Python code

Создаем с помощью чего угодно API калькулятор:

```
from flask import Flask, request, jsonify

calc = Flask(__name__)

@calc.route('/add', methods=['POST'])
def add():
    data = request.get_json()
    if 'a' in data and 'b' in data:
        a = data['a']
        b = data['b']
        result = a + b
        return jsonify({'result': result})
    else:
        return jsonify({'error': 'Missing parameters'}), 400

@calc.route('/subtract', methods=['POST'])
def subtract():
    data = request.get_json()
    if 'a' in data and 'b' in data:
        a = data['a']
        b = data['b']
        result = a - b
        return jsonify({'result': result})
    else:
        return jsonify({'error': 'Missing parameters'}), 400

@calc.route('/multiply', methods=['POST'])
def multiply():
    data = request.get_json()
    if 'a' in data and 'b' in data:
        a = data['a']
        b = data['b']
        result = a * b
        return jsonify({'result': result})
    else:
        return jsonify({'error': 'Missing parameters'}), 400

@calc.route('/divide', methods=['POST'])
def divide():
```

```
    data = request.get_json()
    if 'a' in data and 'b' in data:
        a = data['a']
        b = data['b']
        if b != 0:
            result = a / b
            return jsonify({'result': result})
        else:
            return jsonify({'error': 'Division by zero'}), 400
    else:
        return jsonify({'error': 'Missing parameters'}), 400

if __name__ == '__main__':
    calc.run(host="0.0.0.0",debug=True)
```

Dockerfile

Создаем файл для сборки Docker контейнера

```
# import python library
FROM python:3.8-slim

# install FLask library
RUN pip install Flask

# Create work directory
WORKDIR /app

# Copy python's file to work directory
COPY api_calc.py .

# Use 5000 port
EXPOSE 5000

# Start program
CMD ["python", "api_calc.py"]
```

Build container

Все делаем в консоли и под рутом.

1. Устанавливаем docker.
2. Проверяем, запущен ли Docker.
   `# systemctl status docker`

Должно быть `active (running)`.

3. Переходим в директорию, в которой находятся `api_calc.py` и `Dockerfile`.
4. Создаем образ Docker.

   `# docker build -f Dockerfile -t api_calc .`

   Пробегут несолько строчек. Если будут красные предупреждения - не печалимся, все хорошо, просто версия pip немножка старенькая.

```
WARNING: Running pip as the 'root' user can result in broken permis
nded to use a virtual environment instead: https://pip.pypa.io/warn

[notice] A new release of pip is available: 23.0.1 → 23.2.1
[notice] To update, run: pip install --upgrade pip
```

5. Проверяем созданный образ.

   `# docker images`

```
┌──(root㉿iankie)-[/home/iankie/Desktop/api_calc]
└─# docker images
REPOSITORY   TAG         IMAGE ID       CREATED          SIZE
api_calc     latest      a34876171b24   52 seconds ago   139MB
python       3.8-slim    2edb8612d4fb   5 weeks ago      128MB
```

6. Запускаем docker контейнер

```
┌──(root㉿iankie)-[/home/iankie/Desktop/api_calc]
└─# docker run -it -p 5000:5000 api_calc:latest
 * Serving Flask app 'api_calc'
 * Debug mode: on
WARNING: This is a development server. Do not use it
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.2:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 398-725-309
```

Контейнер с программой запущен, теперь пробуем обратиться к программе.

```
┌──(iankie㉿iankie)-[~/Downloads]
└─$ curl -X POST -H "Content-Type: application/json" -d '{"a": 2, "b": 3}'
http://172.17.0.2:5000/add
{
  "result": 5
}
```

Вывод сигнализирует об успешной контейниризации программы.