

Отчет по тестовому заданию на стажировку в компанию **Selectel**

Выполнил:
студент 4 курса ПМИ ФизМех
Соколов А. Н.

1. Исходный код программы

Исправленная версия была выложена на моем github (github.com/azya0) в публичный репозиторий ([azya0/selecttest-api](https://github.com/azya0/selecttest-api))

В рамках исправления приложения, каждое изменение сохранялось в отдельный commit, который был описан по общепринятому соглашению (<https://www.conventionalcommits.org>)

Так же, во время работы, изменения докумментировались в файл [solution.md](#) ([selecttest-api/solution.md at main · azya0/selecttest-api](#)), который сильно облегчил мне написание данного отчёта

2. Баги

Данный раздел посвящен ошибкам, мешающим корректной работе программы

1. Неверный alias у поля класса Settings

Путь к файлу: app/core/config.py

У поля “database_url” в классе Settings параметр функции Field, а именно - validation_alias, обладал строковым значением DATABSE_URL, вместо ожидаемого DATABASE_URL, указанного в .env.example, что приводило к валидационной ошибке pydantic и не давало программе работать:

```
File "C:\code\selectest-api\app\api\v1\router.py", line 3, in <module>
    from app.api.v1.parse import router as parse_router
File "C:\code\selectest-api\app\api\v1\parse.py", line 4, in <module>
    from app.db.session import async_session_maker
File "C:\code\selectest-api\app\api\v1\session.py", line 3, in <module>
    from app.core.config import settings
File "C:\code\selectest-api\app\core\config.py", line 20, in <module>
    settings = Settings()
File "C:\code\selectest-api\venv\Lib\site-packages\pydantic_settings\main.py", line 242, in __init__
    super().__init__(**__pydantic_self__.__class__.__settings_build_values(sources, init_kwargs))
File "C:\code\selectest-api\venv\Lib\site-packages\pydantic\main.py", line 258, in __init__
    validated_self = self.__pydantic_validator__.validate_python(data, self_instance=self)
pydantic_core._pydantic_core.ValidationError: 1 validation error for Settings
database_url
  Extra inputs are not permitted [type=extra_forbidden, input_value='postgresql+asyncpg://pos...stgres@db:5432/postgres', input_type=str]
  For further information visit https://errors.pydantic.dev/2.12/v/extra_forbidden
```

2. Пропущена проверка на NoneType при вызове параметра “name” у поля “city”

Путь к файлу: app/services/parser.py

При парсинге данных со стороннего api возникла проблема. Для получения параметра “city_name”, использовалось выражение:

```
city_name=item.city.name.strip()
```

При этом тип параметра “city” - “Optional[ExternalCity]”, что говорит о том, что значение city может быть “None”. Однако соответствующей проверкой перед получением значения “name” не было, что мешало корректной работе программы.

Исправление:

```
city_name=item.city.name.strip() if item.city is not None else None
```

```
2026-02-19 16:36:44,070 | ERROR | app.main | Ошибка фонового парсинга: 'NoneType' object has no attribute 'name'
Traceback (most recent call last):
  File "C:\code\selectest-api\app\main.py", line 24, in _run_parse_job
    await parse_and_store(session)
  File "C:\code\selectest-api\app\services\parser.py", line 43, in parse_and_store
    "city_name": item.city.name.strip(),
                  ~~~~~~
AttributeError: 'NoneType' object has no attribute 'name'
```

3. Неправильный временной интервал у фоновой задачи

Путь к файлу: *app/services/scheduler.py*

При указании количества секунд интервала, передавалось количество минут.

```
seconds=settings.parse_schedule_minutes
```

Исправление:

```
seconds=settings.parse_schedule_minutes * 60
```

4. Неверный ответ у POST /api/v1/vacancies/ endpoint'a

Путь к файлу: *app/api/v1/vacancies.py*

При получении в качестве параметра “external_id” (уникального в рамках базы данных) значения, запись с

которым уже была в базе данных, отправлялся JsonResponse с 200-ым status-кодом. Данный случай обработки ошибки явно нельзя назвать успешным, поэтому он был заменён на:

```
raise HTTPException(status.HTTP_409_CONFLICT, detail="Vacancy with  
external_id already exists")
```

“В данной ситуации следует вернуть статус код 409 Conflict.” - сказала мне языковая модель DeepSeek, у которой я решил уточнить, верный ли статус-код я собираюсь возвращать при обработке данной ошибки.

