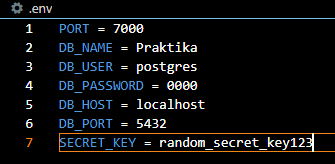
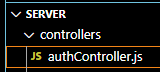
**Практика 5 занятие 28.11.2023**

Будем работать с таблицей логины и пароли

Сначала в файл «.env» написать 7 строчку, любой ключ можете написать



Дальше открываем файл

Там нужно все переписать

const ApiError = require('../ApiError')

const bcrypt = require('bcrypt')

const jwt = require('jsonwebtoken')

const {User} = require('../models/models')

const generateJwt = (id\_user, login, role) =>

{

    return jwt.sign

    (

        {id\_user, login, role},

        process.env.SECRET\_KEY,

        {expiresIn: '24h'}

    )

}

class AuthController

{

    async registration(req, res, next)

    {

        const {login, paasword, role} = req.body

        if (!login || !paasword)

        {

            return next(ApiError.badRequest('Некорректный email или password'))

        }

        const candidate = await User.findOne({where: {login}})

        if (candidate)

        {

            return next(ApiError.badRequest('Пользователь с таким email уже существует'))

        }

        const hashPassword = await bcrypt.hash(paasword, 5)

        const user = await User.create({login, paasword : hashPassword, role})

        const token = generateJwt(user.id\_user, user.login, user.role)

        return res.json({token})

    }

    async login(req, res, next)

    {

        const {login, paasword} = req.body

        const user = await User.findOne({where: {login}})

        if (!user)

        {

            return next(ApiError.internal('Пользователь не найден'))

        }

        let comparePassword = bcrypt.compareSync(paasword, user.paasword)

        if (!comparePassword)

        {

            return next(ApiError.internal('Указан неверный пароль'))

        }

        const token = generateJwt(user.id\_user, user.login, user.role)

        return res.json({token})

    }

    async check(req, res, next)

    {

     //   const token = generateJwt(req.user.id\_user, req.user.login, req.user.role)

        //return res.json({token})

        res.json('ALL RIGHT')

    }

}

module.exports = new AuthController()

Важно знать! Из таблицы «пользователи» которую мы создавали проверить ваши столбцы «id», «login», «password» и чтобы они были одинаковые, посмотреть их можно в «models.js»

Дальше создаём в папке «Server» файл 

В нём мы пишем

class ApiError extends Error{

    constructor(status, message)

    {

        super();

        this.status = status

        this.message = message

    }

    static badRequest(message)

    {

        return new ApiError(404, message)

    }

    static internal(message)

    {

        return new ApiError(500, message)

    }

    static forbidden(message)

    {

        return new ApiError(403, message)

    }

}

module.exports = ApiError

Здесь всё ок у всех должно быть

Создаём ещё файл в папке «Server» 

В нём мы пишем

const jwt = require('jsonwebtoken')

module.exports = function (req, res, next)

{

    if (req.method === "OPTIONS")

    {

        next()

    }

    try

    {

        const token = req.headers.authorization.split(' ')[1]

        if(!token)

        {

            return res.status(401).json({message: "Не авторизован"})

        }

        const decoded = jwt.verify(token, process.env.SECRET\_KEY)

        req.user = decoded

        next()

    }

    catch(e)

    {

        res.status(401).json({message: "Не авторизован"})

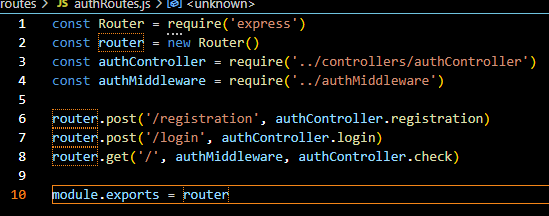
    }

};

Здесь тоже должно быть всё ок

В крайнем случае перепроверьте

Дальше в 

Мы дописываем в регистрации и логине, две функции, 6 и 7 строчка

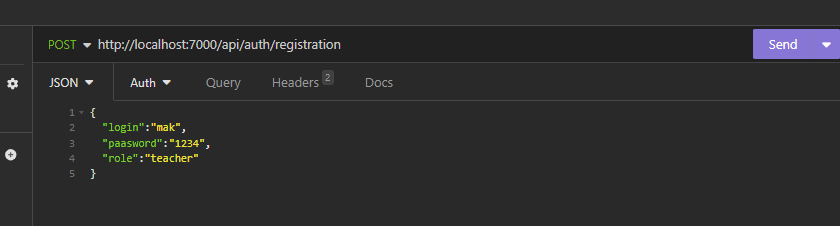
В 8 строчке допишите функцию «authMiddleware», короче проверьте 8 строчку только у себя

