1. Create a Flask application with an `/api` route. When this route is accessed, it should return a JSON list. The data should be stored in a backend file, read from it, and sent as a response.

2. Create a form on the frontend that, when submitted, inserts data into MongoDB Atlas. Upon successful submission, the user should be redirected to another page displaying the message **"Data submitted successfully"**. If there's an error during submission, display the error on the same page without redirection.

## app.py

```
from flask import Flask, jsonify, request, render_template

from pymongo import MongoClient

import json


app = Flask(__name__)


# MongoDB Atlas connection

client = MongoClient("your_mongodb_connection_string")

db = client['your_database_name']

collection = db['your_collection_name']
```

```
@app.route('/api', methods=['GET'])
def get_data():
    with open('backend/data.json') as f:
        data = json.load(f)
    return jsonify(data)


@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        data = request.form['data']
        try:
            collection.insert_one({'data': data})
            return render_template('index.html', success=True)
        except Exception as e:
            return render_template('index.html', error=str(e))
    return render_template('index.html')


if __name__ == '__main__':
    app.run(debug=True)
```

## templates/index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Flask MongoDB App</title>
</head>
<body>
    <h1>Submit Data to MongoDB</h1>
    <form id="dataForm">
        <label for="dataInput">Data:</label>
        <input type="text" id="dataInput" name="dataInput" required>
        <button type="submit">Submit</button>
    </form>
    <div id="message"></div>

    <script>
        document.getElementById('dataForm').addEventListener('submit', function(event) {
            event.preventDefault();
            const dataInput = document.getElementById('dataInput').value;
```

```
        fetch('/api/submit', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ data: dataInput })
        })
        .then(response => response.json())
        .then(data => {
            const messageDiv = document.getElementById('message');
            if (data.success) {
                messageDiv.innerHTML = '<p style="color: green;">Data submitted
successfully!</p>';
            } else {
                messageDiv.innerHTML = '<p style="color: red;">Error: ' + data.error + '</p>';
            }
        })
        .catch(error => {
            const messageDiv = document.getElementById('message');
            messageDiv.innerHTML = '<p style="color: red;">Error: ' + error.message + '</p>';
```

```
            });
        });
    </script>
</body>
</html>
```

## requirements.txt

```
blinker==1.9.0
click==8.2.1
colorama==0.4.6
dnspython==2.7.0
Flask==3.1.2
itsdangerous==2.2.0
Jinja2==3.1.6
MarkupSafe==3.0.2
pymongo==4.14.1
Werkzeug==3.1.3
```

## Commands To Be Execute:

To check pip is installed

- pip --version

To create a virtual environment

- virtualenv.exe env

Or

- py -m venv env

To activate virtual environment

- .\env\Scripts\activate.ps1

To install dependencies

- py -m pip install -r requirements.txt

To run the app

- py .\app.py

**Submission Guidelines -:** Attach Screenshots or command along with explanation and submit in doc (google doc or microsoft doc) format also attach GitHub repo link.

**GitHub Link:** https://github.com/tejaskaher999/tutedude.git