

Отчёт по отладке приложения

Selectel-Vacancies-API

github.com/imronaxl/selectest-api/

12 февраля 2026 г.

Содержание

1	Анализ и первичный запуск	2
2	Исправление багов	3
2.1	Баг №1 – Опечатка в алиасе переменной окружения	3
2.2	Баг №2 – Неверное имя БД по умолчанию	3
2.3	Баг №3 – Падение при отсутствии города (AttributeError)	3
2.4	Баг №4 – Утечка HTTP-соединений (незакрытый клиент)	3
2.5	Баг №5 – Неверный интервал запуска (секунды вместо минут)	4
2.6	Баг №6 – Нарушение контракта при создании дубликата (POST)	4
2.7	Баг №7 – N+1 запросов в upsert_external_vacancies	4
2.8	Баг №8 – Отсутствие проверки уникальности external_id при обновлении (PUT)	5
3	Результаты тестирования	6
3.1	Протокол выполнения	6
4	Итог	8

1 Анализ и первичный запуск

Что сделано

Запущен Docker Compose:

```
1 docker compose up --build
```

Листинг 1: Первичный запуск

Проблема

Контейнер app не может подключиться к базе данных:

```
1 sqlalchemy.exc.OperationalError: (psycopg2.OperationalError)
2 FATAL:  database "postgres_typo" does not exist
```

Листинг 2: Ошибка подключения

Причина

В файле конфигурации допущены две ошибки:

- Опечатка в алиасе переменной окружения (DATABSE_URL вместо DATABASE_URL);
- Имя базы данных по умолчанию (postgres_typo) не соответствует тому, что указано в docker-compose.yml (postgres).

Решение

Исправлен файл app/core/config.py.

Было:

```
1 database_url: str = Field(
2     "postgresql+asyncpg://postgres:postgres@db:5432/postgres_typo",
3     validation_alias="DATABSE_URL",
4 )
```

Стало:

```
1 database_url: str = Field(
2     "postgresql+asyncpg://postgres:postgres@db:5432/postgres",
3     validation_alias="DATABASE_URL",
4 )
```

После исправления приложение успешно подключилось к БД, миграции Alembic применились автоматически.

2 Исправление багов

2.1 Баг №1 – Опечатка в алиасе переменной окружения

- **Файл:** app/core/config.py
- **Строка:** 13
- **Код до:** validation_alias="DATABSE_URL"
- **Код после:** validation_alias="DATABASE_URL"
- **Причина:** Пропущена буква «A» в имени переменной, из-за чего значение из .env не считывалось.

2.2 Баг №2 – Неверное имя БД по умолчанию

- **Файл:** app/core/config.py
- **Строка:** 12
- **Код до:** ...@db:5432/postgres_typo
- **Код после:** ...@db:5432/postgres
- **Причина:** Несоответствие дефолтного значения и реальной конфигурации Docker.

2.3 Баг №3 – Падение при отсутствии города (AttributeError)

- **Файл:** app/services/parser.py
- **Строка:** 39
- **Код до:** "city_name": item.city.name.strip()
- **Код после:** "city_name": item.city.name.strip() if item.city else None
- **Причина:** Поле city в JSON-ответе Selectel может быть null. Обращение к атрибуту name у None вызывает исключение и останавливает парсинг.

2.4 Баг №4 – Утечка HTTP-соединений (незакрытый клиент)

- **Файл:** app/services/parser.py
- **Строка:** 26–28
- **Код до:**

```
1 client = httpx.AsyncClient(timeout=timeout)
2 # ...
3 # client.close()
```

- **Код после:**

```
1 async with httpx.AsyncClient(timeout=timeout) as client:
2     # ...
```

- **Причина:** HTTP-клиент создавался, но никогда не закрывался, что приводило к исчерпанию файловых дескрипторов и утечке памяти при длительной работе.

2.5 Баг №5 – Неверный интервал запуска (секунды вместо минут)

- Файл: app/services/scheduler.py
- Стока: 11
- Код до: seconds=settings.parse_schedule_minutes
- Код после: minutes=settings.parse_schedule_minutes
- Причина: В настройках указаны минуты, но параметр ошибочно передан в seconds. Задача запускалась каждые 5 секунд вместо 5 минут.

2.6 Баг №6 – Нарушение контракта при создании дубликата (POST)

- Файл: app/api/v1/vacancies.py

- Стока: 43–46

- Код до:

```
1 return JSONResponse(
2     status_code=status.HTTP_200_OK,
3     content={"detail": "Vacancy with external_id already
4         exists"},
```

- Код после:

```
1 raise HTTPException(
2     status_code=status.HTTP_409_CONFLICT,
3     detail="Vacancy with external_id already exists",
4 )
```

- Причина: Возврат статуса 200 вместо 409 нарушает REST-практики и не соответствует объявленной Pydantic-схеме, что приводило к ошибкам валидации в Swagger.

