Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
УРАЛЬСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
ИМЕНИ ПЕРВОГО ПРЕЗИДЕНТА РОССИИ Б.Н. ЕЛЬЦИНА
(УрФУ имени первого Президента России Б.Н. Ельцина)
Институт радиоэлектроники и информационных технологий — РТФ
Школа профессионального и академического образования

ФОРМАТИРОВАНИЕ И ЛИНТИНГ ПРОЕКТА. СБОРКА ОБРАЗА ПРОЕКТА

Отчет по лабораторной работе №5

по дисциплине «Разработка приложений»

	Дата	Подпись	
Преподаватель:	_____	_____	Кузьмин Д.
Студенты:	_____	_____	Пантелеев Е.А.

Группа: РИМ-150950

Екатеринбург

2025

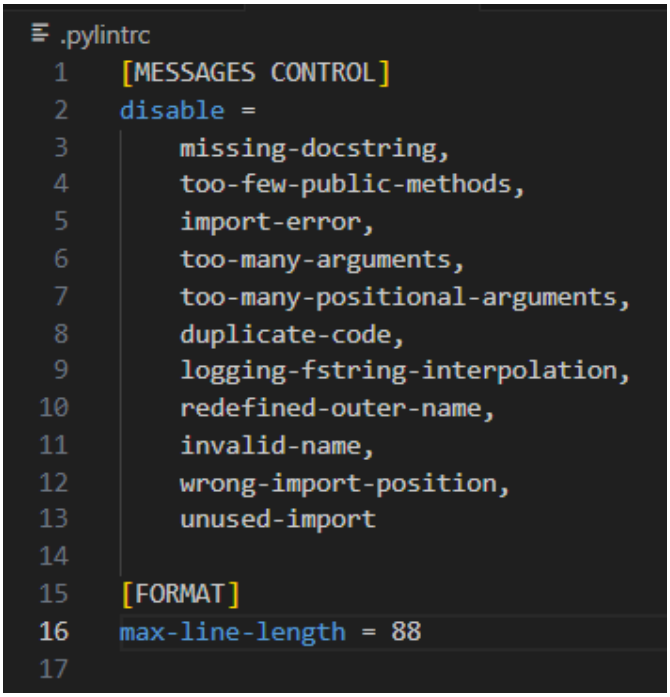
ЦЕЛЬ РАБОТЫ

Познакомится со способами поддержки качества кода и сборки образа приложения.

ХОД РАБОТЫ

В корне проекта настраиваем прекоммит .pre-commit-config.yaml

Файл .pylintrc содержит данную конфигурацию

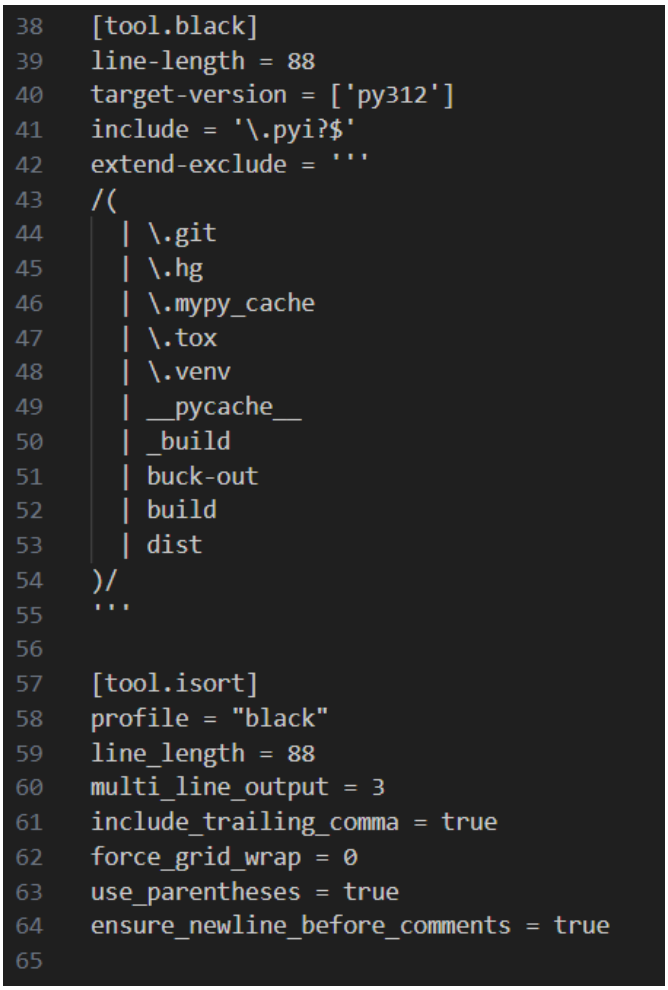


```
.pylintrc
[MESSAGES CONTROL]
disable =
    missing-docstring,
    too-few-public-methods,
    import-error,
    too-many-arguments,
    too-many-positional-arguments,
    duplicate-code,
    logging-fstring-interpolation,
    redefined-outer-name,
    invalid-name,
    wrong-import-position,
    unused-import

[FORMAT]
max-line-length = 88

