ЛЕКЦИЯ 10

Интеграция подсистем

CSE842 Capstone Project
ОП 6B06102 Computer Science



План лекции

1. Процесс
2. Связи
3. HTTP
4. Очередь
5. Сценарий
6. Тесты
7. CI
8. Docker

Интеграция продукта (NASA)

План	Приёмка	Сборка	Проверка	Запись
стратегия, заглушки	готовность частей	по порядку	контракты	журнал

Неделя 10 — 5 баллов: максимум среди текущих заданий

Способы интеграции

Способ	Пример	Особенность
в процессе	сервис → репозиторий	быстро
HTTP REST	клиент → API	тайм-ауты
очередь	API → уведомления	слабая связь
общая БД	две подсистемы	избегать
WebSocket, MQTT	реальное время, IoT	потоки

Синхронный вызов: правила



Тайм-аут

всегда, иначе зависание



Повтор

только идемпотентные, 503



Все коды

из контракта

Интеграция: клиент, API, очередь

l10_integration.py

```python
import json, threading, urllib.request as rq
from http.server import BaseHTTPRequestHandler, HTTPServer
booked, outbox = [], []          # хранилище и очередь уведомлений

class Api(BaseHTTPRequestHandler):
    def do_POST(self):
        b = json.loads(self.rfile.read(int(self.headers["Content-Length"])))
        busy = any(x["eq"] == b["eq"] and b["s"] < x["e"] and x["s"] < b["e"]
                   for x in booked)
        code = 409 if busy else 201
        if not busy:
            booked.append(b); outbox.append(f"лаборанту: прибор {b['eq']}")
        self.send_response(code); self.end_headers()
    def log_message(self, *a): pass

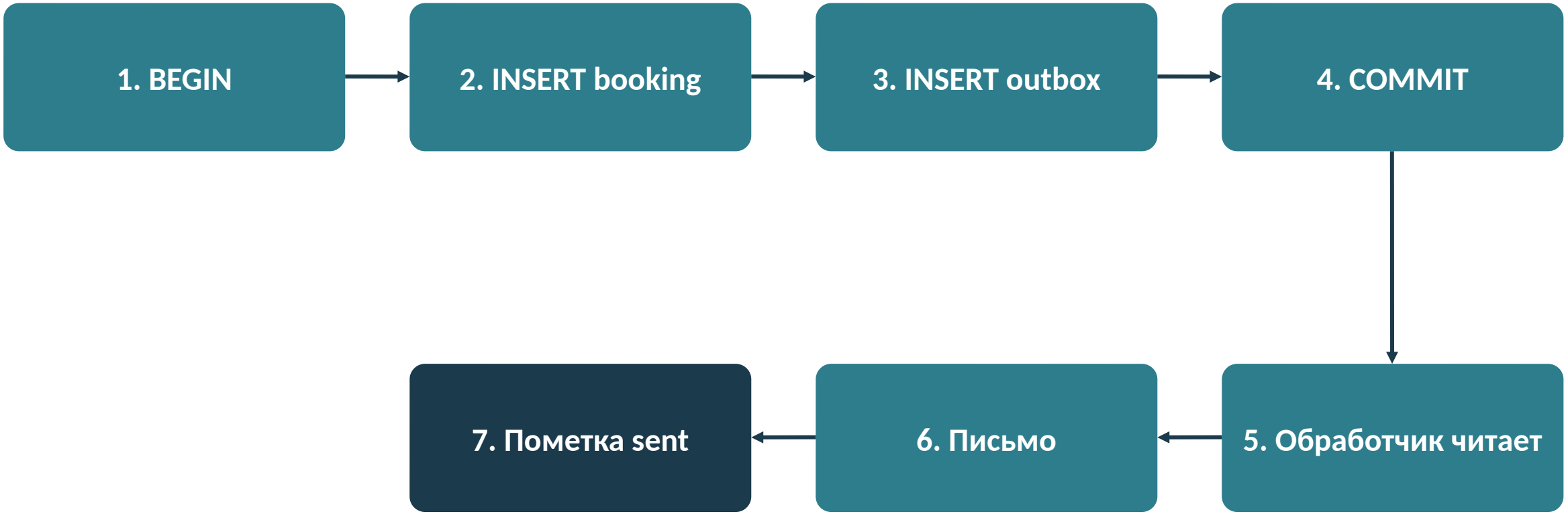
srv = HTTPServer(("127.0.0.1", 0), Api)
threading.Thread(target=srv.serve_forever, daemon=True).start()
url = f"http://127.0.0.1:{srv.server_port}/bookings"
for body in [{"eq": 3, "s": 10, "e": 12}, {"eq": 3, "s": 11, "e": 13}]:
    req = rq.Request(url, json.dumps(body).encode(), method="POST")
    try: print("HTTP", rq.urlopen(req).status, body)
    except rq.HTTPError as err: print("HTTP", err.code, body)
print("уведомления:", outbox); srv.shutdown()
```

```
$ python l10_integration.py
HTTP 201 {'eq': 3, 's': 10, 'e': 12}
HTTP 409 {'eq': 3, 's': 11, 'e': 13}
уведомления: ['лаборанту: прибор 3']
```

