INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CEARÁ
Campus Crato

PROJETO DE SISTEMAS WEB 2

Aula 4 – Conexão com Banco de dados

PROFESSOR: HARLEY MACÊDO DE MELLO

Roteiro

- Conceitos
- MongoDB
- Relacionamentos
- Mongoose

Conceitos

- Maior demanda por armazenamento de dados
- Necessidade de escalonamento rápido
- Maior velocidade nas consultas
- Permissão para armazenar dados sem estrutura fixa
- Dados mais próximos do formato JSON

Conceitos

- Banco de dados não relacional
- Sem tabelas e relacionamentos
- Registros independentes
- Estrutura livre, ou ‘schema free’

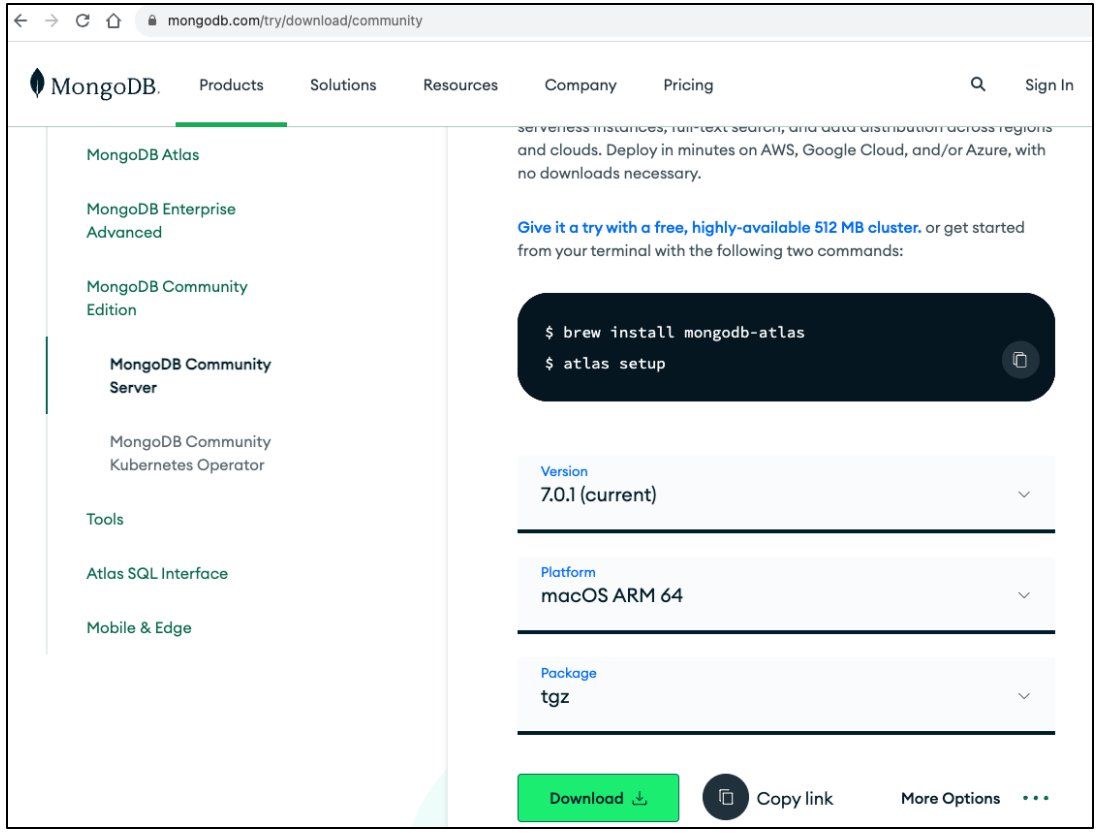
MongoDB

- Banco NoSQL e open source
- Muito utilizado atualmente
- Segundo pesquisa do StackOverflow, usado por mais de 25% do programadores
- Orientado à documentos
- Usa dados em formato JSON
- Software como serviço(SaaS) disponível
- Escalabilidade, flexibilidade, disponibilidade, desempenho