```

Black и isort конфигурируются следующим образом:



```
[tool.black]
line-length = 88
target-version = ['py312']
include = '\.pyi?$'
extend-exclude = '''
/(
  | \.git
  | \.hg
  | \.mypy_cache
  | \.tox
  | \.venv
  | __pycache__
  | _build
  | buck-out
  | build
  | dist
)/
'''

[tool.isort]
profile = "black"
line_length = 88
multi_line_output = 3
include_trailing_comma = true
force_grid_wrap = 0
use_parentheses = true
ensure_newline_before_comments = true

```

Проверяем работу линтеров:

```
(.venv) C:\Users\highs\OneDrive\Документы\GitHub\lab5>git add -u

(.venv) C:\Users\highs\OneDrive\Документы\GitHub\lab5>pre-commit run --all-files
black....................................................................Passed
isort....................................................................Passed
pylint...................................................................Passed
```

Сделаем сборку образа docker в Dockerfile

```
Dockerfile > ...
    GigaCode: explain | explain step by step
1   FROM python:3.12-slim
2
    GigaCode: explain | explain step by step
3   WORKDIR /app
4
5   # Устанавливаем UV
    GigaCode: explain | explain step by step
6   RUN pip install uv
7
8   # Копируем зависимости
    GigaCode: explain | explain step by step
9   COPY pyproject.toml uv.lock ./
10
11  # Устанавливаем зависимости
    GigaCode: explain | explain step by step
12  RUN uv sync --frozen
13
14  # Копируем код приложения
    GigaCode: explain | explain step by step
15  COPY . .
16
    GigaCode: explain | explain step by step
17  ENV PYTHONPATH="/app"
18
19  # Запускаем приложение
    GigaCode: explain | explain step by step
20  CMD ["sh", "-c", \
21      "echo '=== Запуск лабораторной работы ===' && \
22       uv run python scripts/load_data.py && \
23       uv run python scripts/update_data.py && \
24       uv run python -m app.main"]
```

