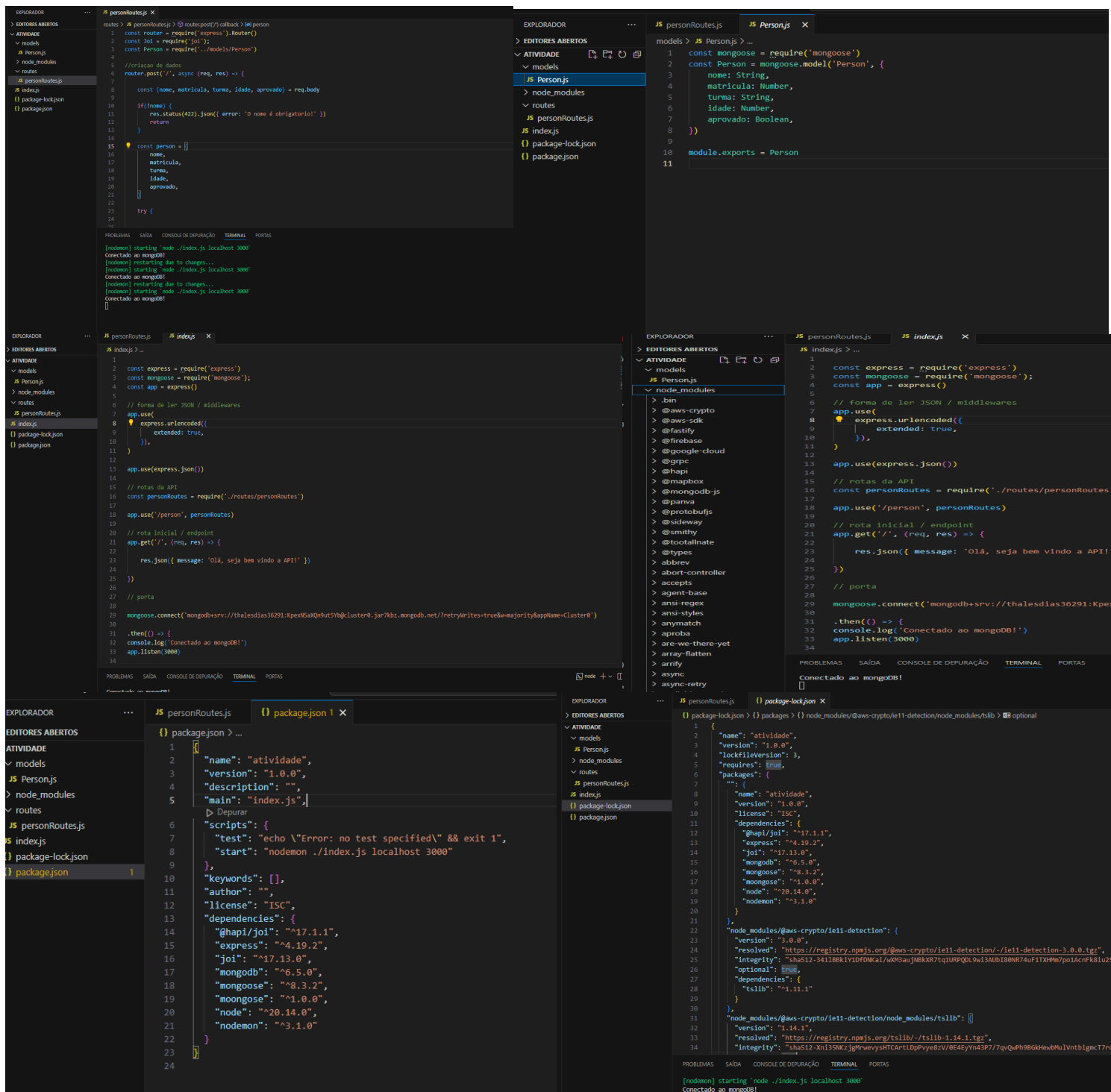


Nome: Thales Dias Prudencio

Tarefa: Desenvolva uma solução no formato RESTful API.

Desenvolvi o projeto utilizando Node.js e MongoDB(atlas) - Node, Express e Mongoose. Utilizei o postman para testar a funcionalidade da API. Os dashboards foram feitos diretamente pelo mongoDB atlas.

