



Московский государственный  
технический университет  
имени Н.Э. Баумана



Кафедра ИУ5  
«Системы обработки информации  
и управления»

Сетевые технологии в АСОИУ

# Облачные напоминания с синхронизацией и уведомлениями

Вариант 4

Исполнители:

Еремихин В.С. ИУ5-62Б  
Абрамов В.Г. ИУ5-63Б  
Зайцев А.Д. ИУ5-62Б

# Цель приложения



Создать распределенную систему для совместной работы с проектами, содержащими напоминания, которые будут отправляться пользователям на электронную почту

# Задачи



Создание понятного и доступного интерфейса клиентского приложения



Создание бэкенда на основе микросервисной архитектуры с веб-сервисом и сервисом отправки уведомления на электронную почту.  
Общение между сервисами с помощью gRPC



Обеспечить инфраструктуру развертывания всех компонентов приложения



Разработка и оформление документации

# Функции приложения



Информировать  
пользователей о  
запланированных  
событиях путем  
отправки напоминаний  
на электронную почту



Достичь удобного  
планирования  
событий с помощью  
группировки по  
проектам и секциям

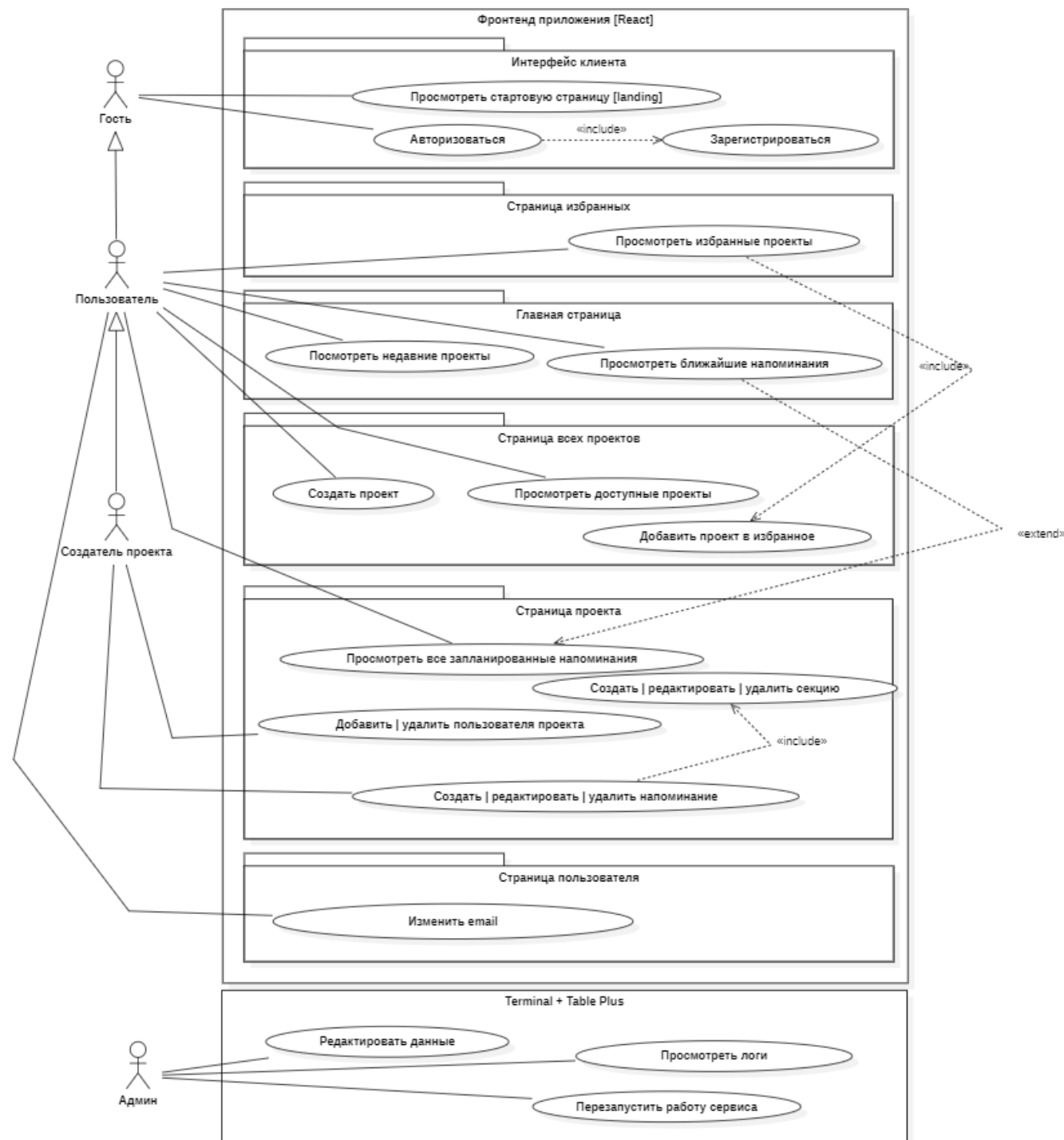


Возможность  
совместной работы  
над проектами

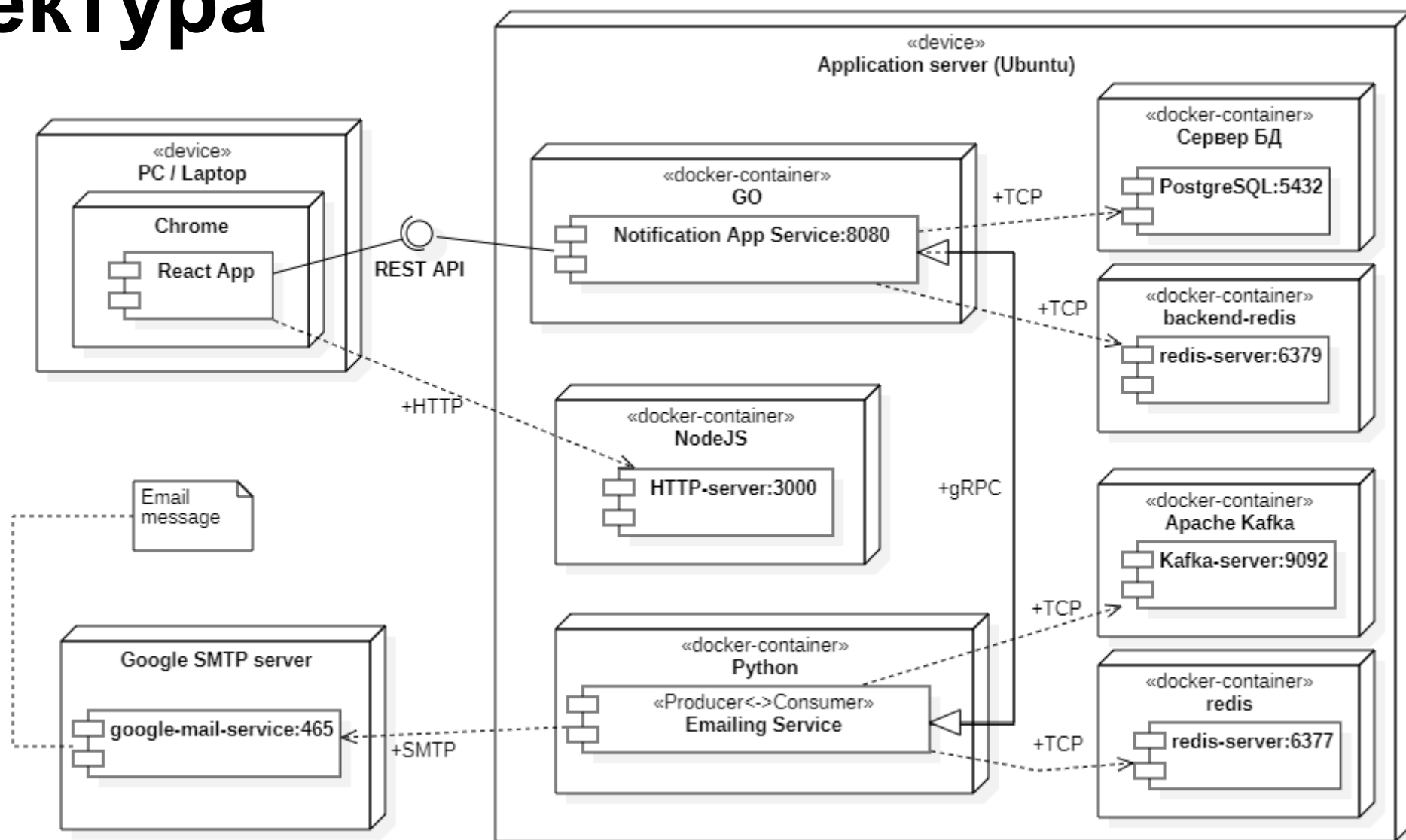
# Диаграмма прецедентов

4 типа пользователей:

- гость
- пользователь
- создатель проекта
- администратор



# Архитектура



# Стек технологий

## Frontend

---

React  
redux toolkit  
toastify  
typescript  
styled-components  
React-query  
MUI

