```
from flask import Flask, request, jsonify
from flask_cors import CORS
import mysql.connector
import time, datetime

app = Flask(__name__)
CORS(app, resources={r"/*": {"origins": "*"}})

class Mensaje:
    #-----------------------------------------------------------------
    # Constructor de la clase
    def __init__(self, host, user, password, database):
        # Primero, establecemos una conexión sin especificar la base de datos
        self.conn = mysql.connector.connect(
            host=host,
            user=user,
            password=password
        )
        self.cursor = self.conn.cursor()

        # Intentamos seleccionar la base de datos
        try:
            self.cursor.execute(f"USE {database}")
        except mysql.connector.Error as err:
            # Si la base de datos no existe, la creamos
            if err.errno == mysql.connector.errorcode.ER_BAD_DB_ERROR:
                self.cursor.execute(f"CREATE DATABASE {database}")
                self.conn.database = database
            else:
                raise err

        self.cursor.execute('''CREATE TABLE IF NOT EXISTS connectbox (
            id int(11) NOT NULL AUTO_INCREMENT,
            nombre varchar(30) NOT NULL,
            celular varchar(15) NOT NULL,
            email varchar(60) NOT NULL,
            mensaje varchar(500) NOT NULL,
            fecha_envio datetime NOT NULL,
            leido tinyint(1) NOT NULL,
            gestion varchar(500) DEFAULT NULL,
            fecha_gestion datetime DEFAULT NULL,
            PRIMARY KEY(`id`)
            ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci;
            ''')
        self.conn.commit()

        # Cerrar el cursor inicial y abrir uno nuevo con el parámetro dictionary=True
        self.cursor.close()
        self.cursor = self.conn.cursor(dictionary=True)

    def enviar_mensaje(self, nombre, celular, email, mensaje):
        sql = "INSERT INTO connectbox(nombre, celular, email, mensaje, fecha_envio) VALUES (%s,
%s, %s, %s, %s)"
        fecha_envio = datetime.datetime.now()
        valores = (nombre, celular, email, mensaje, fecha_envio)
        with self.conn.cursor() as cursor:
            cursor.execute(sql, valores)
        self.conn.commit()
        return True

    def listar_mensajes(self):
        self.cursor.execute("SELECT * FROM connectbox")
        mensajes = self.cursor.fetchall()
        return mensajes

    def responder_mensaje(self, id, gestion):
        fecha_gestion = datetime.datetime.now()
        sql = "UPDATE connectbox SET leido = 1, gestion = %s, fecha_gestion = %s WHERE id = %s"
        valores = (gestion, fecha_gestion, id)
```

```
        with self.conn.cursor() as cursor:
            cursor.execute(sql, valores)
        self.conn.commit()
        return cursor.rowcount > 0

    def eliminar_mensaje(self, id):
        with self.conn.cursor() as cursor:
            cursor.execute(f"DELETE FROM connectbox WHERE id = {id}")
        self.conn.commit()
        return cursor.rowcount > 0

    def mostrar_mensaje(self, id):
        sql = f"SELECT id, nombre, celular, email, mensaje, fecha_envio, leido, gestion,
fecha_gestion FROM connectbox WHERE id = {id}"
        with self.conn.cursor() as cursor:
            cursor.execute(sql)
            return cursor.fetchone()

# Create the object with the provided database configuration
mensaje = Mensaje('ghostbin4.mysql.pythonanywhere-services.com', 'ghostbin4', 'admin2014',
'ghostbin4$default')

@app.route("/mensajes", methods=["GET"])
def listar_mensajes():
    respuesta = mensaje.listar_mensajes()
    return jsonify(respuesta)


@app.route("/mensajes", methods=["POST"])
def agregar_mensaje():
    try:
        nombre = request.form['nombre']
        celular = request.form['celular']
        email = request.form['email']
        mensaje_texto = request.form['mensaje']

        if mensaje.enviar_mensaje(nombre, celular, email, mensaje_texto):
            return jsonify({"mensaje": "Mensaje agregado"}), 201
        else:
            return jsonify({"mensaje": "No fue posible registrar el mensaje"}), 400
    except Exception as e:
        return jsonify({"error": str(e)}), 500

@app.route("/mensajes/<int:id>", methods=["PUT"])
def responder_mensaje_route(id):
    try:
        gestion = request.form.get("gestion")
        if mensaje.responder_mensaje(id, gestion):
            return jsonify({"mensaje": "Mensaje modificado"}), 200
        else:
            return jsonify({"mensaje": "Mensaje no encontrado"}), 403
    except Exception as e:
        return jsonify({"error": str(e)}), 500

if __name__ == "__main__":
    app.run(debug=True, port=5505)
```