**Documentación de la Aplicación de Gestión de Tareas**

**Descripción General**

Esta aplicación de gestión de tareas es una aplicación web completa desarrollada con el stack MEAN (MongoDB, Express.js, Angular, Node.js). Permite a los usuarios gestionar tareas diarias mediante funcionalidades para crear, leer, actualizar y eliminar tareas (operaciones CRUD).

**Arquitectura**

La aplicación sigue una arquitectura cliente-servidor:

1. **Frontend**: Desarrollado con Angular, proporciona una interfaz de usuario reactiva que se comunica con el backend.
2. **Backend**: API RESTful desarrollada con Node.js y Express.js que maneja las solicitudes del cliente.
3. **Base de datos**: MongoDB Atlas alojado en la nube, para almacenamiento persistente de las tareas.

**Estructura del Proyecto**

gestion\_tareas/

├── backend/

│ ├── models/

│ │ └── Task.js

│ ├── routes/

│ │ └── tasks.js

│ └── server.js

└── gestor-tareas/

├── src/

│ ├── app/

│ │ ├── components/

│ │ │ ├── task-list/

│ │ │ └── task-form/

│ │ ├── services/

│ │ │ └── task.service.ts

│ │ ├── app.routes.ts

│ │ └── app.component.\*

│ ├── index.html

│ └── main.ts

**Modelo de Datos (Task.js)**

const mongoose = require('mongoose');

const taskSchema = new mongoose.Schema({

title: {

type: String,

required: true,

trim: true

},

description: {

type: String,

trim: true

},

completed: {

type: Boolean,

default: false

},

createdAt: {

type: Date,

default: Date.now

}

});

module.exports = mongoose.model('Task', taskSchema);

**Server.js**

const express = require('express');

const mongoose = require('mongoose');

const cors = require('cors');

const app = express();

const PORT = process.env.PORT || 3000;

app.use(cors());

app.use(express.json());

const taskRoutes = require('./routes/tasks');

app.use('/api/tasks', taskRoutes);

mongoose.connect('mongodb+srv://<user>:<pass>@cluster.mongodb.net/?retryWrites=true&w=majority')

.then(() => {

console.log('Connected to MongoDB Atlas');

app.listen(PORT, () => {

console.log(`Server running on port ${PORT}`);

});

})

.catch(err => {

console.error('Failed to connect to MongoDB', err);

});

**Servicio de Tareas (task.service.ts)**

import { Injectable } from '@angular/core';

import { HttpClient } from '@angular/common/http';

import { Observable } from 'rxjs';

export interface Task {

\_id?: string;

title: string;

description?: string;

completed: boolean;

createdAt?: Date;

}

@Injectable({

providedIn: 'root'

})

export class TaskService {

private apiUrl = 'http://localhost:3000/api/tasks';

constructor(private http: HttpClient) { }

getTasks(): Observable<Task[]> {

return this.http.get<Task[]>(this.apiUrl);

}

getTask(id: string): Observable<Task> {

return this.http.get<Task>(`${this.apiUrl}/${id}`);

}

addTask(task: Task): Observable<Task> {

return this.http.post<Task>(this.apiUrl, task);

}

updateTask(id: string, task: Task): Observable<Task> {

return this.http.put<Task>(`${this.apiUrl}/${id}`, task);

}

deleteTask(id: string): Observable<any> {

return this.http.delete(`${this.apiUrl}/${id}`);

}

}

**Configuración de Rutas (app.routes.ts)**

import { Routes } from '@angular/router';

import { TaskListComponent } from './components/task-list/task-list.component';

import { TaskFormComponent } from './components/task-form/task-form.component';

export const routes: Routes = [

{ path: '', component: TaskListComponent },

{ path: 'add', component: TaskFormComponent },

{ path: 'edit/:id', component: TaskFormComponent },

{ path: '\*\*', redirectTo: '', pathMatch: 'full' }

];

**Funcionalidades Principales**

1. **Listado de Tareas**
   * Visualización de todas las tareas almacenadas
   * Indicación visual del estado
   * Editar/eliminar tareas
2. **Creación de Tareas**
   * Formulario validado
   * Título obligatorio
   * Marcar como completada
3. **Edición**
   * Modificación del estado, título, descripción
4. **Eliminación**
   * Confirmación previa
5. **Marcar como Completada**
   * Checkbox con cambio visual inmediato

**Tecnologías Utilizadas**

* MongoDB Atlas
* Express.js
* Angular
* Node.js
* Mongoose
* Bootstrap

**Cómo Ejecutar la Aplicación**

**Backend:**

$ cd backend

$ npm install

$ node server.js

**Frontend:**

$ cd gestor-tareas

$ npm install

$ ng serve

Abrir <http://localhost:4200> en el navegador.