5. Отсутствие проверки уникального external_id у PATCH* /api/v1/vacancies/endpoint'a

Путь к файлу: *app/api/v1/vacancies.py*

** В оригинальном коде задачи этот метод был PUT по REST'у, однако был намерено заменён на PATCH. Причина этого будет объяснена далее по отчёту.*

При установке external_id на значение, которым уже обладает запись в базе данных, приложение выдавало необработанное исключение и возвращала пользователю 500-ый status-code.

Исправление:

```
if payload.external_id is not None:  
    vacancy_by_external_id = await  
get_vacancy_by_external_id(session, payload.external_id)
```

```
if vacancy_by_external_id is not None and
vacancy_by_external_id.id != vacancy.id:
    raise HTTPException(status.HTTP_409_CONFLICT,
detail="Vacancy with external_id already exists")
```

6. Клиент httpx не был закрыт

Путь к файлу: *app/service/parser.py*

в асинхронной функции **parse_and_store** открывается асинхронный клиент “AsyncClient” библиотеки httpx. Однако, этот клиент не закрывается.

Исправление:

Т.к. клиент открывается в try-ехсепт блоке, был добавлен блок finally, внутри которого закрывается клиент:

```
finally:
    await client.aclose()
```

Так же можно бы было открывать клиент, используя `async with`, но это добавило бы дополнительную табуляцию, заставив разбить функцию `parse_and_store` на несколько подфункций.

7. Нарушение типизации

Путь к файлу: *app/crud/vacancy.py*

в асинхронной функции **upsert_external_vacancies** в случае наличия элементов в списке **external_ids** формируется запрос к базе данных, который потом конвертируется в множество:

```
existing_ids = set(existing_result.scalars()).all())
```

Что логично, потому что далее можно будет легко проверять элемент на принадлежность этому множеству за $O(1)$ по time complexity. Однако, если элементов в списке **external_ids** нет, то **existing_ids** задавался как:

```
existing_ids = {}
```

Что является ошибкой, т.к. r-value “{}” по умолчанию является словарём, а не множеством.

Исправление:

```
existing_ids = set()
```

Проблемы

Данный раздел посвящен ошибкам при разработке api, неудачных архитектурных решениям и т.п.

1. Замена типа PUT на PATCH

Путь к файлу: app/api/v1/vacancies.py

По **CRUD** третья буква “U” означает Update и говорит об обновлении уже существующих данных, а не о записи новых. Поэтому PUT как метод REST’а здесь не подходит. Также для обновления модели в PUT требуется указывать все параметры модели, что неудобно, особенно если пользователь на frontend’е не обладает всеми данными для этого. Гораздо уместнее использовать PATCH, который только обновляет данные, как частично, так и полностью.

Для исправления я изменил метод у router’а, изменил pydantic схему:

```
class VacancyUpdate(BaseModel):
    title: Optional[str] = None
    timetable_mode_name: Optional[str] = None
    tag_name: Optional[str] = None
    city_name: Optional[str] = None
    published_at: Optional[datetime] = None
    is_remote_available: Optional[bool] = None
    is_hot: Optional[bool] = None
    external_id: Optional[int] = None
```

Добавил дополнительную валидацию:

```
if not payload.model_fields_set:
    raise HTTPException(status_code=status.HTTP_400_BAD_REQUEST,
                        detail="Empty update data")
```


2. Изменение typehint'а у асинхронных функций `get_session`.