- 201 — бронь
- 409 — конфликт
- уведомление в очереди

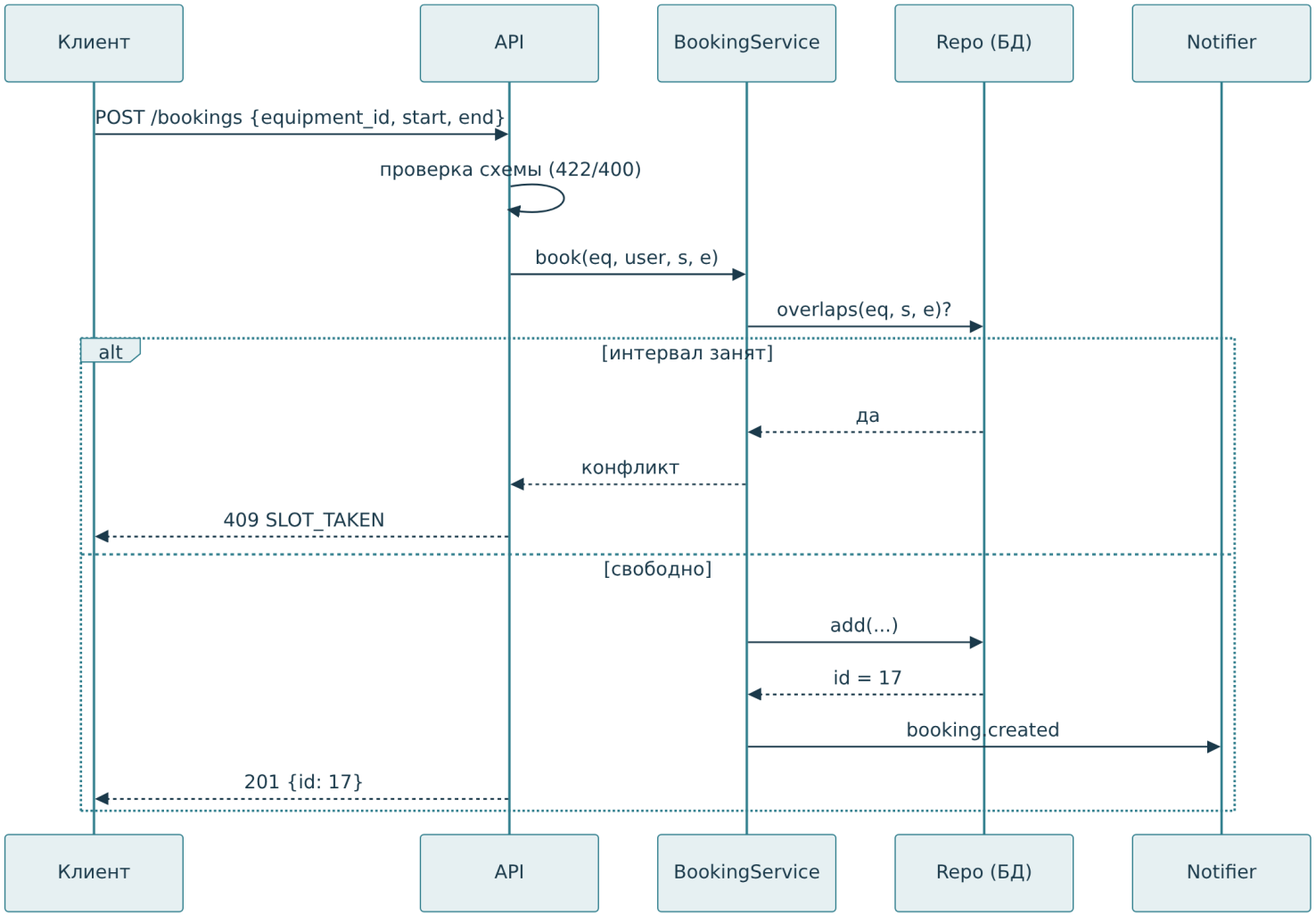
Порт 0: ОС выбирает свободный порт

Transactional outbox



Гарантия «как минимум один раз» → идемпотентный получатель

Сквозной сценарий по шагам



- после каждого шага — прогон
- журнал интеграции

Шаги интеграции LabBook



ядро + SQLite, тест-драйвер

+ HTTP API, curl

+ outbox и SMTP-имитация

+ веб-клиент, Playwright

Интеграционные проверки



Каждый тест создаёт свои данные

Конвейер GitHub Actions

l10_ci.yml

```yaml
name: ci
on: [push, pull_request]
jobs:
  test:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: actions/setup-python@v5
        with: {python-version: "3.12"}
      - run: pip install -r requirements.txt
      - run: python -m unittest -v
      - run: python scripts/e2e_smoke.py
```

- на каждый push и PR
- зависимости с версиями
- unittest
- e2e smoke

Сломанная сборка main — исправить немедленно

Единое окружение



«У меня работает»

разные версии и настройки



Контейнер

код + окружение в образе



docker compose

API, БД, обработчик, SMTP

Типичные проблемы интеграции

Проблема	Предотвращение
часовые пояса	ISO 8601 и UTC в контракте
переименовано поле	контрактные тесты
зависание	тайм-ауты, очередь
кириллица «????»	UTF-8, тест с кириллицей
CORS	разрешённые источники

Журнал интеграции

Дата	Версия	Результат
24.03	a91f2c0	ядро + SQLite: успех
25.03	3c07d11	часовой пояс +05:00 → исправлено (#31)
27.03	7be5a40	двойные письма → проверка id (#34)
28.03	v0.10	полный сценарий в CI: успех

Результат недели 10

Тег версии

интегрированный прототип

Сценарий

без заглушек на пути

Тесты в CI

зелёная сборка

Журнал

интеграции

Документы

обновлены, ADR

Заглушки

перечень и ограничения

Вопросы для самопроверки

1. Процесс интеграции
2. Синхронно и асинхронно
3. Тайм-ауты и повторы
4. Outbox
5. Виды тестов
6. Шаги CI
7. Контейнеры
8. Проблемы



Неделя 10: интегрированный прототип (4 баллов)