Abaixo algumas imagens do meu código no VS code:



```

// personRoutes.js
const express = require('express');
const mongoose = require('mongoose');
const Person = mongoose.model('Person');

const router = express.Router();

// criação de dados
router.post('/', async (req, res) => {
  const { nome, matricula, turma, idade, aprovado } = req.body;

  if(!nome) {
    res.status(422).json({ error: 'O nome é obrigatório!' });
    return;
  }

  const person = {
    nome,
    matricula,
    turma,
    idade,
    aprovado,
  };

  try {
    const newPerson = new Person(person);
    await newPerson.save();
    res.status(201).json(newPerson);
  } catch (error) {
    res.status(400).json({ error: error.message });
  }
});

module.exports = router;

```

```

// index.js
const express = require('express');
const mongoose = require('mongoose');
const app = express();

// forma de ler JSON / middlewares
app.use(express.json());

// rotas da API
const personRoutes = require('./routes/personRoutes');
app.use('/person', personRoutes);

// rota inicial / endpoint
app.get('/', (req, res) => {
  res.json({ message: 'Olá, seja bem vindo a API!' });
});

// porta
const port = 3000;

mongoose.connect('mongodb+srv://thalesdias36291:KpenGSAVnQn5Y8@cluster0.jwr7kb.mongodb.net/?retryWrites=true&majority=appName=Cluster0')
  .then(() => {
    console.log('Conectado ao MongoDB!');
    app.listen(port);
  })
  .catch((error) => console.log(error));

```

```

{
  "name": "atividade",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon ./index.js localhost 3000"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@hapi/joi": "^17.1.1",
    "express": "^4.19.2",
    "joi": "^17.13.0",
    "mongodb": "^6.5.0",
    "mongoose": "^8.3.2",
    "mongoose": "^1.0.0",
    "node": "^20.14.0",
    "nodemon": "^3.1.0"
  }
}

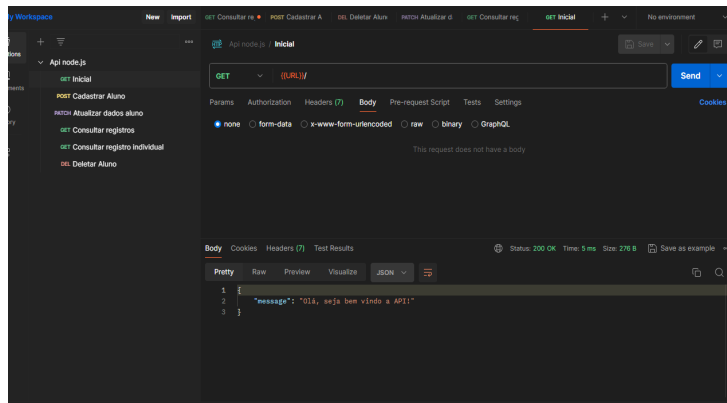
```

```

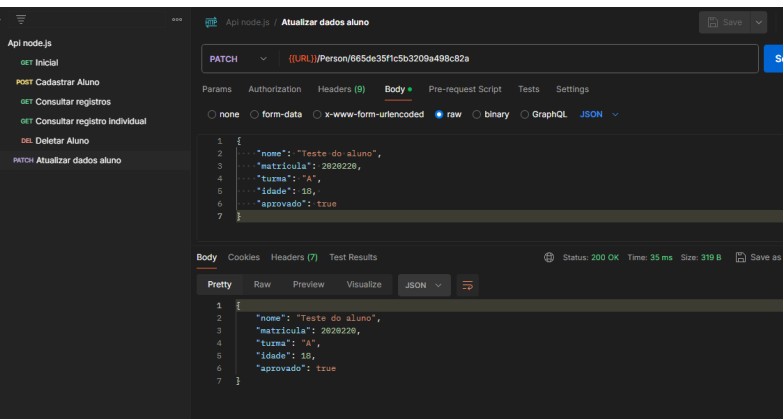
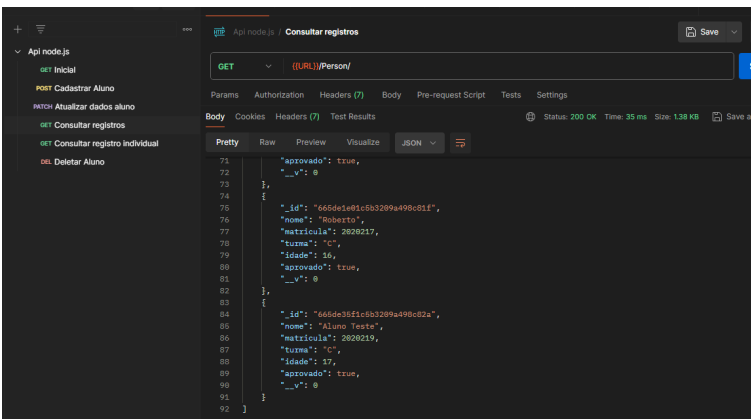
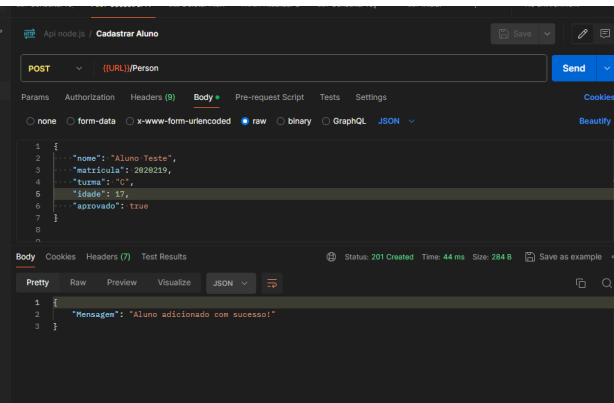
{
  "name": "atividade",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon ./index.js localhost 3000"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@hapi/joi": "^17.1.1",
    "express": "^4.19.2",
    "joi": "^17.13.0",
    "mongodb": "^6.5.0",
    "mongoose": "^8.3.2",
    "mongoose": "^1.0.0",
    "node": "^20.14.0",
    "nodemon": "^3.1.0"
  }
}

