Task1: 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.

* Create a json file named: data.json and add json data.
* Python code:

from flask  import Flask

app = Flask(\_\_name\_\_)

@app.route('/api')

def api():

    files = open('data.json','r')

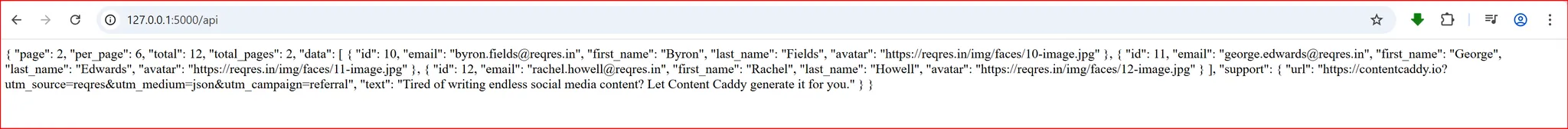
    data = files.read()

    return (data)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

* Output:



Task2: 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.

GITHUB CODE PATH: <https://github.com/abhinathts/assignment/tree/main/Python>

* Frontend app.py:
* import requests
* from flask import Flask, request,render\_template
* from datetime import datetime
* BACKEND\_URL = "http://127.0.0.1:8081"
* app = Flask(\_\_name\_\_)
* @app.route('/')
* def hello():
* try:
* today = datetime.today().strftime('%A')
* current\_time = datetime.now().strftime('%H:%M:%S')
* return render\_template('index.html', day\_is=today, current\_time=current\_time)
* except:
* print("Something went wrong")
* @app.route('/signup',methods=['POST'])
* def signup():
* try:
* form\_data = dict(request.form)
* requests.post(BACKEND\_URL + '/signup', json=form\_data)
* return "Data added successfully!"
* except:
* print("Something went wrong")
* @app.route('/get\_data')
* def get\_data():
* try:
* response = requests.get(BACKEND\_URL+'/view')
* return response.json()
* except:
* print("Something went wrong")
* if \_\_name\_\_ == '\_\_main\_\_':
* app.run(host='127.0.0.1',port=8080,debug=True)
* Frontend index.html:
* <!DOCTYPE html>
* <html lang="en">
* <head>
* <meta charset="UTF-8">
* <title>Signup Form</title>
* </head>
* <body>
* <div class="welcome">
* <p>Welcome to the web application</p>
* <div class="day">
* <p>Today is {{ day\_is }}</p>
* <p>Time is {{ current\_time }}</p>
* <div class="form\_data">
* <form method="POST" action="/signup">
* <label for="name">Name</label>
* <input type="text" name="name" id="name" placeholder="Enter your name">
* <br>
* <label for="email">Email</label>
* <input type="email" name="email" id="email" placeholder="Enter your email">
* <br>
* <label for="password">Password</label>
* <input type="password" name="password" id="password" placeholder="Enter your password">
* <br>
* <input type="submit" value="Sign Up">
* </form>
* <a href="get\_data"><button>View Data</button></a>
* </div>
* </div>
* </div>
* <style>
* body {
* background-color: cadetblue;
* }
* .welcome {
* text-align: center;
* font-size: 30px;
* font-family: "Arial Narrow";
* color: blue;
* }
* .day {
* text-align: center;
* font-size: 30px;
* font-family: "Arial Narrow";
* color: yellow;
* }
* .form\_data {
* text-align: center;
* font-size: 20px;
* font-family: "Arial Narrow";
* color: black;
* }
* </style>
* </body>
* </html>
* Backend app.py:
* import os
* import pymongo
* from flask import Flask, request, jsonify
* from dotenv import load\_dotenv
* load\_dotenv()
* MONGO\_URI = os.getenv('MONGO\_URI')
* client = pymongo.MongoClient(MONGO\_URI)
* db = client.test
* collection = db['flask-learn']
* app = Flask(\_\_name\_\_)
* @app.route('/signup',methods=['POST'])
* def signup():
* try:
* form\_data = dict(request.json)
* collection.insert\_one(form\_data) # it should be python dictionary
* return "Data added successfully!"
* except:
* print("Something went wrong!!")
* @app.route('/view')
* def view():
* try:
* data = collection.find()
* #to get data from the db
* data = list(data)
* # we need to make the data to a list format
* for item in data:
* print(item)
* del item['\_id']
* #to hide a particular value from db
* return jsonify(data)
* except:
* print("Something went wrong")
* if \_\_name\_\_ == '\_\_main\_\_':
* app.run(host='127.0.0.1',port=8081,debug=True)
* Output:

