Peer Assignment
Sheikh Muhammad Sabih (2303.KHI.DEG.010)
M Humza Moeen (2303.KHI.DEG.019)

Assignment (4.4)

Browse to:

tasks/4_microservices_development/day_4_best_practices/
app_that_doesnt_follow_best_practices/

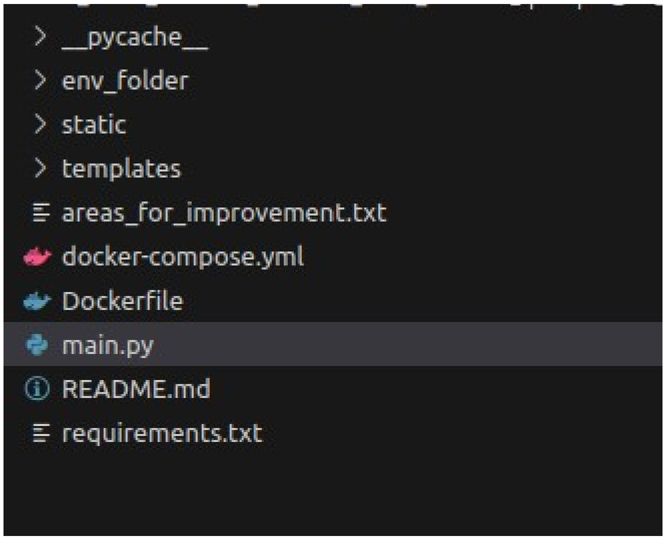
Analyze the application - which Microservice best practices does it not follow?

Think about what needs to be improved first. Have a look at the areas_for_improvement.txt file for hints.

Improve the application.

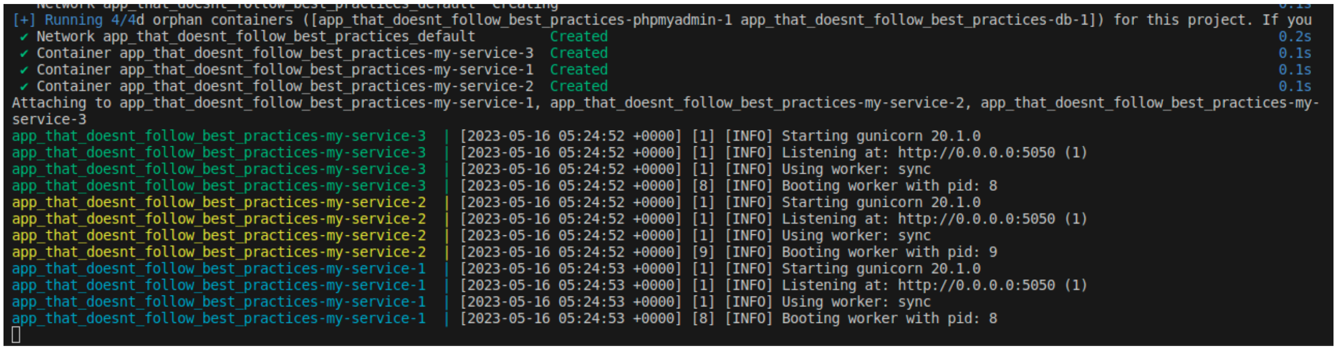
Solution

1. **__init__.py** is an extra file. So we remove it.



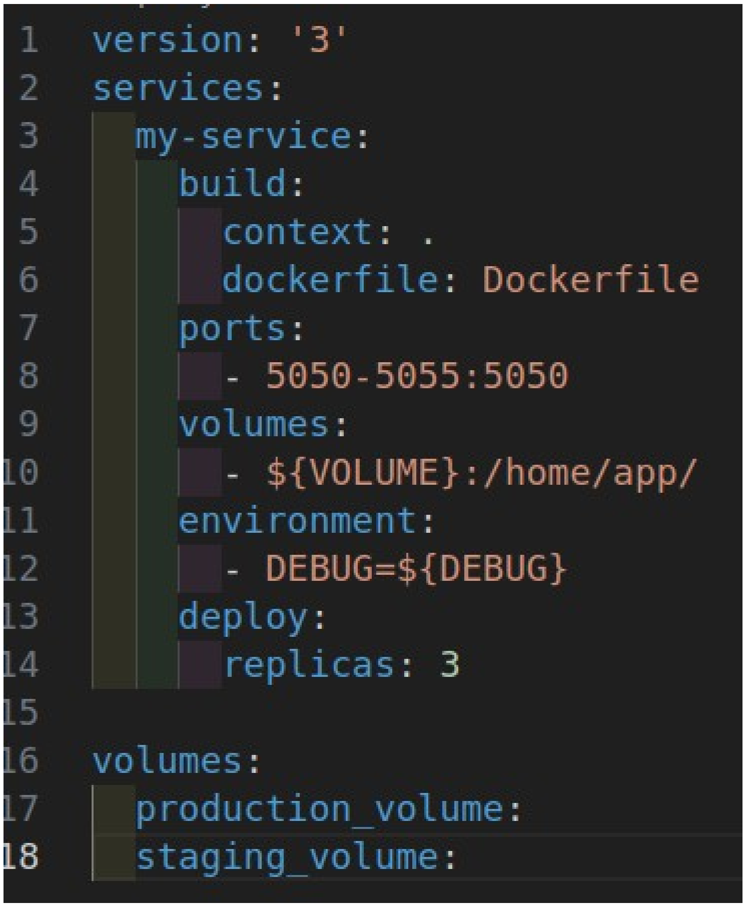
```
> __pycache__
> env_folder
> static
> templates
areas_for_improvement.txt
docker-compose.yml
Dockerfile
main.py
README.md
requirements.txt
```