```

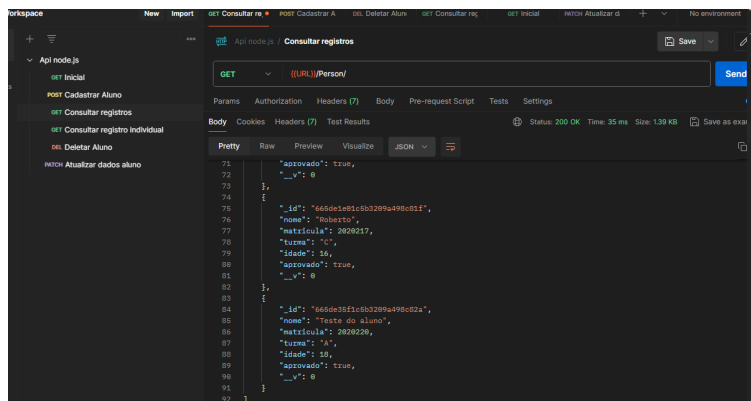
Tela inicial da API



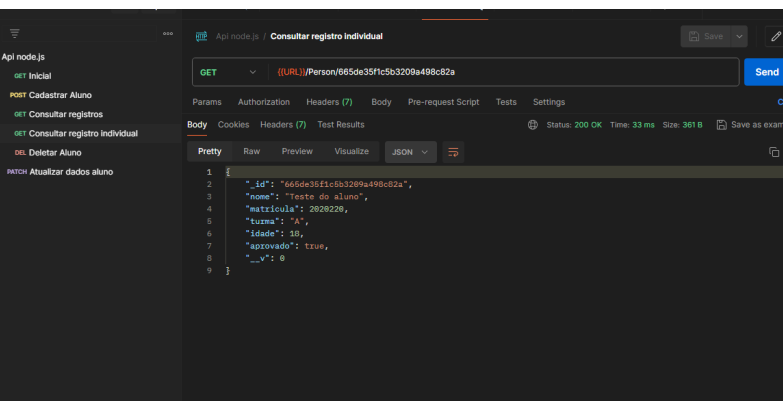
Tela de cadastrar novo aluno



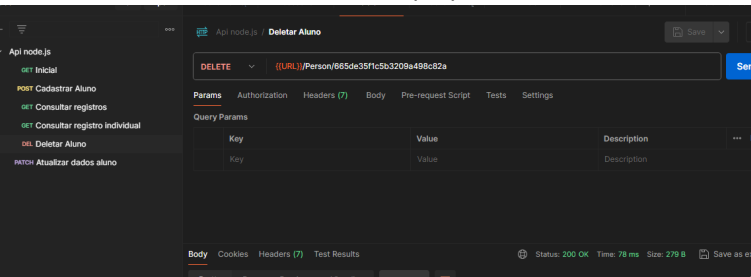
Registros após a atualização



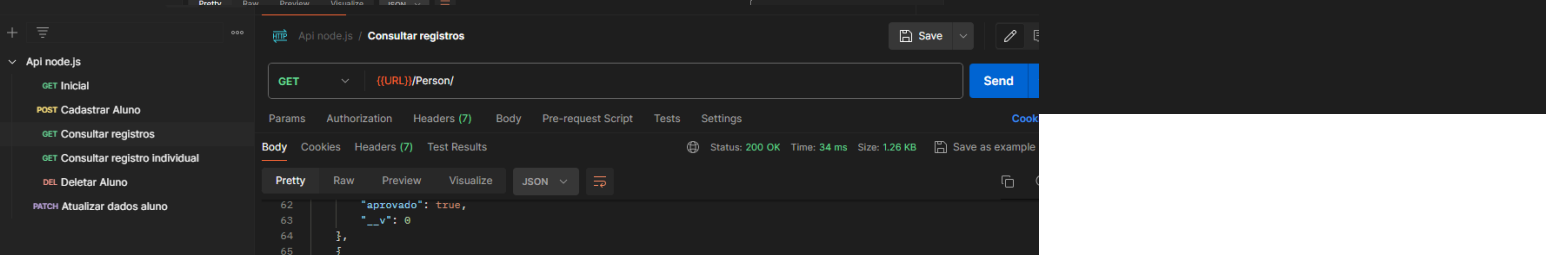