Путь к файлам:

- `app/api/v1/vacancies.py`
- `app/api/v1/parse.py`

Функция `get_session`, использующая слово “yield” является генератором. А если быть конкретнее, то асинхронным генератором. Поэтому согласно PEP 484, возвращаемый тип должен обозначаться как “**AsyncGenerator[AsyncSession, None]**”, а не просто `AsyncSession`.

P.S. Во время инъекции зависимости через “Depends” от FastAPI действительно `get_session` преобразуется к `AsyncSession`, но это как раз и делает сам “Depends”. Сама же функция `get_session` пишется **вне контекста этого механизма инъекции**, поэтому должна указывать на правильный тип возвращаемого значения.

Также написание двух разных `get_session` в разных файлах (при этом идентичных по содержанию) не является хорошей практикой, нарушая принцип DRY, усложняя рефакторинг кода (как в моём случае)

3. Миграция в app контейнере

На взгляд автора данного отчёта в зону ответственность backend-приложения не входит миграция для базы данных. Обычно этим занимаются devops'ы, отдельно от логики приложения. На крайний случай, было бы лучше создать

отдельный контейнер под миграцию, что я и реализовал (файл Dockerfile.migration).

4. Удаление данных через delete

В рамках предмета “Проектирование баз данных” преподаватель, стоявший у истоков разработки движков, сказал, что данные зачастую слишком ценны, чтобы выкидывать их на ветер. Поэтому вместо удаления записей из базы данных, он рекомендовал завести булево поле, отвечающее за актуальность данных. Но вместо этого при методе DELETE запись удаляется из базы данных, что совершенно лишено какого-либо смысла для не созданных вручную записей, потому что уже через 5 минут (а то и меньше), парсер вернёт эту запись (с отличием в поле `created_at`, разве что)

Я не был уверен в намерениях разработчика, писавшего это, поэтому оставил данную часть без исправлений.

5. База данных - одна таблица

Хранить повторяющиеся данные (как, например, названия городов) - непрактичное решение. Лучше бы было создать отдельную таблицу `city` и заносить в нее города, назначая им `id`, которые можно бы было использовать в поле `city_id`, вместо `city_name`. Таким образом, мы бы сэкономили память, установив связь `one-to-many`. В логике ORM можно было бы пользоваться `relationships` от `sqlalchemy`.

Я не был уверен, что это одна из тех ошибок, которую мне стоит исправлять, т.к. несмотря на то, что это плохое архитектурно решение, оно при этом рабочее.

3. Итог

- Docker-контейнеры собираются

```
[+] Running 4/4
✔ selectest-api-migration Built
✔ selectest-api-app Built
✔ Container selectest-api-migration-1 Recreated
✔ Container selectest-api-app-1 Recreated
```

- Приложения запускаются без ошибок

```
adminer-1 | [Sat Feb 21 00:26:53 2026] PHP 8.4.17 Development Server (http://[::]:8888) started
db-1 | PostgreSQL Database directory appears to contain a database; Skipping initialization
db-1 |
db-1 | 2026-02-21 00:26:53.923 UTC [1] LOG: starting PostgreSQL 16.12 (Debian 16.12-1.pgdg13+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 14.2.0-19) 14.2.0, 64-bit
db-1 | 2026-02-21 00:26:53.924 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432
db-1 | 2026-02-21 00:26:53.924 UTC [1] LOG: listening on IPv6 address "::", port 5432
db-1 | 2026-02-21 00:26:53.924 UTC [1] LOG: listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
db-1 | 2026-02-21 00:26:53.948 UTC [20] LOG: database system was shut down at 2026-02-21 00:26:26 UTC
db-1 | 2026-02-21 00:26:53.954 UTC [1] LOG: database system is ready to accept connections
migration-1 | INFO [alembic.runtime.migration] Context impl PostgresqlImpl.
app-1 | INFO [alembic.runtime.migration] Will assume transactional DDL.
app-1 | INFO [alembic.runtime.migration] Context impl PostgresqlImpl.
app-1 | INFO [alembic.runtime.migration] Will assume transactional DDL.
app-1 | 2026-02-21 00:27:03.665 DEBUG | httpcore.connection | connect_tcp.complete return value=httpcore.backends.anyio.AnyIOStream object at 0x7a99f78b4910>
app-1 | 2026-02-21 00:27:03.665 DEBUG | httpcore.connection | start_tls.started ssl_context=ssl.SSLContext object at 0x7a99f78b4c30> server_hostname='api.selectel.ru' timeout=10.0
app-1 | 2026-02-21 00:27:03.692 DEBUG | httpcore.connection | start_tls.complete return value=httpcore.backends.anyio.AnyIOStream object at 0x7a99f78b4c10>
app-1 | 2026-02-21 00:27:03.693 DEBUG | httpcore.http11 | send_request_headers.started request=<Request [b'GET']>
app-1 | 2026-02-21 00:27:03.693 DEBUG | httpcore.http11 | send_request_headers.complete
app-1 | 2026-02-21 00:27:03.694 DEBUG | httpcore.http11 | send_request_body.started request=<Request [b'GET']>
app-1 | 2026-02-21 00:27:03.694 DEBUG | httpcore.http11 | send_request_body.complete
app-1 | 2026-02-21 00:27:03.694 DEBUG | httpcore.http11 | receive_response_headers.started request=<Request [b'GET']>
app-1 | 2026-02-21 00:27:03.983 DEBUG | httpcore.http11 | receive_response_headers.complete return_value=(b'HTTP/1.1', 200, b'OK', [(b'date', b'Sat, 21 Feb 2026 00:27:03 GMT'), (b'content-type', b'application/json'), (b'transfer-encoding', b'chunked'), (b'server-id', b'1'), (b'access-control-allow-origin', b'*'), (b'access-control-allow-methods', b'GET, POST, PUT, DELETE, COPY, PATCH, OPTIONS, HEAD'), (b'access-control-allow-credentials', b'true'), (b'access-control-expose-headers', b'X-Token, X-Limit, X-Offset, X-Total-Count, Content-Disposition, Link, X-Available-Location, X-Request-Id, X-Banned, X-Min-Balance, X-User-Created, X-Subject-To-ken, X-File-ID'), (b'access-control-max-age', b'1228800'), (b'x-frame-options', b'SAMEORIGIN'), (b'set-cookie', b'cid=Chni2mV49dtt8aaVK3Uag==; expires=Sun, 21-Feb-27 00:27:03 GMT; path=/'), (b'p3p', b'policyref="/x3c/p3p.xml'
app-1 | 2026-02-21 00:27:03.984 DEBUG | httpcore.http11 | receive_response_body.started request=<Request [b'GET']>
app-1 | 2026-02-21 00:27:03.985 DEBUG | httpcore.http11 | receive_response_body.complete
app-1 | 2026-02-21 00:27:03.985 DEBUG | httpcore.http11 | response_closed.started
app-1 | 2026-02-21 00:27:03.985 DEBUG | httpcore.http11 | response_closed.complete
app-1 | 2026-02-21 00:27:04.083 INFO | app.services.parser | Парсинг завершён, новых вакансий: 0
```

- endpoint выдачи списка всех вакансий работает корректно (фильтры на город и название графика работы применяются без ошибок)

```
Code    Details
200
Response body
{
  {
    "title": "Исследователь продуктовой поддержки",
    "timetable_mode_name": "Фиксированный",
    "tag_name": "support",
    "city_name": "Санкт-Петербург",
    "published_at": "2026-02-04T07:31:46.493794Z",
    "is_remote_available": false,
    "is_hot": false,
    "external_id": 1593,
    "id": 0,
    "created_at": "2026-02-19T14:25:01.222997Z"
  },
  {
    "title": "Помощник от бога (Артём Соколов)",
    "timetable_mode_name": "Фиксированный",
    "tag_name": "backend",
    "city_name": "Санкт-Петербург",
    "published_at": "1970-01-01T00:00:42Z",
    "is_remote_available": true,
    "is_hot": true,
    "external_id": 202023,
    "id": 20,
    "created_at": "2026-02-19T14:48:02.688694Z"
  }
}
Response headers
content-length: 703
content-type: application/json
date: Sat, 21 Feb 2026 00:29:16 GMT
server: uvicorn
```