Создаем образ

```
(.venv) C:\Users\highs\OneDrive\Документы\GitHub\lab5>docker build -t lab5-app .
[+] Building 144.7s (12/12) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 640B
 => [internal] load metadata for docker.io/library/python:3.12-slim
 => [auth] library/python:pull token for registry-1.docker.io
 => [internal] load .dockerignore
 => => transferring context: 152B
 => [1/6] FROM docker.io/library/python:3.12-slim@sha256:590cad70271b6c1795c6a11fb5c110efca593adbd0d4883cd19c36df6a56467b
 => => resolve docker.io/library/python:3.12-slim@sha256:590cad70271b6c1795c6a11fb5c110efca593adbd0d4883cd19c36df6a56467b
 => => sha256:1f384a3df5003cc3a739008d6e3c2b2afc752887e9ce09757747c0bbb6e68983 250B / 250B
 => => sha256:dfff024aded812f05863f68d31b04030038e01017329961ea2d5f37e6a1c70ef 1.29MB / 1.29MB
 => => sha256:89933f78055059f29cf8b5a9c0b6df0fe9d96c388b99215881bf653ed6f1ca35 12.11MB / 12.11MB
 => => sha256:1733a4cd59540b3470ff7a90963bcdea5b543279dd6bdaf022d7883fdad221e5 29.78MB / 29.78MB
 => => extracting sha256:1733a4cd59540b3470ff7a90963bcdea5b543279dd6bdaf022d7883fdad221e5
 => => extracting sha256:dfff024aded812f05863f68d31b04030038e01017329961ea2d5f37e6a1c70ef
 => => extracting sha256:89933f78055059f29cf8b5a9c0b6df0fe9d96c388b99215881bf653ed6f1ca35
 => => extracting sha256:1f384a3df5003cc3a739008d6e3c2b2afc752887e9ce09757747c0bbb6e68983
 => [internal] load build context
 => => transferring context: 467.29kB
 => [2/6] WORKDIR /app
 => [3/6] RUN pip install uv
 => [4/6] COPY pyproject.toml uv.lock ./
 => [5/6] RUN uv sync --frozen
 => [6/6] COPY . .
 => exporting to image
 => => exporting layers
 => => exporting manifest sha256:b59a90e6238b2476a8815fb7bcb640c885b729b98075141e700cf99809f538c5
 => => exporting config sha256:e21761a04c601e58d23db5ba66a61bef4d8769b0c5701deb37bed44793d9fef8
 => => exporting attestation manifest sha256:c2cb7ade1691f8112f0d1eb125d0bd8a1cfa136b7da4d1f49e7cb6dd6ef45caa
 => => exporting manifest list sha256:cdbc4d5e304cbce76b1c83cd20ffabb9189e5434bb6a68ff38b851f1e7c88e8e
 => => naming to docker.io/library/lab5-app:latest
 => => unpacking to docker.io/library/lab5-app:latest
```

Запускаем контейнер

```
(.venv) C:\Users\highs\OneDrive\Документы\GitHub\lab5>docker run -d -p 8000:8000 --name lab5-container lab5-app
7787778dcd89101cb1c267e2d3b07cd4bda851a9b26d348596fdd34584792f20
```

После всего делаем коммит на гит и смотрим на работу линтеров и форматтеров.

```
(.venv) C:\Users\highs\OneDrive\Документы\GitHub\lab5>git commit -m "add: add linters and docker"
black....................................................................Passed
isort....................................................................Passed
pylint...................................................................Passed
[main (root-commit) b1c04c0] add: add linters and docker
 39 files changed, 3146 insertions(+)
 create mode 100644 .coverage
 create mode 100644 .dockerignore
```

Ответы на вопросы

1. Что такое Docker-контейнер и чем он отличается от виртуальной машины?

Docker-контейнер – изолированное окружение для приложения, использующее ядро хоста. Виртуальная машина – полная эмуляция оборудования с отдельным ядром. Контейнеры легче и быстрее.

2. Как работает кеширование слоев в Docker и почему это важно?

Каждый слой Dockerfile кешируется. Изменение одного слоя пересобирает только его и последующие, это ускоряет сборку образов.

3. Что означает инструкция `depends_on` в docker-compose?

depends_on в docker-compose гарантирует порядок запуска сервисов (например, БД перед приложением). Не ждёт готовности сервиса, только запуск.

4. Почему миграции БД выполняются в entrypoint.sh, а не во время сборки образа?

Миграции в entrypoint, а не при сборке, потому что схема БД связана с данными, а не с кодом. Сборка образа должна быть детерминированной.

5. Что произойдет, если миграции завершатся ошибкой при запуске контейнера?

При ошибке миграций контейнер упадёт. Приложение не запустится, нужно исправить миграции и перезапустить.

6. В чем разница между линтерами (flake8) и форматерами (black)?

Линтеры (flake8) проверяют код на ошибки и стиль, форматеры (black) автоматически форматируют код. Первые анализируют, вторые правят.

7. Как pre-commit хуки помогают в разработке?

Pre-commit хуки автоматически проверяют код перед коммитом. Ловут ошибки стиля, типов, безопасности до попадания в репозиторий.

ЗАКЛЮЧЕНИЕ

В ходе работы были освоены способы поддержки качества кода и сборки образа приложения.