2. Now we can see the **logs as output**.



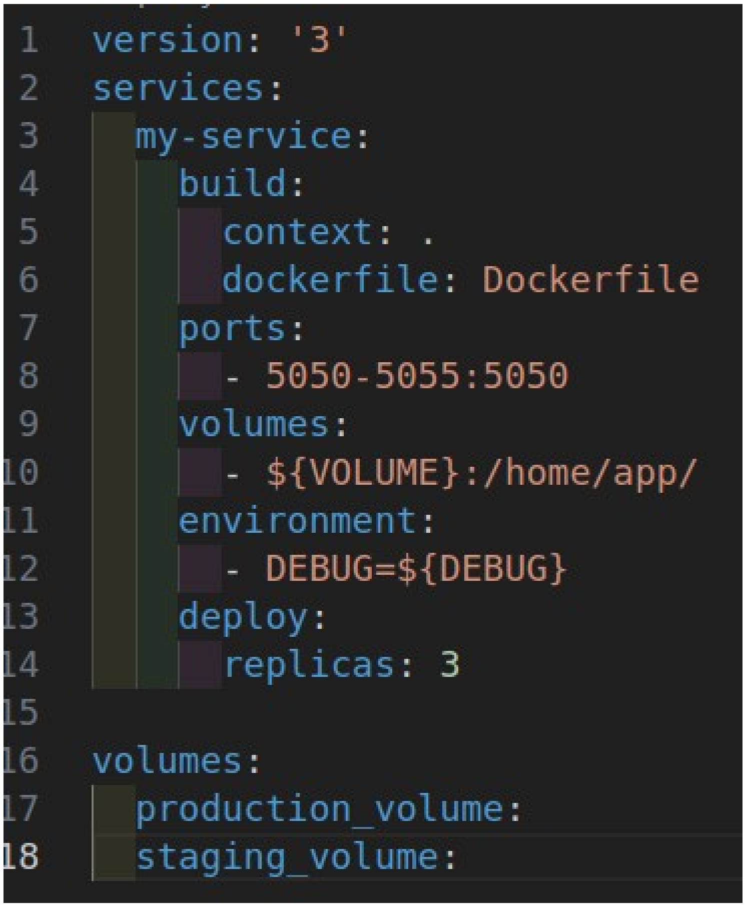
```
  Network app_that_doesnt_follow_best_practices_default  Creating                                                    0.1s
[+] Running 4/4d orphan containers ([app_that_doesnt_follow_best_practices-phpmyadmin-1 app_that_doesnt_follow_best_practices-db-1]) for this project. If you
 ✔ Network app_that_doesnt_follow_best_practices_default      Created                                                 0.2s
 ✔ Container app_that_doesnt_follow_best_practices-my-service-3  Created                                              0.1s
 ✔ Container app_that_doesnt_follow_best_practices-my-service-1  Created                                              0.1s
 ✔ Container app_that_doesnt_follow_best_practices-my-service-2  Created                                              0.1s
Attaching to app_that_doesnt_follow_best_practices-my-service-1, app_that_doesnt_follow_best_practices-my-service-2, app_that_doesnt_follow_best_practices-my-
service-3
app_that_doesnt_follow_best_practices-my-service-3  | [2023-05-16 05:24:52 +0000] [1] [INFO] Starting gunicorn 20.1.0
app_that_doesnt_follow_best_practices-my-service-3  | [2023-05-16 05:24:52 +0000] [1] [INFO] Listening at: http://0.0.0.0:5050 (1)
app_that_doesnt_follow_best_practices-my-service-3  | [2023-05-16 05:24:52 +0000] [1] [INFO] Using worker: sync
app_that_doesnt_follow_best_practices-my-service-3  | [2023-05-16 05:24:52 +0000] [8] [INFO] Booting worker with pid: 8
app_that_doesnt_follow_best_practices-my-service-2  | [2023-05-16 05:24:52 +0000] [1] [INFO] Starting gunicorn 20.1.0
app_that_doesnt_follow_best_practices-my-service-2  | [2023-05-16 05:24:52 +0000] [1] [INFO] Listening at: http://0.0.0.0:5050 (1)
app_that_doesnt_follow_best_practices-my-service-2  | [2023-05-16 05:24:52 +0000] [1] [INFO] Using worker: sync
app_that_doesnt_follow_best_practices-my-service-2  | [2023-05-16 05:24:52 +0000] [9] [INFO] Booting worker with pid: 9
app_that_doesnt_follow_best_practices-my-service-1  | [2023-05-16 05:24:53 +0000] [1] [INFO] Starting gunicorn 20.1.0
app_that_doesnt_follow_best_practices-my-service-1  | [2023-05-16 05:24:53 +0000] [1] [INFO] Listening at: http://0.0.0.0:5050 (1)
app_that_doesnt_follow_best_practices-my-service-1  | [2023-05-16 05:24:53 +0000] [1] [INFO] Using worker: sync
app_that_doesnt_follow_best_practices-my-service-1  | [2023-05-16 05:24:53 +0000] [8] [INFO] Booting worker with pid: 8
```

3. We also make 3 instance to **scale our container**.



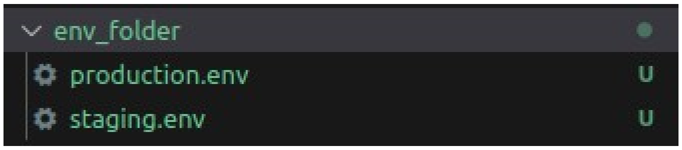
```yaml
version: '3'
services:
  my-service:
    build:
      context: .
      dockerfile: Dockerfile
    ports:
      - 5050-5055:5050
    volumes:
      - ${VOLUME}:/home/app/
    environment:
      - DEBUG=${DEBUG}
    deploy:
      replicas: 3

volumes:
  production_volume:
  staging_volume:
```

4. We also make 2 different volume for **staging** and **production**.

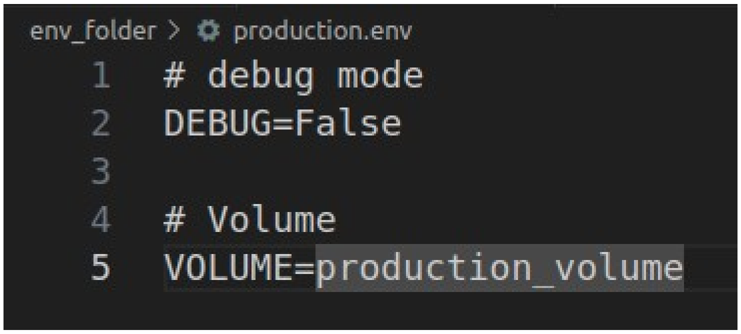


```yaml
version: '3'
services:
  my-service:
    build:
      context: .
      dockerfile: Dockerfile
    ports:
      - 5050-5055:5050
    volumes:
      - ${VOLUME}:/home/app/
    environment:
      - DEBUG=${DEBUG}
    deploy:
      replicas: 3

volumes:
  production_volume:
  staging_volume:
```

5. We have assign the 2 different **env-files** to separate our **environment**.

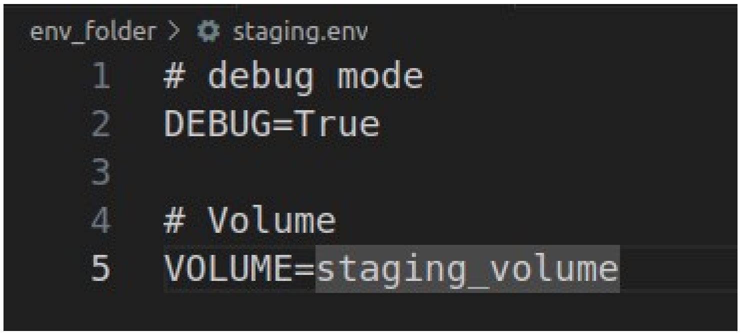


```
env_folder
  production.env    U
  staging.env       U
```



env_folder > production.env

```
# debug mode
DEBUG=False

# Volume
VOLUME=production_volume
```

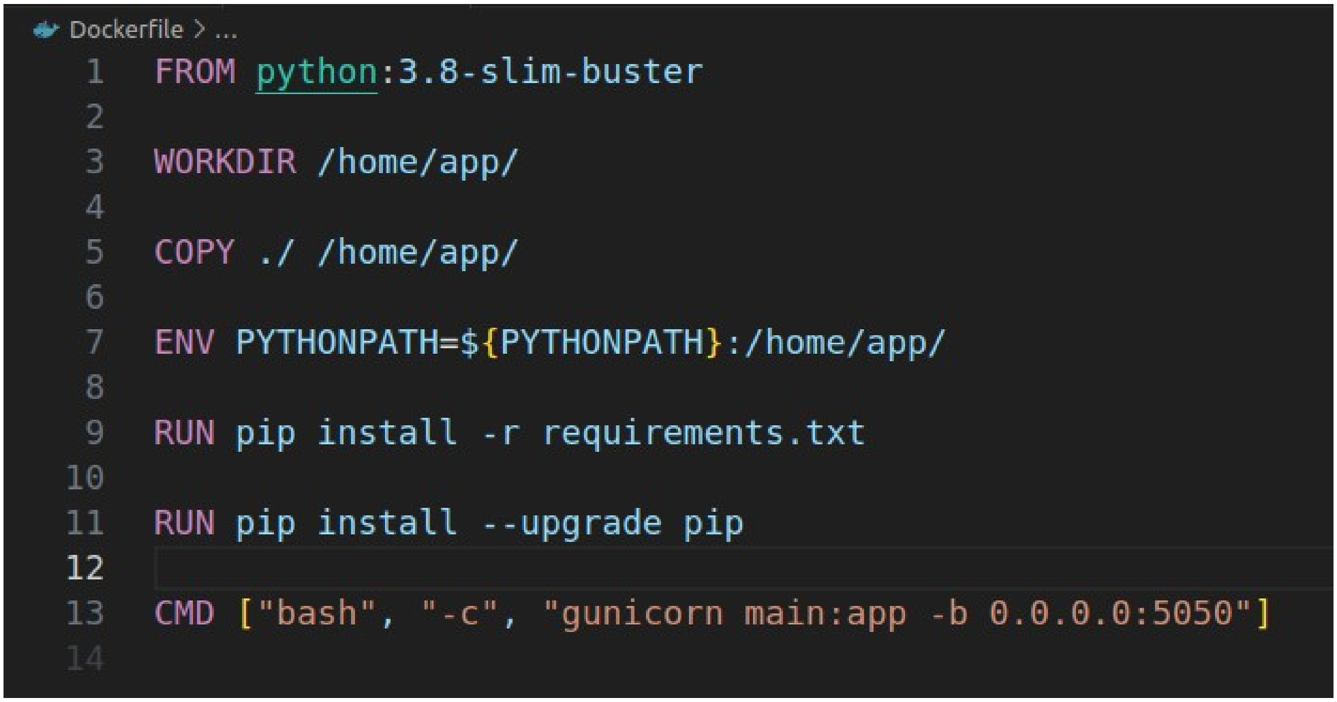


```
# debug mode
DEBUG=True

# Volume
VOLUME=staging_volume
```

6. For the above environment files we use docker compose up command and give parameters.

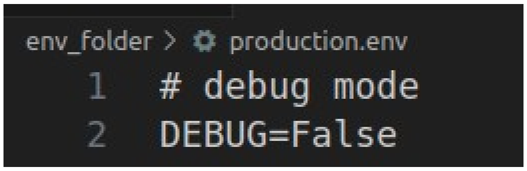
   - docker compose **--env-file env_folder/production.env** up
   - docker compose **--env-file env_folder/staging.env** up

7. We also change requirements installation from run time to build time.

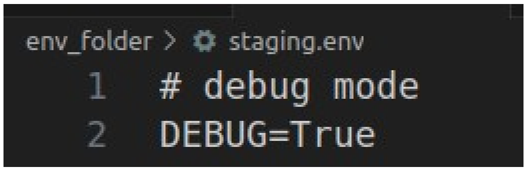


```
FROM python:3.8-slim-buster

WORKDIR /home/app/

COPY ./ /home/app/

ENV PYTHONPATH=${PYTHONPATH}:/home/app/

RUN pip install -r requirements.txt

RUN pip install --upgrade pip

CMD ["bash", "-c", "gunicorn main:app -b 0.0.0.0:5050"]
```

9. We also pass debugging form **environment file**.

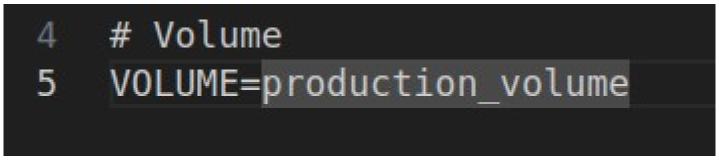


```
# debug mode
DEBUG=False
```

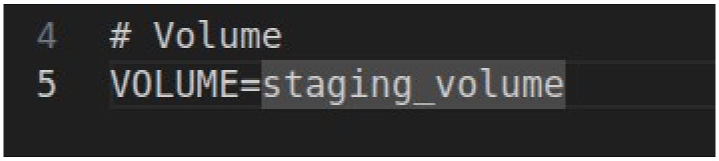


```
# debug mode
DEBUG=True
```

10. We also pass volumes from environment files so we can easily replace the database.

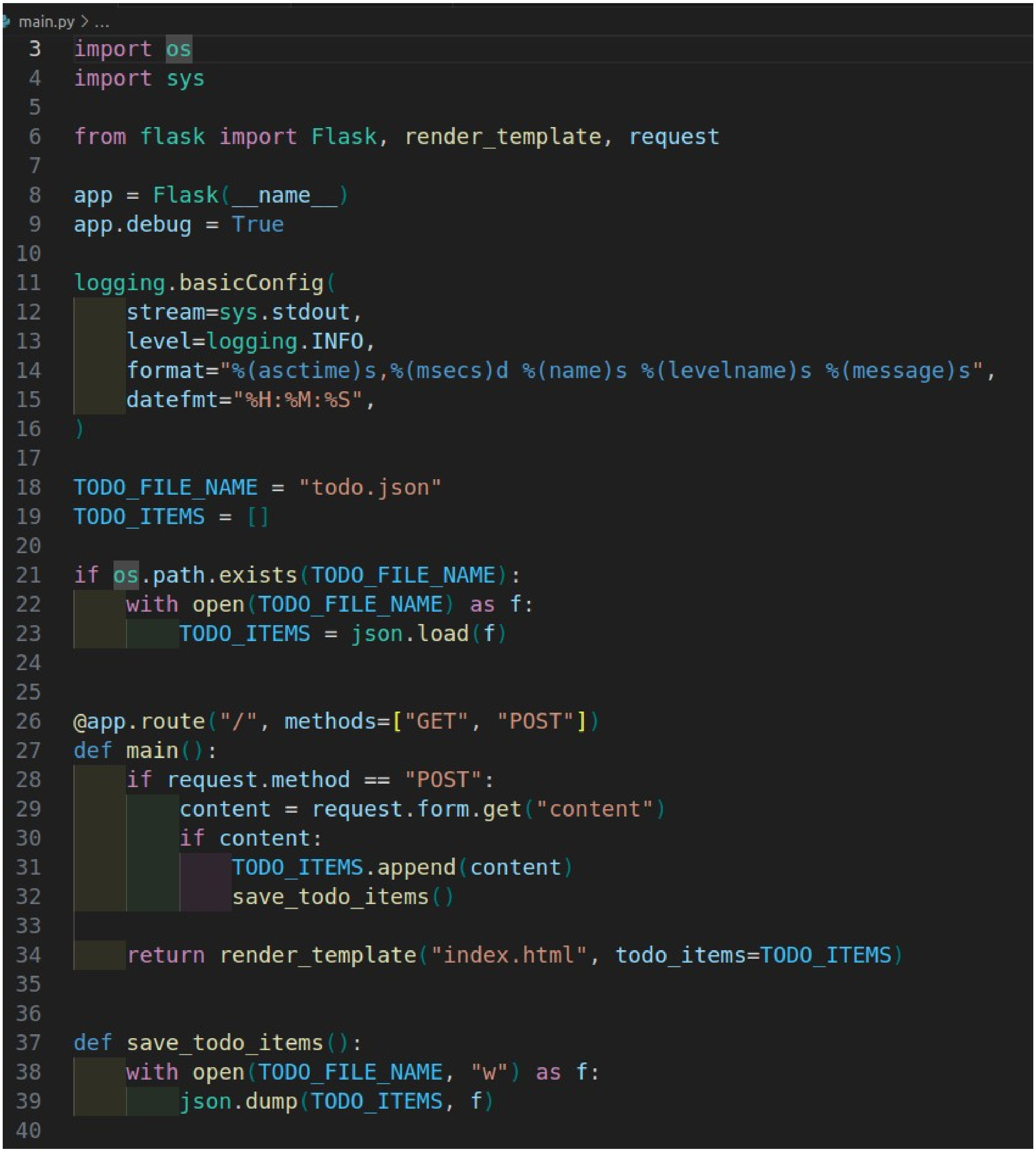


```
# Volume
VOLUME=production_volume
```



```
# Volume
VOLUME=staging_volume
```

11. We also update the main.py file and remove t**hreading, sleep time** and **show logs in output**.



```python
import os
import sys

from flask import Flask, render_template, request

app = Flask(__name__)
app.debug = True

logging.basicConfig(
    stream=sys.stdout,
    level=logging.INFO,
    format="%(asctime)s,%(msecs)d %(name)s %(levelname)s %(message)s",
    datefmt="%H:%M:%S",
)

TODO_FILE_NAME = "todo.json"
TODO_ITEMS = []

if os.path.exists(TODO_FILE_NAME):
    with open(TODO_FILE_NAME) as f:
        TODO_ITEMS = json.load(f)


@app.route("/", methods=["GET", "POST"])
def main():
    if request.method == "POST":
        content = request.form.get("content")
        if content:
            TODO_ITEMS.append(content)
            save_todo_items()

    return render_template("index.html", todo_items=TODO_ITEMS)


def save_todo_items():
    with open(TODO_FILE_NAME, "w") as f:
        json.dump(TODO_ITEMS, f)

```