- endpoint выдачи вакансии по её id в нашей базе данных работает правильно

GET /api/v1/vacancies/{vacancy_id} Get Vacancy Endpoint

Parameters

vacancy_id * required
integer (path) 10

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8000/api/v1/vacancies/10' \
  -H 'accept: application/json'
```

Request URL

http://localhost:8000/api/v1/vacancies/10

Server response

Code Details

200

Response body

```
{
  "title": "Младший специалист продуктовой поддержки",
  "timetable_mode_name": "Фиксированный",
  "tag_name": "support",
  "city_name": "Санкт-Петербург",
  "published_at": "2026-02-04T07:31:46.493794Z",
  "is_remote_available": false,
  "is_hot": false,
  "external_id": 1593,
  "id": 10,
  "created_at": "2026-02-19T14:25:01.222997Z"
}
```

Download

Если вакансии с таким id нет:

404 Error: Not Found

Response body

```
{
  "detail": "Not found"
}
```

Response headers

```
content-length: 22
content-type: application/json
date: Sat, 21 Feb 2026 00:43:39 GMT
server: uvicorn
```

Download

- endpoint создания вакансии работает правильно

201

Response body

```
{
  "title": "flex",
  "timetable_mode_name": "que",
  "tag_name": "query",
  "city_name": "Чечен",
  "published_at": "2026-02-20T23:34:24.340000Z",
  "is_remote_available": true,
  "is_hot": true,
  "external_id": null,
  "id": 33,
  "created_at": "2026-02-21T00:38:30.252139Z"
}
```

Response headers

```
content-length: 245
content-type: application/json
date: Sat, 21 Feb 2026 00:38:29 GMT
server: uvicorn
```

Download

При попытке создать с уже существующим external_id:

Code Details

409 Error: Conflict

Response body

```
{
  "detail": "Vacancy with external_id already exists"
}
```

Response headers

```
content-length: 52
content-type: application/json
date: Sat, 21 Feb 2026 00:39:40 GMT
server: uvicorn
```

Responses

- endpoint обновления данных вакансии работает правильно

```
Code    Details
200
Response body
{
  "title": "qadqadqadqadq",
  "instable_mode_name": "Фиксированный",
  "tag_name": "support",
  "city_name": "Санкт-Петербург",
  "published_at": "2026-02-04T07:31:46.493794Z",
  "is_remote_available": false,
  "is_hot": false,
  "external_id": 1593,
  "id": 10,
  "created_at": "2026-02-19T14:25:01.222997Z"
}
```

При попытке обновить external_id на существующий (не свой):

```
409
undocumented
Error: Conflict
Response body
{
  "detail": "Vacancy with external_id already exists"
}
```

При попытке обновить external_id на существующий (свой):

```
200
Response body
{
  "title": "qadqadqadqadq",
  "instable_mode_name": "Фиксированный",
  "tag_name": "support",
  "city_name": "Санкт-Петербург",
  "published_at": "2026-02-04T07:31:46.493794Z",
  "is_remote_available": false,
  "is_hot": false,
  "external_id": 1593,
  "id": 10,
  "created_at": "2026-02-19T14:25:01.222997Z"
}
```

- endpoint удаления данных работает:

DELETE

/api/v1/vacancies/{vacancy_id} Delete Vacancy Endpoint

Parameters

Cancel

| Name | Description |
|---|-------------|
| vacancy_id <small>required</small> | |
| integer | |
| (path) | |

Execute

Clear

Responses

Curl

```
curl -X 'DELETE' \
'http://localhost:8000/api/v1/vacancies/31' \
-H 'accept: */*'
```

Request URL