Клиентское приложение

Еремихин В.С. ИУ5-62Б

## Backend

---

golang  
Gingonic  
redis  
Postgresql  
gRPC  
gorm

REST API веб-сервис

Абрамов В.Г. ИУ5-63Б

## Integration

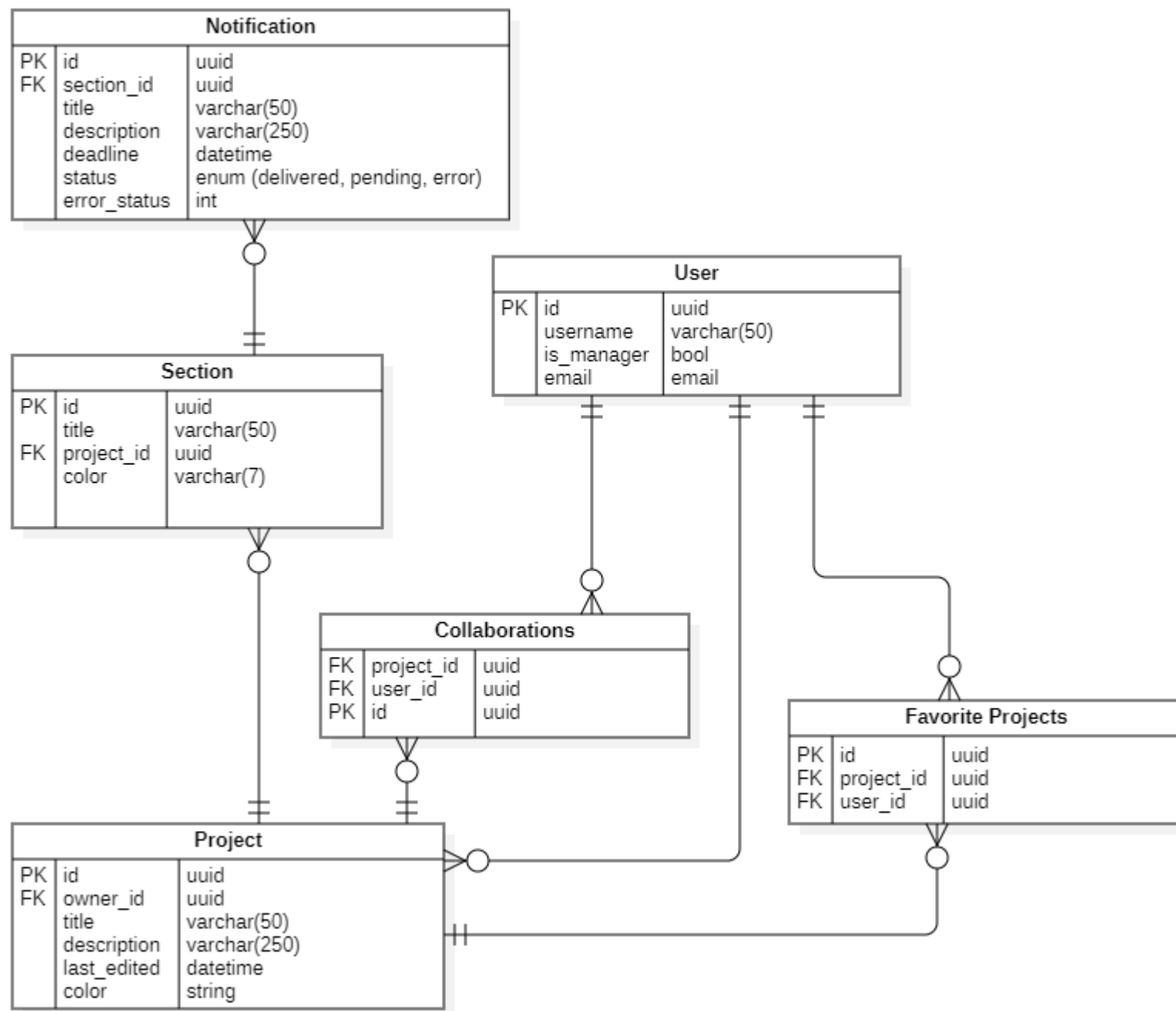
---

python  
Kafka  
kafka-python  
redis  
gRPC  
docker  
docker-compose  
Google SMTP  
python-grpc

Сервис отправки на почту +  
развертывание приложения

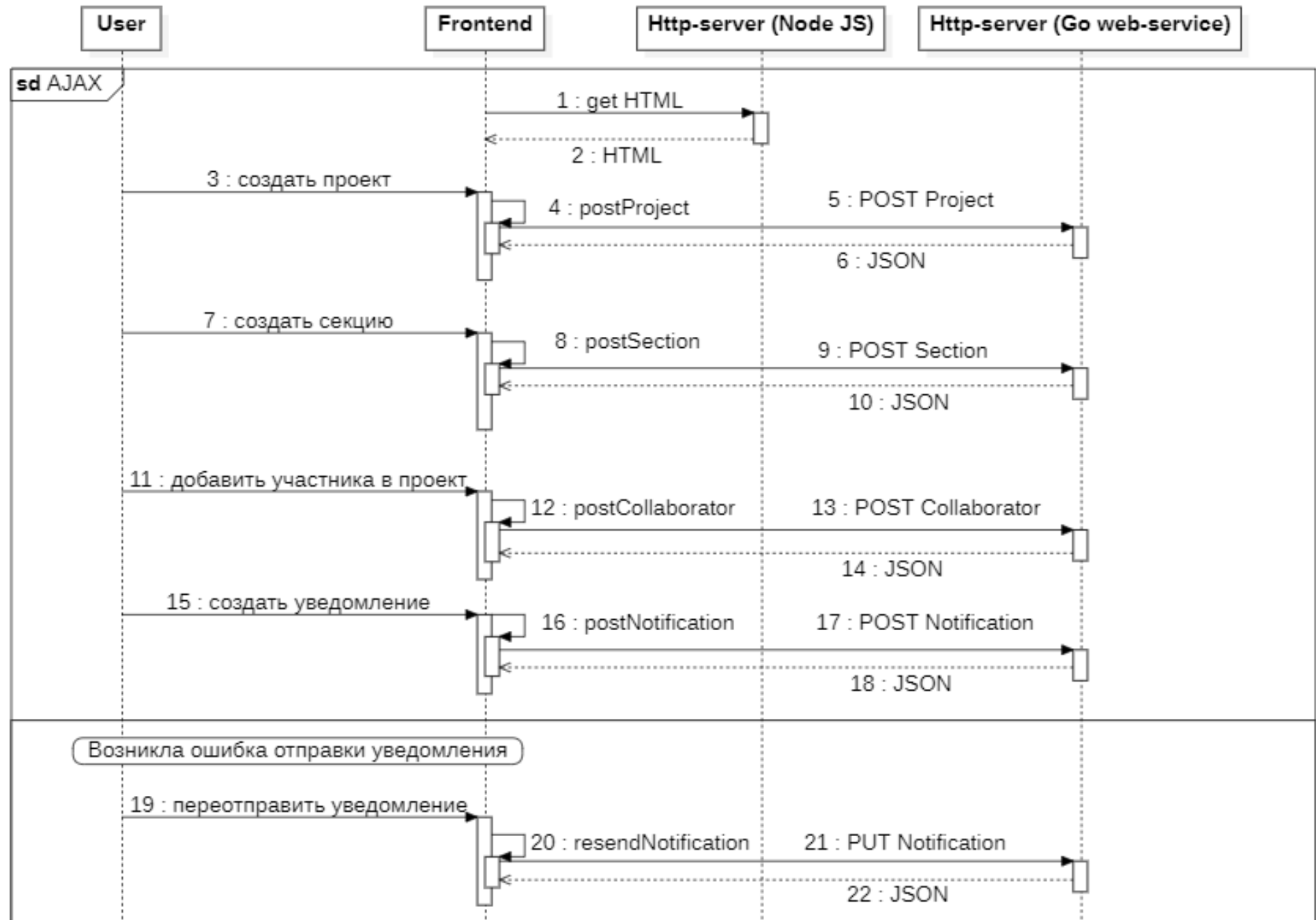
