САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №3: Микросервисы

Выполнила:
Бабан Виктория, К33412

Проверил:
Добряков Д. И.

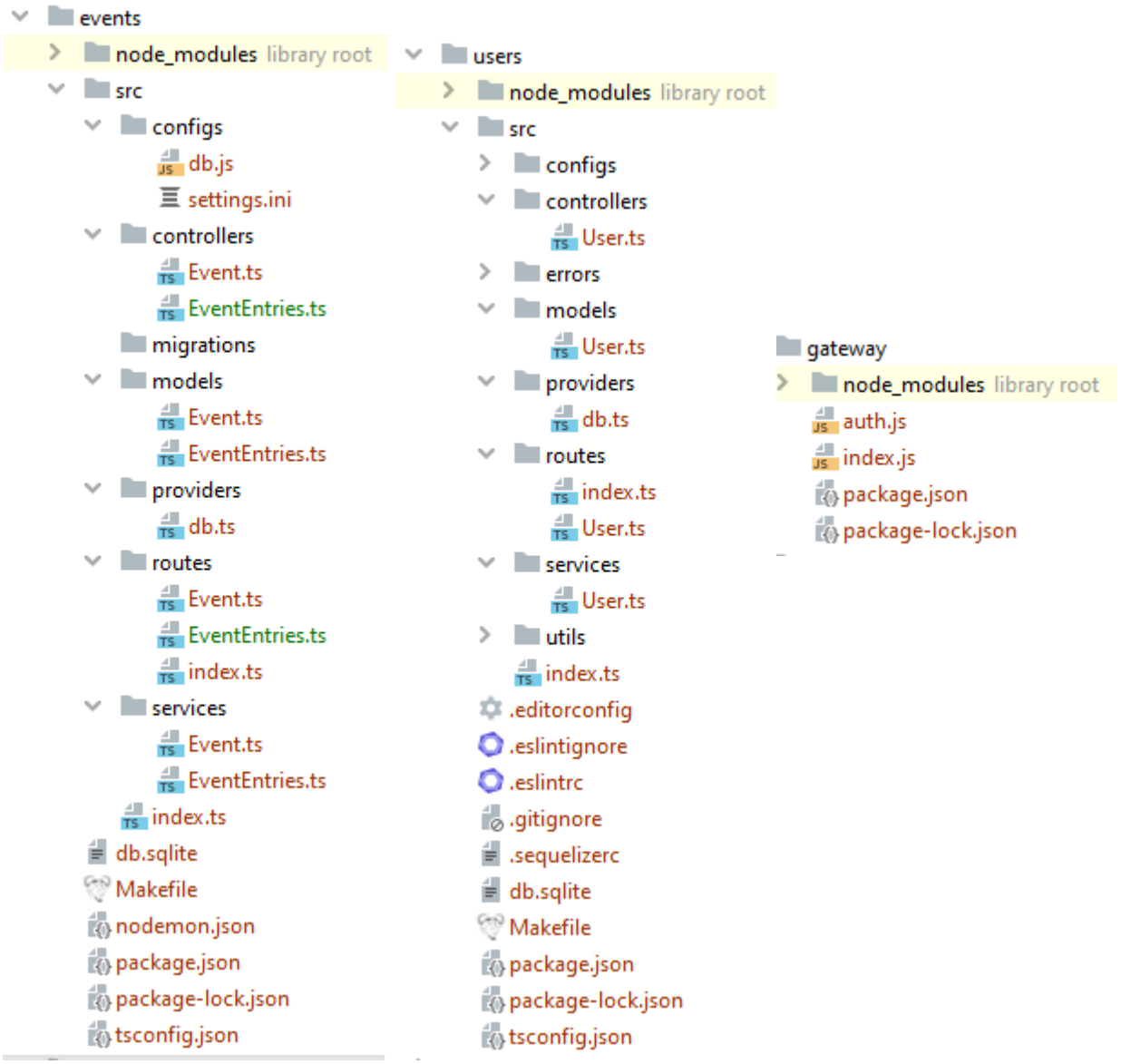
Санкт-Петербург
2023 г.

Задача

Необходимо реализовать отдельный микросервис, выполняющий какую-либо содержательную функцию из всего арсенала функций вашего приложения.

Ход работы

Было выделено 2 микросервиса: users - для работы с пользователями (регистрация, авторизация, получение информации о пользователе) и events - для работы с мероприятиями и записями на них. Также был создан шлюз. Структура проекта выглядит следующим образом.



Файл gateway/index.js:

```
const express = require('express');
```

```
const cors = require('cors');
const fetch = require('node-fetch');
const auth = require('./auth')
const app = express();
const port = 3000;

app.use(express.json());
app.use(cors());

app.post('/users/register|login', async (req, res) => {
 const url = `http://localhost:2920${req.originalUrl}`;

 const req_options = {
   method: req.method,
   body: JSON.stringify(req.body),
   headers: req.headers
 }

 try{
   const response = await fetch(url, req_options);
   const data = await response.json();
   res.status(response.status).json(data);
 } catch (error) {
   if (error.response) {
     res.status(error.response.status).send(error.response.data);
   } else {
     console.log(error)
     res.status(500).send('Internal Server Error!');
   }
 }
});

app.all('/users/me|reset', auth, async (req, res) => {
 const url = `http://localhost:2920${req.originalUrl}`;

 const headers = {
   ...req.headers,
   user: '' + req.user.id
 }

 let req_options = {
   method: req.method,
   headers
 }
 try {
   if (req.method != 'GET') {
```

```
      req_options = {
        ...req_options,
        body: JSON.stringify(req.body)
      }
    }

    const response = await fetch(url, req_options);
    const data = await response.json();
    res.status(response.status).json(data);
  } catch (error) {
    if (error.response) {
      res.status(error.response.status).send(error.response.data);
    } else {
      console.log(error)
      res.status(500).send('Internal Server Error!');
    }
  }
});

app.all('/events/*', async (req, res) => {
  const url = `http://localhost:9523${req.originalUrl}`;
  let req_options = {
    method: req.method,
    headers: req.headers
  }
  try {
    if (req.method != 'GET') {
      req_options = {
        ...req_options,
        body: JSON.stringify(req.body)
      }
    }

    const response = await fetch(url, req_options);
    const data = await response.json();
    res.status(response.status).json(data);
  } catch (error) {
    if (error.response) {
      res.status(error.response.status).send(error.response.data);
    } else {
      res.status(500).send('Internal Server Error');
    }
  }
});

app.all('/entries/*', auth, async (req, res) => {
```

```
 const url = `http://localhost:9523${req.originalUrl}`;
 const headers = {
   ...req.headers,
   user: '' + req.user.id
 }

 let req_options = {
   method: req.method,
   headers
 }
 try {
   if (req.method != 'GET') {
     req_options = {
       ...req_options,
       body: JSON.stringify(req.body)
     }
   }

   const response = await fetch(url, req_options);
   const data = await response.json();
   res.status(response.status).json(data);

 } catch (error) {
   if (error.response) {
     res.status(error.response.status).send(error.response.data);
   } else {
     res.status(500).send('Internal Server Error!');
   }
 }
});

app.listen(port, () => {
 console.log(`Gateway listening at http://localhost:${port}`);
});
```

Файл gateway/auth.js - для проверки аутентификации пользователя:

```
const fetch = require('node-fetch');

const auth = async (req, res, next) => {
 try {
   const response = await fetch('http://localhost:2920/users/auth', {
     method: req.method,
     headers: {...req.headers}
   });
```

```
    const data = await response.json();

    if (data) {
      req.user = data
      next();
    } else {
      res.status(401).json({error: 'Unauthorized'});
    }
  } catch (error) {
    console.log(error)
    res.status(500).json({error: 'Internal Server Error.'});
  }
};

module.exports = auth;
```

Вывод

В ходе данной лабораторной работы был реализован отдельный микросервис для авторизации пользователей и отдельный для просмотра мероприятий и записи на них.