfica avanzada (Intel UHD o equivalente).

Explicación: Una GPU dedicada no es obligatoria, pero cualquier tarjeta gráfica moderna (incluso integrada) con soporte para aceleración gráfica avanzada mejora el renderizado de las animaciones y actividades.

3. Requisitos de Conectividad:

* Velocidad de Internet:

Mínima: 5 Mbps de descarga / 1 Mbps de subida.

Las actividades interactivas y animaciones requieren cargar recursos desde la base de datos y desde el frontend desplegado en Vercel. Aunque los recursos son optimizados, es necesario un ancho de banda mínimo para asegurar que no haya retrasos visibles.

* Recomendada: 10 Mbps de descarga / 5 Mbps de subida.

Una mejor conexión asegura que las actividades y el contenido multimedia se carguen sin problemas, incluso cuando hay múltiples usuarios interactuando con la plataforma.

1. Consideraciones del Sistema Operativo:

* Windows, macOS o Linux (versiones modernas y soportadas):

Cualquier sistema operativo que soporte las versiones recomendadas de Node.js y MongoDB Atlas funcionará adecuadamente con esta plataforma.

## Instalación y Configuración del Front-End y Back-End

### Front-End

Paso 1: Clonar el repositorio del frontend

Abre la terminal y clona el repositorio del proyecto frontend:

git clone <https://github.com/Allan07022003/frontend.git>

Navega a la carpeta del frontend:

“cd proyecto frontend”

Paso 2: Instalar las dependencias

Dentro del directorio del frontend, instala las dependencias de incluidas en el archivo packajge.json:

“npm install”

Paso 3: Despliegue del frontend en Vercel

“Instala la CLI de Vercel:”

“npm install -g vercel”

Ejecuta el comando para desplegar:

“vercel –prod”

Sigue las instrucciones para conectar el proyecto a Vercel. Una vez desplegado, la aplicación estará disponible en la URL que proporcione Vercel.

Paso 5: Actualización del frontend

Cuando quieras realizar cambios en el frontend, asegúrate de actualizar el repositorio con los nuevos commits:

“git add .”

“git commit -m "mensaje de descripción del commit" “

“git push origin main”

Ejecuta el comando de despliegue nuevamente:

“vercel –prod”

### Back-End

Instalación y Configuración del Backend (Node.js/Express en Heroku)

Paso 1: Clonar el repositorio del backend

1. Abre la terminal y clona el repositorio del backend:

git clone https://github.com/Allan07022003/backend.git

1. Navega a la carpeta del backend:

“cd backend”

Paso 2: Instalar las dependencias

Instala las dependencias del backend (Node.js) que están en el archivo package.json:

“npm install”

Paso 3: Configurar variables de entorno

Crea un archivo .env en la raíz del proyecto y añade las siguientes variables de entorno:

MONGO\_URI=<url-mongo>

JWT\_SECRET=<secreto-jwt>

EMAIL\_USER=<correo-electronico>

EMAIL\_PASS=<contraseña-correo>

**Paso 4: Despliegue del backend en Heroku**

1. Inicia sesión en Heroku:

“heroku login”

1. Crea una nueva aplicación en Heroku:

“heroku create nombre-app”

1. Configura las variables de entorno en Heroku:

heroku config:set MONGO\_URI=<tu-url-mongo>

heroku config:set JWT\_SECRET=<tu-secreto-jwt>

heroku config:set EMAIL\_USER=<correo-electronico>

heroku config:set EMAIL\_PASS=<contraseña-correo>

1. Despliega el backend a Heroku:

“git push heroku main”

Paso 5: Actualización del backend

Para actualizar el backend, realiza los cambios en el código y asegúrate de confirmar los nuevos commits:

“git add .”

“git commit -m "mensaje de actualización"

“git push origin main”

1. Despliega el backend actualizado a Heroku:

“git push heroku main”

## Estructura del Proyecto

### Frontend

src/components: Contiene todos los componentes React utilizados en la interfaz.

src/pages: Páginas principales de la plataforma.

src/styles: Estilos globales, utilizando Tailwind CSS.

### Backend:

src/controllers: Contiene los controladores para manejar las solicitudes HTTP.

src/models: Modelos de la base de datos (Mongoose).

src/routes: Define las rutas de la API.

### Base de datos (MongoDB):

Alumnos: Almacena los datos de los estudiantes.

Usuarios: Almacena los datos de autenticación (profesores y estudiantes).

## Funcionamiento General del Sistema

### Autenticación JWT:

El sistema utiliza JSON Web Tokens (JWT) para gestionar la autenticación. Una vez que el usuario inicia sesión, se genera un token JWT que es enviado al frontend y se almacena para gestionar la sesión de forma segura.

### CRUD de Alumnos:

El maestro puede realizar operaciones CRUD sobre los alumnos, como crear nuevos alumnos, ver la lista de estudiantes, actualizar su información o eliminar un registro.

### Actividades Interactivas:

Los estudiantes pueden acceder a diversas actividades interactivas de matemáticas, ciencias y lenguaje. Las actividades no tienen calificaciones, pero se registra si la actividad fue completada.