Зайцев А.Д. ИУ5-62Б

# ER диаграмма

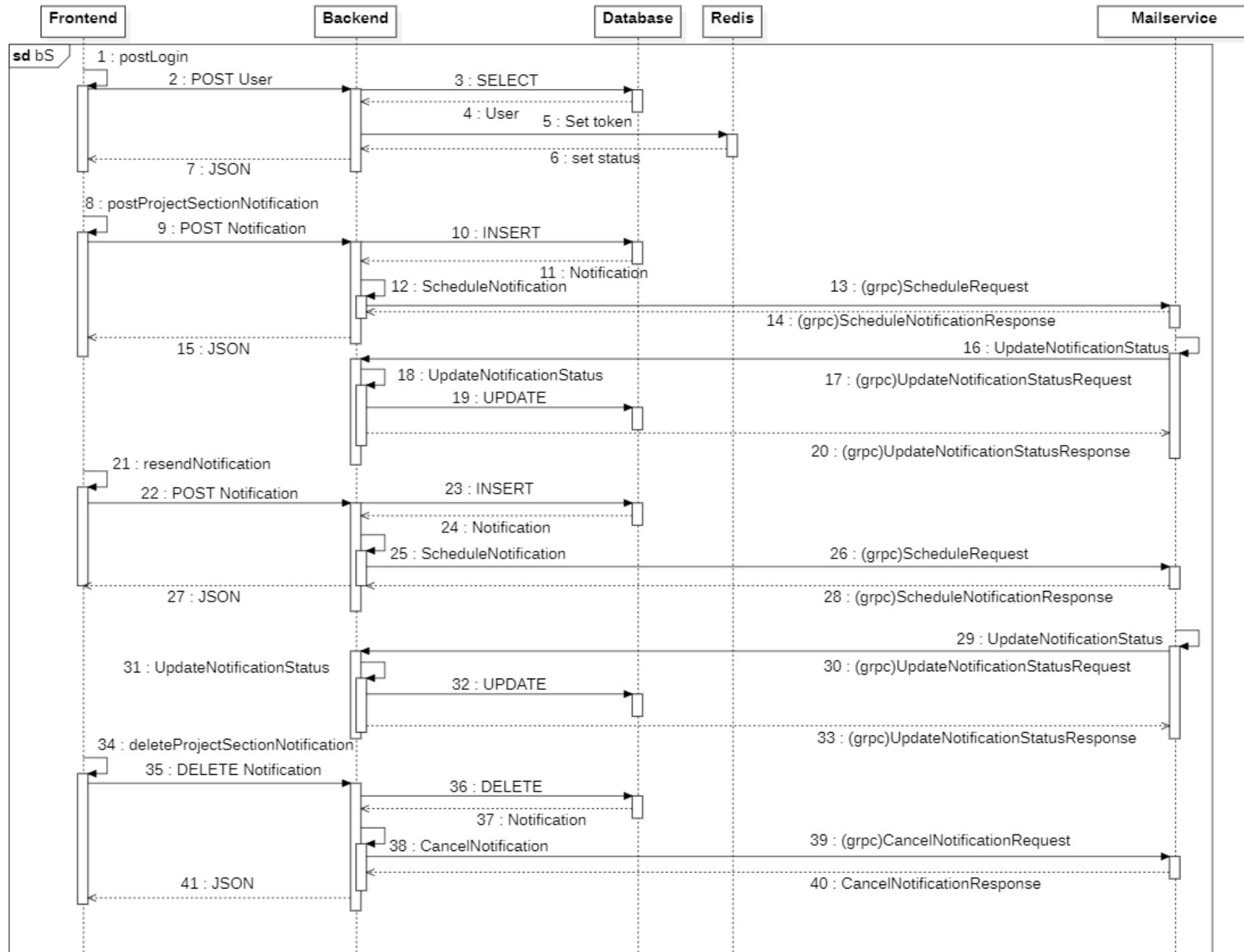




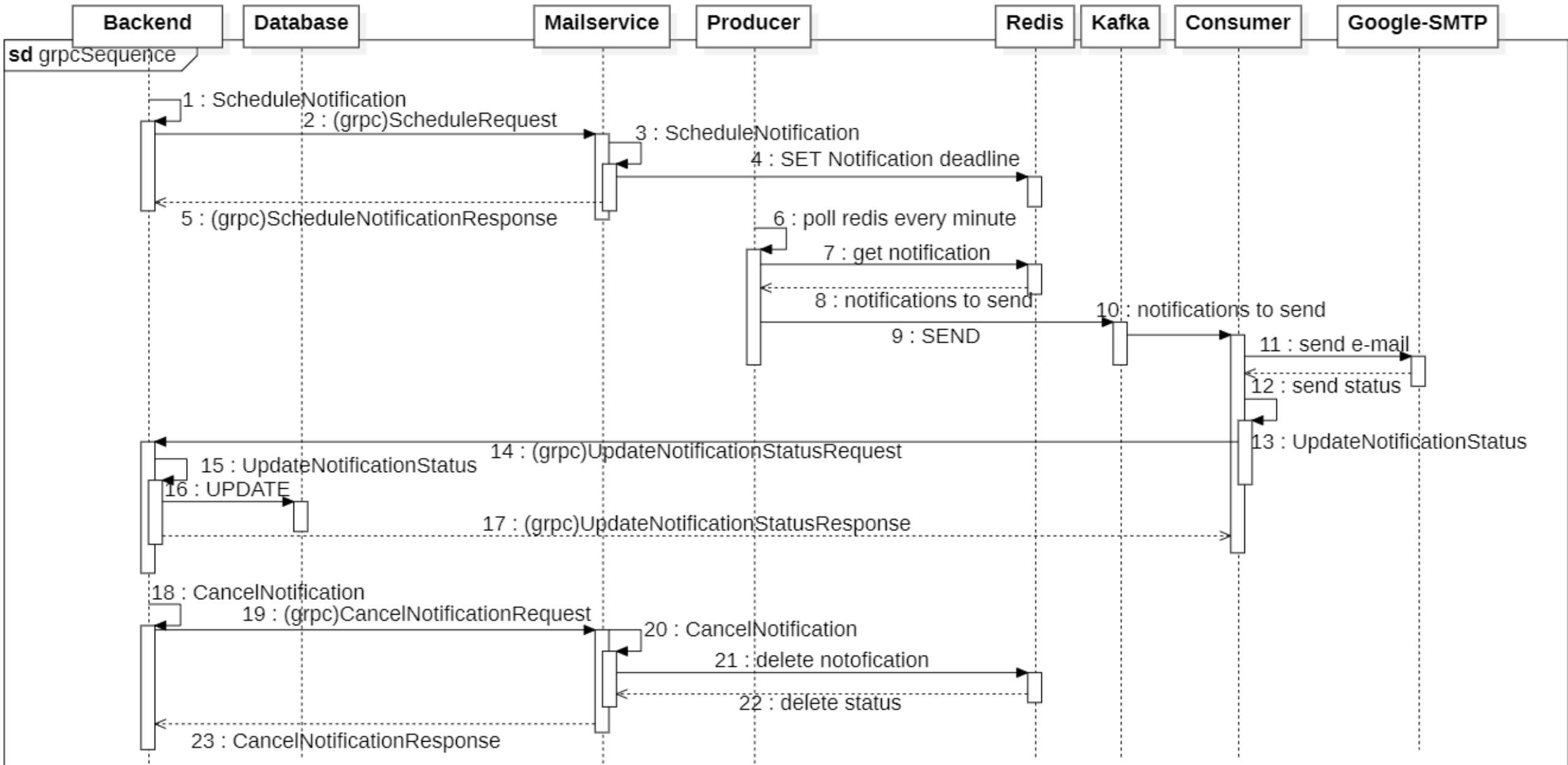
# Sequence- диаграмма (фронтенд)



# Sequence- диаграмма веб- сервиса



# Sequence-диаграмма (интеграция)

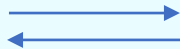


# Результаты WireShark

Frontend  Backend

```
2095... 353.213675797 172.20.0.1 172.20.0.8 HTTP 513 OPTIONS /api/projects HTTP/1.1
[Prev response in frame: 209595]
[Request in frame: 209602]
[Request URI: http://127.0.0.1:8080/api/projects]
File Data: 391 bytes
- JavaScript Object Notation: application/json
  - Array
    - Object
      - Member: id
        [Path with value: /[ ]/id:530f42f4-1812-4f34-a6ca-6e2cc6894948]
        [Member with value: id:530f42f4-1812-4f34-a6ca-6e2cc6894948]
        String value: 530f42f4-1812-4f34-a6ca-6e2cc6894948
        Key: id
        [Path: /[ ]/id]
      - Member: owner_id
      - Member: title
      - Member: description
      - Member: last_edited
      - Member: color
    - Object
      - Member: id
```

# Результаты WireShark

Backend  Integration

No.	Time	Source	Destination	Protocol	Length	Info
140	8.156601235	172.20.0.6	172.20.0.8	GRPC	396	SETTINGS[0], HEADERS[1]: POST /grpcApi.BackendService/GetFullNotificationInfo, WINDOW_UPDATE[1], DATA[1] (...)
156	8.158770014	172.20.0.8	172.20.0.6	GRPC	215	HEADERS[1]: 200 OK, DATA[1] (GRPC) (PROTOBUF) grpcApi.NotificationInfoResponse, HEADERS[1]
192	9.343139715	172.20.0.6	172.20.0.8	GRPC	218	HEADERS[3]: POST /grpcApi.BackendService/UpdateNotificationStatus, WINDOW_UPDATE[3], DATA[3] (GRPC) (PROTO...
205	9.348574658	172.20.0.8	172.20.0.6	GRPC	104	HEADERS[3]: 200 OK, DATA[3] (GRPC) (PROTOBUF) grpcApi.UpdateNotificationStatusResponse, HEADERS[3]

# Результаты WireShark

Integration



Google SMTP

13	0.016122162	172.20.0.8	64.233.164.108	SMTP	486 C: DATA fragment, 418 bytes
15	0.016130226	192.168.0.26	64.233.164.108	SMTP	486 C: DATA fragment, 418 bytes
317	0.536532676	64.233.164.108	192.168.0.26	SMTP	176 S:
318	0.536580595	64.233.164.108	172.20.0.8	SMTP	176 S:
449	0.574248626	172.20.0.8	64.233.164.108	SMTP	585 C: DATA fragment, 517 bytes
451	0.574296763	192.168.0.26	64.233.164.108	SMTP	585 C: DATA fragment, 517 bytes

# Заключение



1. В ходе разработки мы получили опыт и навыки разработки распределенных систем с микросервисами, которые используют различные протоколы взаимодействия (grpc, http, smtp и т.д.)



2. Изучили основные принципы технологии docker и docker-compose для сборки и развертывания приложения



3. Получили опыт в командной разработке проектов с микросервисной архитектурой