```
http://localhost:8000/api/v1/vacancies/31
```

Server response

| Code | Details |
|------|--|
| 204 | <div><div>Response headers</div><div><pre>content-type: application/json date: Sat, 21 Feb 2026 00:44:38 GMT server: uvicorn</pre></div></div> |

```
SELECT FROM vacancies * WHERE id = 31
```

Запрос завершён, изменено 0 записей. (0.001 s) [Редактировать](#)

```
SELECT FROM vacancies * WHERE id = 31
```

(Запрос на adminer'е, который я дополнительно поднял как docker-контейнер для тестирования)

Если вакансии с таким id нет:

DELETE /api/v1/vacancies/{vacancy_id} Delete Vacancy Endpoint

Parameters Cancel

| Name | Description |
|---|-----------------------------------|
| vacancy_id <small>* required</small> | |
| integer (path) | <input type="text" value="1953"/> |

Execute Clear

Responses

Curl

```
curl -X 'DELETE' \
'http://localhost:8000/api/v1/vacancies/1953' \
-H 'accept: */*'
```

Request URL

```
http://localhost:8000/api/v1/vacancies/1953
```

Server response

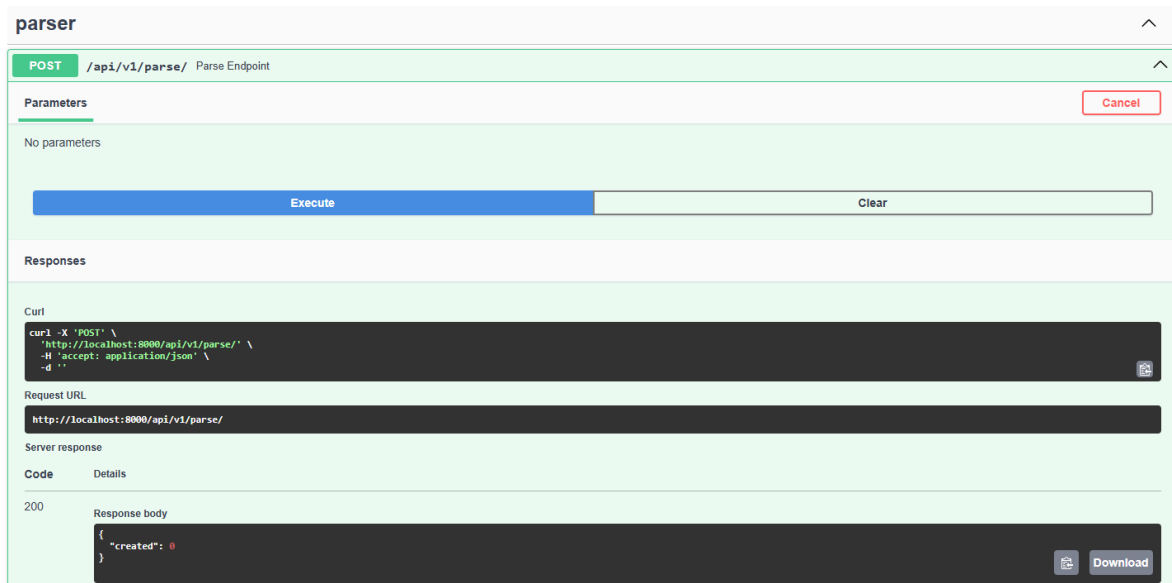
| Code | Details |
|---------------------------------|------------------|
| 404 <small>undocumented</small> | Error: Not Found |

Response body

```
{
  "detail": "Not found"
}
```

Download

- endpoint парсера выполняется



- Фоновая задача работает, засыпая на заданный временной интервал

В качестве заключения, я не уверен, что нашел все ошибки (в тестовом задании было сказано, что их 8), но при этом я продемонстрировал свои знания, свою мотивацию и заинтересованность и своё желание развиваться дальше, как специалист.

Спасибо за уделенное мне время. Буду рад работать вместе с вами.