### Envío de Correos Electrónicos (Nodemailer):

Nodemailer se utiliza para enviar correos electrónicos automáticos a los profesores con notificaciones importantes, como nuevos registros o actualizaciones.

## Diagramas UML

### Diagrama de arquitectura del sistema

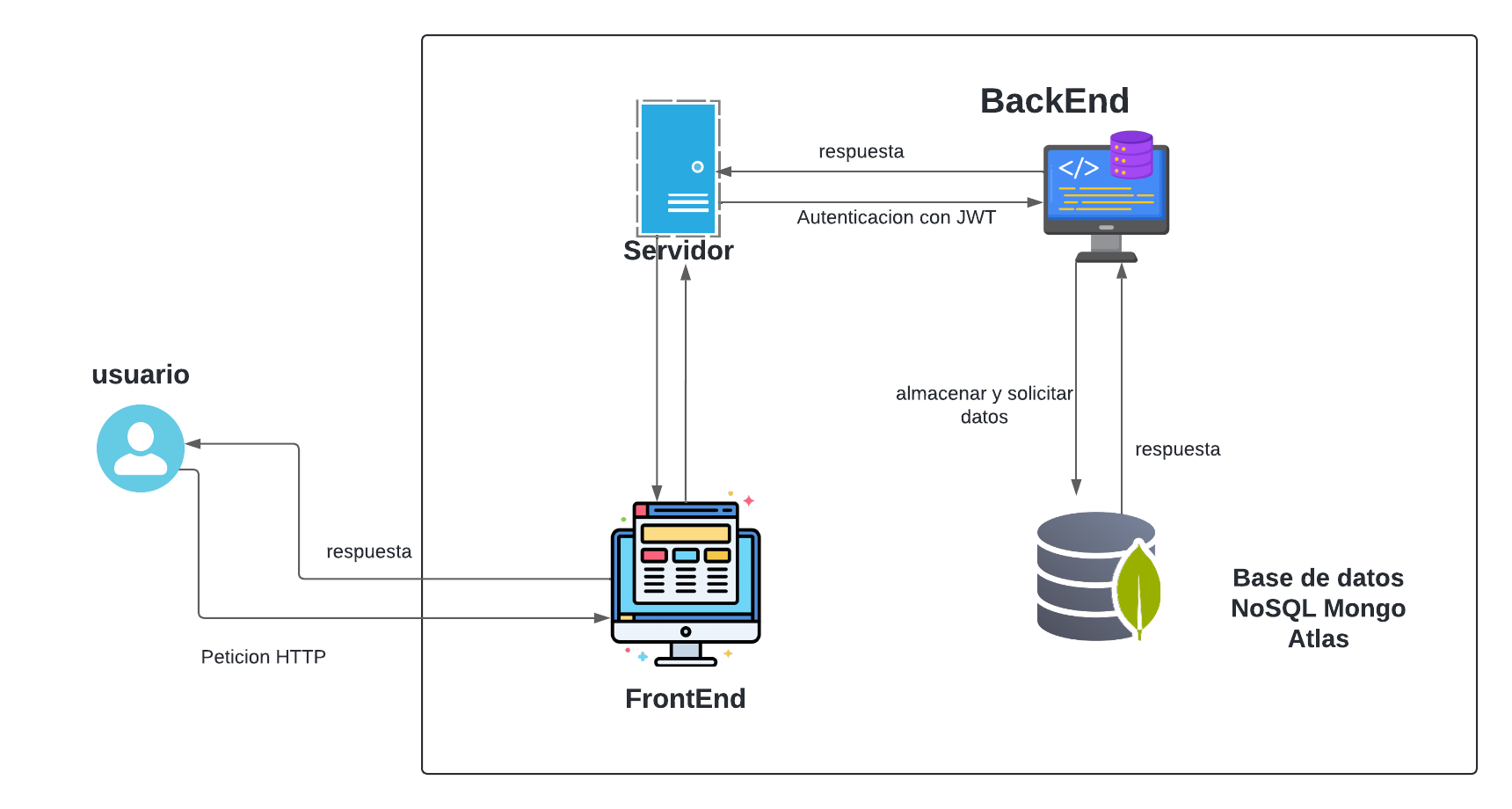


Ilustración 1: Diagrama de la arquitectura del sistema

### Diagrama de secuencia

Ilustración 2: Diagrama de secuencia

### Interfaz de usuario gráfica Descripción generada automáticamenteDiagrama de componentes

Ilustración 3: Diagrama de componentes

### Diagrama Descripción generada automáticamenteDiagrama de actividades

Ilustración 4: Diagrama de actividades

### Diagramas de estados

#### Imagen cargadaDiagrama de Inicio de sesión

Ilustración 5: Diagrama de inicio de sesión

#### Diagrama de actividad completada

Ilustración 6: diagrama de actividad completada

### Tabla Descripción generada automáticamente con confianza bajaDiagrama de base de datos

Ilustración 7: Diagrama de base de datos

### Diagrama de caso de uso

Diagrama

Descripción generada automáticamente

Ilustración 8: Diagrama de caso de uso