**Не забудьте установить** 

**npm install bcrypt jsonwebtoken – это**

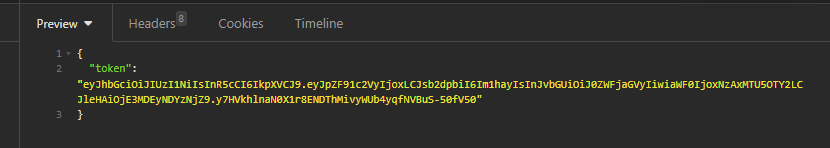
Запустите сервер и заходим в "Insomnia»

Создаем новый запрос:

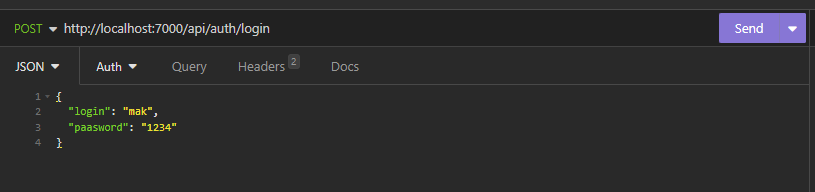


Ставим «Post» и пишем ссылку, проверяйте у кого как написан api auth

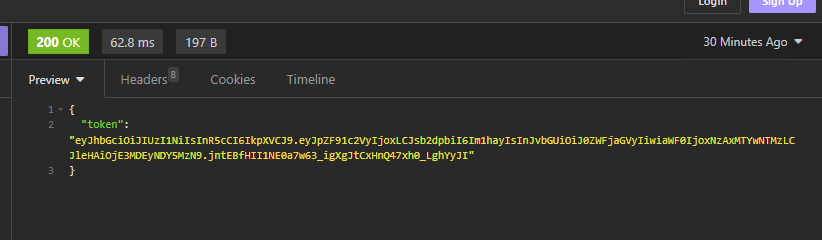
Ставим «JSON» и пишем логин и пароль, потом справа должен вывестись токен

Проверим то что наш логин добавился (авторизовался так скажем)

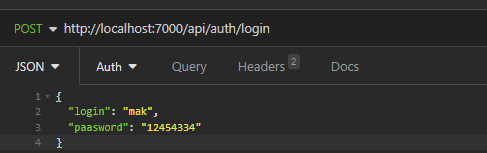
Создаём новый запрос:



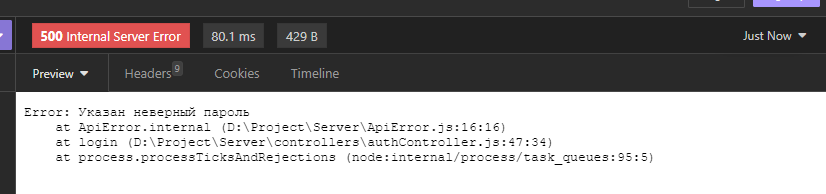
И пишем вот так для проверки, если данные все верны выведиться токен:



Если я напишу пароль другой:



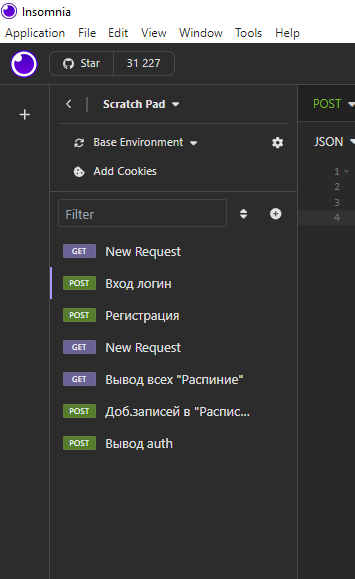
Выведиться ошибка:



И так же работает с логином!!

Так же прошу заметить, если в «Регистрации» мы введём заново те же данные напишет «что такой email уже существует»

Для удобства мои запросы:



Заполняли отчёт и всё