Registro individual (busca por ID)



Remover Aluno(ID)

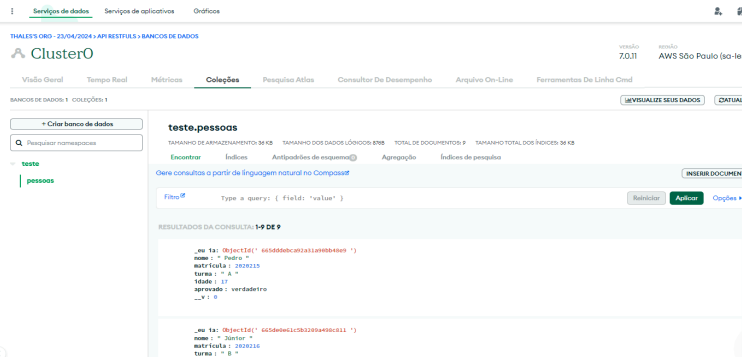
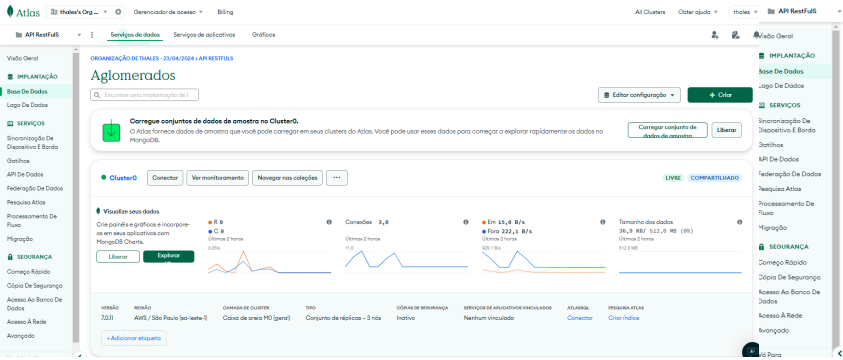


Aluno removido

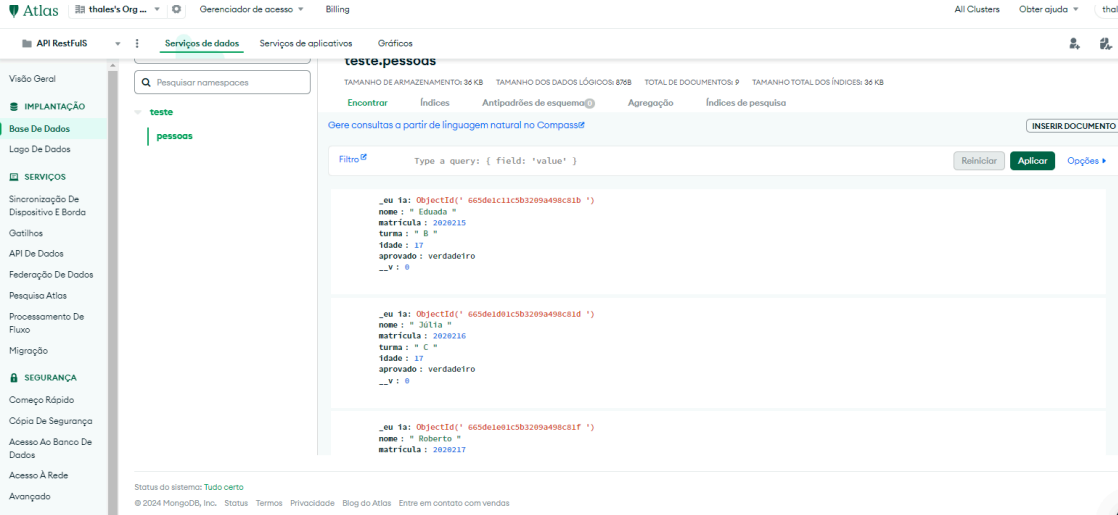


Banco de dados pelo Atlas:

Alunos cadastrados na API



Alunos cadastrados no banco de dados



Dashboards - Atlas



Código:

Person.js

```
const mongoose = require('mongoose')

const Person = mongoose.model('Person', {

  nome: String,

  matricula: Number,

  turma: String,

  idade: Number,

  aprovado: Boolean,

})

module.exports = Person
```

PersonRoutes.js

```
const router = require('express').Router()

const Joi = require('joi');

const Person = require('../models/Person')

//criação de dados

router.post('/', async (req, res) => {

  const {nome, matricula, turma, idade, aprovado} = req.body

  if(!nome) {

    res.status(422).json({ error: 'O nome é obrigatorio!' })

    return

  }
```

```
}

const person = {

  nome,

  matricula,

  turma,

  idade,

  aprovado,

}

try {

const personSchema = Joi.object({

  nome: Joi.string().required(),

  matricula: Joi.number().required(),

  turma: Joi.string().required(),

  idade: Joi.number().required(),

  aprovado: Joi.boolean().required(),

});

function validarDadosPerson(req, res, next) {

  const { error } = personSchema.validate(req.body);

  if (error) {

    return res.status(400).send(error.details[0].message);
```

```
}

    next();

}

// criando dados

await Person.create(person)

res.status(201).json({ Mensagem: 'Aluno adicionado com
sucesso!' })

} catch (error) {

    res.status(500).json({error: error})

}

}))

//leitura de dados

router.get('/', async (req, res) => {

    try {

        const people = await Person.find()

        res.status(200).json(people)

    } catch (error) {
```

```
        res.status(500).json({error: error})

    }

}))

router.get('/:id', async (req, res) => {

    // extrair do dado da requisição, pela url = req.params

    const id = req.params.id

    try {

        const person = await Person.findOne({ _id: id })

        if(!person) {

            res.status(422).json({ Mensagem: 'O usuario não foi encontrado!' })

            return

        }

        res.status(200).json(person)

    } catch (error) {

        res.status(500).json({error: error})

    }

})
```



```
    }  
  })  
  
  // atualização de dados  
  router.patch('/:id', async (req, res) => {  
  
    const id = req.params.id  
  
    const {nome, matricula, turma, idade, aprovado} = req.body  
  
    const person = {  
  
      nome,  
  
      matricula,  
  
      turma,  
  
      idade,  
  
      aprovado,  
  
    }  
  
    try {  
  
      const updatedPerson = await Person.updateOne({ _id: id}, person  
    )  
  
      if(updatedPerson.matchedCount === 0 ){  
  
        res.status(422).json({ Mensagem: 'O usuario não foi  
encontrado!' })
```

```
        return

    }

    res.status(200).json(person)

} catch (error) {

    res.status(500).json({error: error})

}

}))

//deletar dados

router.delete('/:id', async (req, res) => {

    const id = req.params.id

    const person = await Person.findOne({ _id: id })

    if(!person) {

        res.status(422).json({ Mensagem: 'O usuario não foi encontrado!' })

        return

    }

}
```

```
    try {

        await Person.deleteOne({ _id: id })

        res.status(200).json({ Mensagem: 'Usuario removido com sucesso!' })

    } catch (error) {

        res.status(500).json({error: error})

    }

}))

module.exports = router
```

index.js

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

const mongoose = require('mongoose');

const app = express()

// forma de ler JSON / middlewares

app.use(

    express.urlencoded({
```

```
        extended: true,

      )),

    )

app.use(express.json())

// rotas da API

const personRoutes = require('./routes/personRoutes')

app.use('/person', personRoutes)

// rota inicial / endpoint

app.get('/', (req, res) => {

    res.json({ message: 'Olá, seja bem vindo a API!' })

})

// porta

mongoose.connect('mongodb+srv://thalesdias36291:KpexNSaXQn9ut5Yb@cluster0.jar7kbz.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0')

    .then(() => {

        console.log('Conectado ao mongoDB!')
```

```
app.listen(3000)

}))

.catch((err) => console.log(err))
```

Github: <https://github.com/OdiasThales23>

<https://github.com/OdiasThales23/Atividade-avaliativa>

Obs: Depois de várias tentativas não foi possível colocar no repositório o **NODE_MODULES**, por ser um arquivo muito grande o repositório não está aceitando.