2.7 Баг №7 – N+1 запросов в upsert_external_vacancies

- Файл: app/crud/vacancy.py

- Стока: 72–89 (в исходной версии)

- Код до (упрощённо):

```
1 existing_ids = set(...)      #           external_id
2 for payload in payloads:
3     if ext_id in existing_ids:
4         result = await session.execute(select(Vacancy).where
5             (...))
6         vacancy = result.scalar_one()    # N+1!
7         ...
```

- Код после:

```

1 existing_map = {}
2 if external_ids:
3     res = await session.execute(
4         select(Vacancy).where(Vacancy.external_id.in_(
5             external_ids)))
6     existing_map = {v.external_id: v for v in res.scalars().all()}
7
8 for payload in payload_list:
9     ext_id = payload["external_id"]
10    if ext_id in existing_map:
11        vacancy = existing_map[ext_id]
12        for field, value in payload.items():
13            setattr(vacancy, field, value)
14    else:
15        session.add(Vacancy(**payload))
16        created_count += 1

```

- **Причина:** Отдельный SELECT для каждой существующей вакансии. При 1000 записей — 1000 лишних запросов к БД.

2.8 Баг №8 – Отсутствие проверки уникальности external_id при обновлении (PUT)

- Файл: app/api/v1/vacancies.py

- Стока: 56–71

- Код до:

```

1 #
2 return await update_vacancy(session, vacancy, payload)

```

- Код после:

```

1 if payload.external_id is not None:
2     existing = await get_vacancy_by_external_id(session,
3         payload.external_id)
4     if existing and existing.id != vacancy_id:
5         raise HTTPException(
6             status_code=status.HTTP_409_CONFLICT,
7             detail="External ID already in use by another
8                   vacancy",
9         )
10    return await update_vacancy(session, vacancy, payload)

```

- **Причина:** Отсутствие бизнес-валидации позволяло присвоить вакансии external_id, уже закреплённый за другой записью, что вызывало IntegrityError (500).

3 Результаты тестирования

Все исправления были проверены с помощью автоматизированного тестового скрипта `test_project.py`. Скрипт выполняет:

- запуск Docker-контейнеров;
- проверку доступности API;
- 10 интеграционных тестов (включая граничные случаи);
- анализ логов планировщика;
- остановку и очистку контейнеров.

3.1 Протокол выполнения

```
1 =====
2
3      : 2026-02-12 11:41:07
4 =====
5
6      : 10
7      : 0
8
9      #1:
10     : POST http://localhost:8000/api/v1/parse/
11     : 200,
12     : {"created":0}
13
14     #2:
15     : POST http://localhost:8000/api/v1/vacancies/
16     : 201,
17     : {"title":"Python Developer", ..., "external_id"
18     : 1001}
19
20     #3:                                     (409 Conflict)
21     : POST http://localhost:8000/api/v1/vacancies/
22     : 409,
23     : {"detail":"Vacancy with external_id already exists"}
24
25     #4:
26     : GET http://localhost:8000/api/v1/vacancies/
27     : 200,
28
29     #5:                                     ID
29     : GET http://localhost:8000/api/v1/vacancies/26
```

```

30          : 200,           :
31
32      200
33
34      #6:
35          : PUT http://localhost:8000/api/v1/vacancies/26
36          : 200,           :
37
38      200
39
40      #7:
41          : POST http://localhost:8000/api/v1/vacancies/
42          : 201,           :
43
44      201
45
46      #8:
47          external_id (409)
48          : PUT http://localhost:8000/api/v1/vacancies/26
49          : 409,           :
50
51      409
52          : {"detail":"External ID already in use by another
53          : vacancy"}
54
55      #9:
56          : DELETE http://localhost:8000/api/v1/vacancies/27
57          : 204,           :
58
59      204
60
61      #10:
62
63          : LOG
64          : >=2
65          : 70 ,           :
66
67          : 3

```

Листинг 3: Результаты тестирования

4 Итог

После внесения всех исправлений:

- **Приложение стабильно запускается** в Docker без ошибок подключения к БД.
- **Парсер** корректно обрабатывает любые ответы внешнего API (включая null-поля), HTTP-клиент закрывается, утечек нет.
- **Фоновая задача** запускается строго с заданным интервалом (5 минут по умолчанию).
- **CRUD-операции** полностью работоспособны:
 - Создание валидирует уникальность `external_id` и возвращает 201 `Created` / 409 `Conflict`;
 - Обновление проверяет конфликты по `external_id` и отдаёт 409 `Conflict` при попытке занять чужой идентификатор;
 - Чтение, фильтрация, удаление работают согласно спецификации.
- **Производительность** операций массового обновления (`upsert`) оптимизирована: N+1 запросов устраниён.
- **Все 8 багов исправлены**, ни одного нового дефекта не внесено.

Код исправленного приложения доступен на GitHub:

<https://github.com/Imronaxl/selectest-api/tree/main/selectest-api>