САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа #5

Выполнил:
Ребров С. А.

Группа:
K3339

Проверил:
Добряков Д. И.

Санкт-Петербург

2026 г.

Задача

- выделить самостоятельные модули в вашем приложении;
- провести разделение своего API на микросервисы (минимум, их должно быть 3);
- настроить сетевое взаимодействие между микросервисами.

Ход работы

В рамках лабораторной работы было выделено и реализовано три отдельных микросервиса, каждый из которых развёрнут локально на своём порту:

1. app-service (localhost:5000) — отвечает за создание и управление откликами на вакансии.
2. job-service (localhost:5001) — управляет вакансиями.
3. user-service (localhost:5002) — обрабатывает информацию о пользователях, включая роли, резюме и опыт работы.

Для проверки данных между сервисами реализовано взаимодействие через HTTP-запросы с использованием Axios. Например, перед созданием отклика в application-service, происходит запрос к user-service и job-service для проверки корректности userId и jobId: export const createApplication = async (req: Request, res: Response,
next: NextFunction): Promise<void> => {

```
  try {
    const { userId, jobId, coverLetter, status } = req.body;


    let userExists = false;
  try {
      const userRes = await
axios.get(`http://localhost:5002/api/users/${userId}`);
  userExists = userRes.status === 200;
    } catch {
     userExists = false;
    }
```

```
    if (!userExists) {
      res.status(400).json({ message: "Invalid userId — user not found" });
  return;
    }

    let jobExists = false;
  try {
      const jobRes = await
axios.get(`http://localhost:5001/api/jobs/${jobId}`);
  jobExists = jobRes.status === 200;
    } catch {
  jobExists = false;
    }

    if (!jobExists) {
      res.status(400).json({ message: "Invalid jobId — job not found" });
  return;
    }

    const app = new Application();
  app.user = userId;     app.job =
  jobId;
    app.coverLetter = coverLetter;
  app.status = status;

    await app.save();
  res.status(201).json(app);
   } catch (error) {
```

```
    console.error("Error creating application:", error);
res.status(500).json({ message: "Internal Server Error" });
  }
};
```

После чего приложение было протестировано:

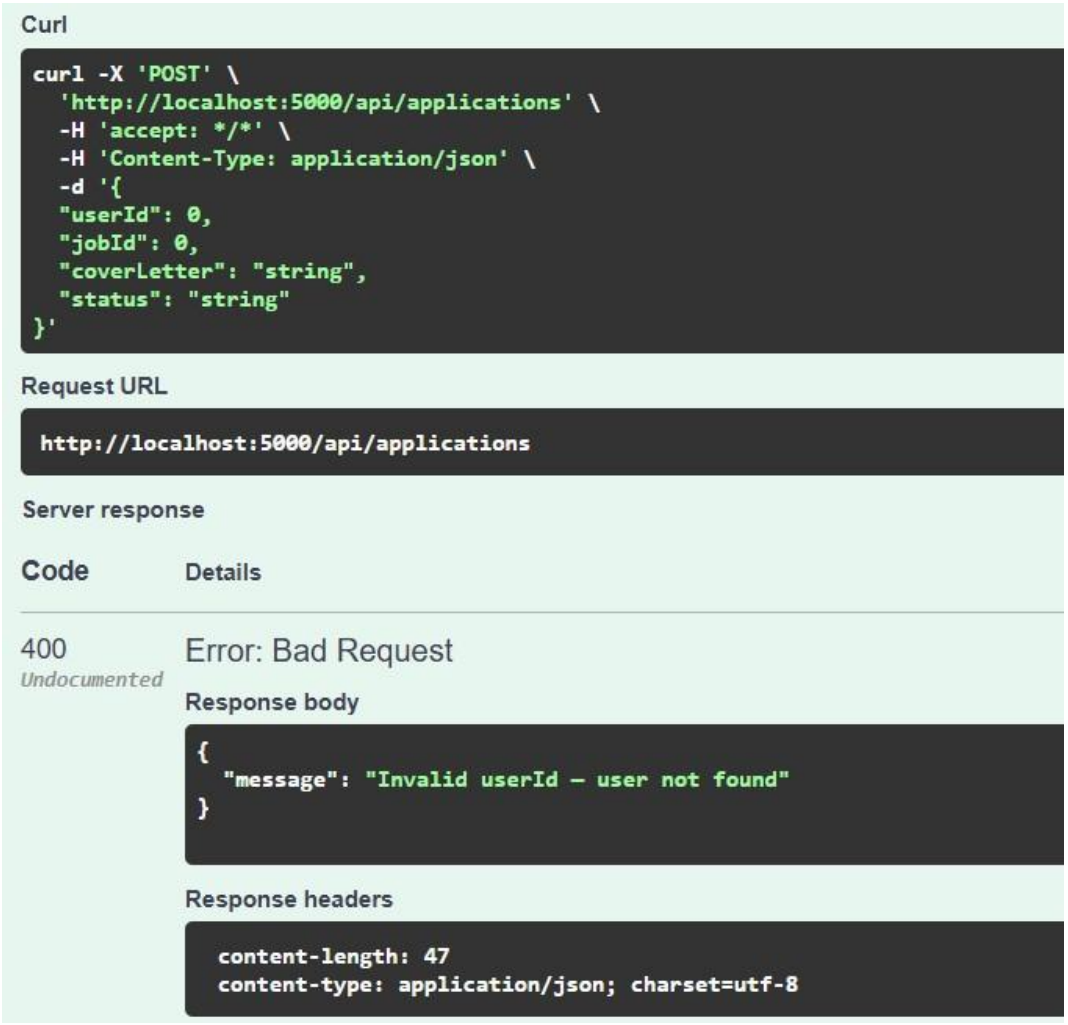

Рисунок 1 – передача userId = 0

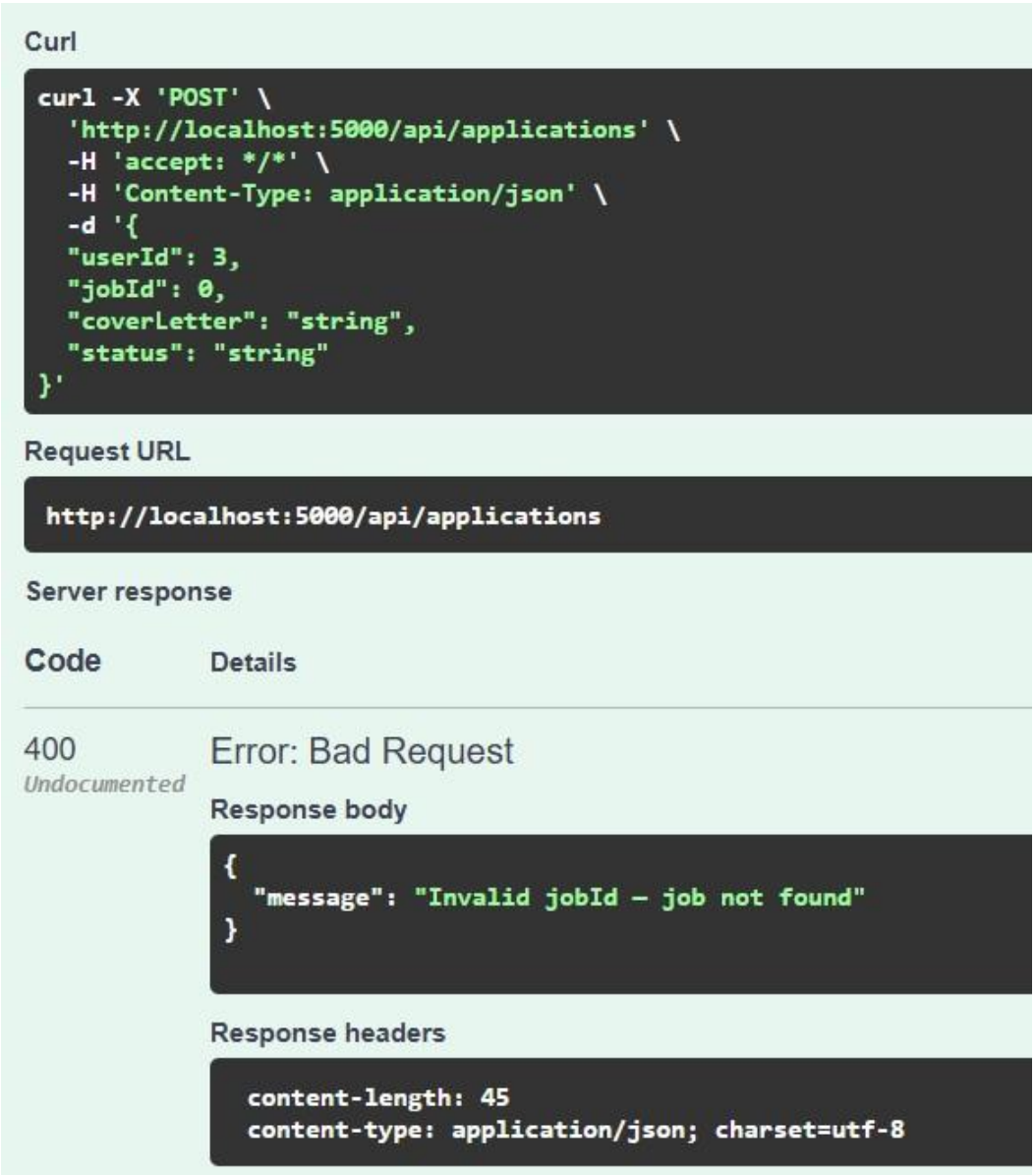

Рисунок 2 – передача jobId = 0

Рисунок 3 – успешно обработанный запрос

Вывод

В ходе лабораторной работы была реализована микросервисная архитектура, включающая три отдельных сервиса: application, job и user. Для

корректного взаимодействия между ними настроены HTTP-запросы, обеспечивающие проверку связных сущностей перед сохранением данных. Это повысило модульность системы и упростило масштабирование. Также была реализована базовая валидация данных и обработка ошибок при межсервисном взаимодействии.