MongoDB

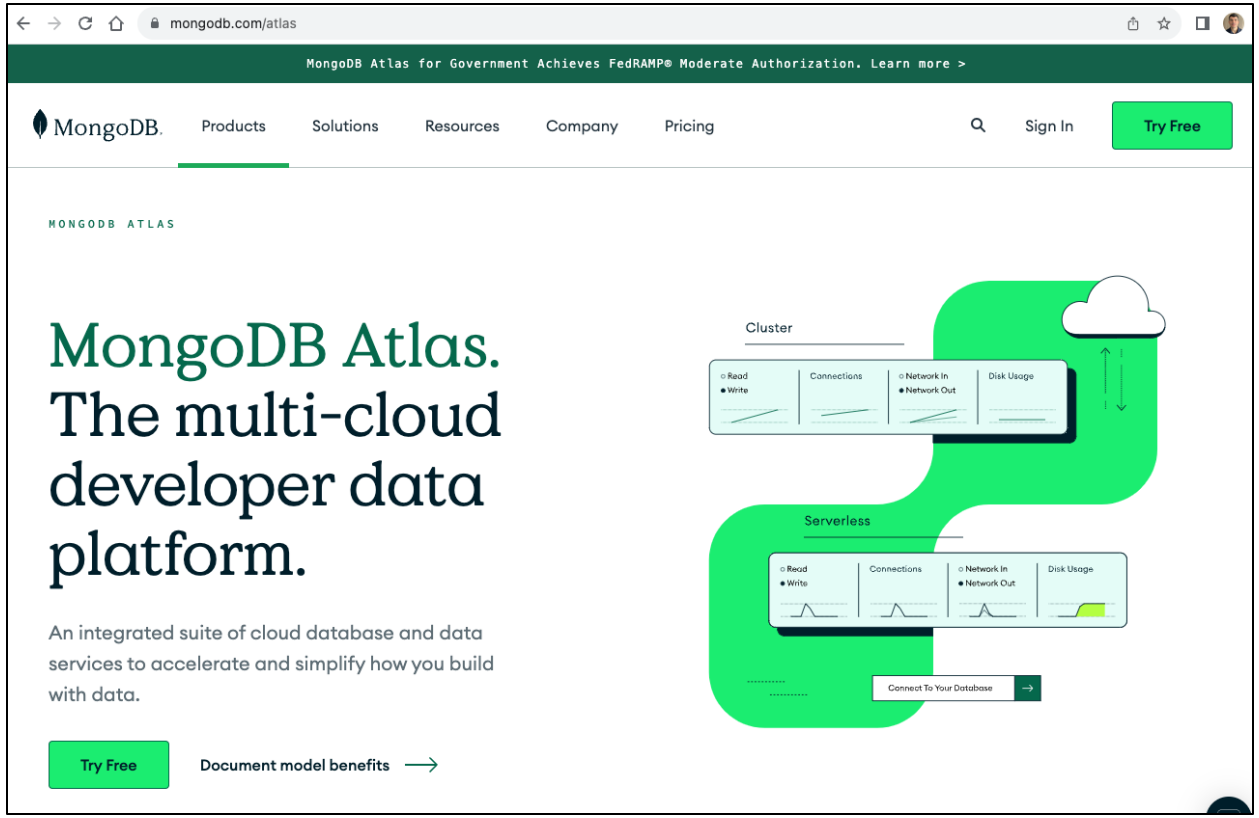
- Download no site mongodb.com/try/download/community
- Usar como serviço em mongodb.com/atlas
- Serviço atlas disponibiliza 500 MB no plano grátis
- Painel que auxilia a construção do DB
- Painel para ajudar na configuração e monitoramento do DB

MongoDB



Tela de download do MongoDB para uso local.

MongoDB



Tela para logar e acessar o serviço atlas MongoDB, para ter um DB em nuvem.

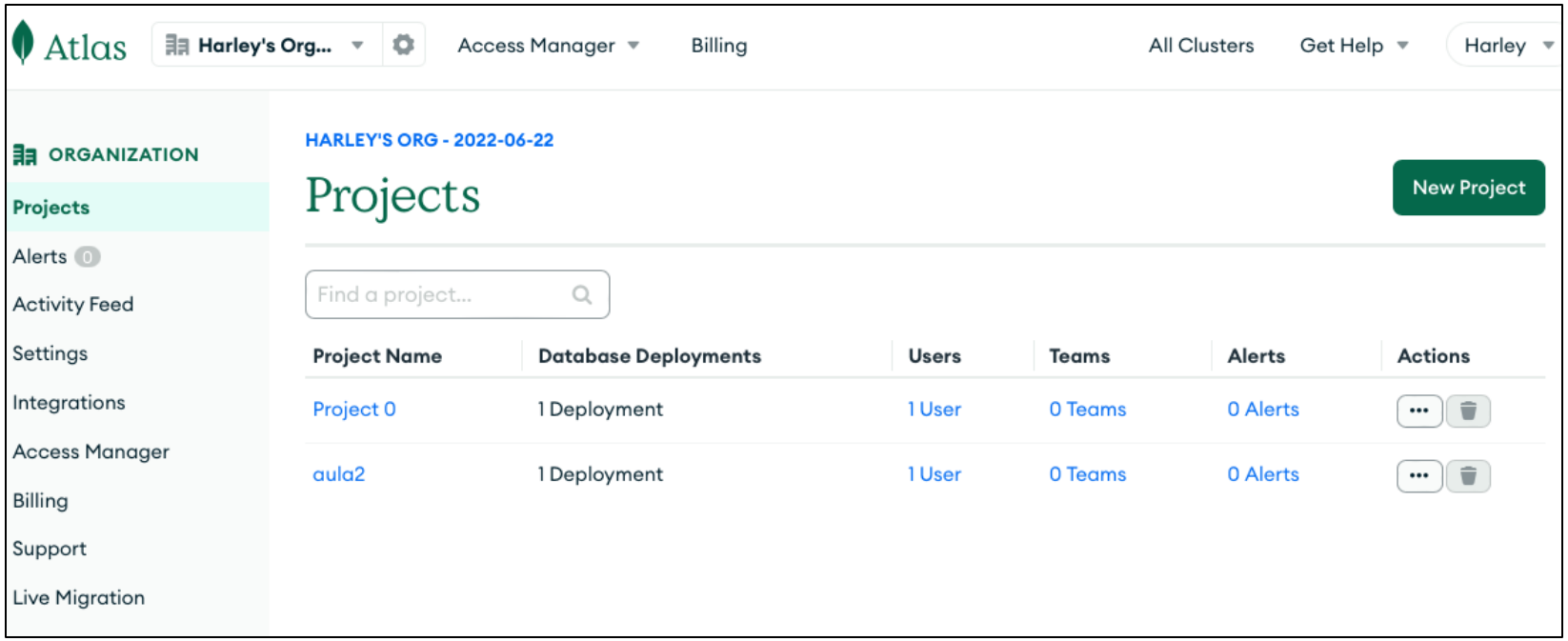
MongoDB

- Organização
  - É a empresa, órgão ou grupo que você está associado
  - Para projetos individuais, é representada pela sua pessoa
- Projeto
  - Uma unidade de trabalho, que pode ter 1 ou mais DBs(Cluster) associados
  - Pode ter 1 time com vários usuários trabalhando no projeto

MongoDB

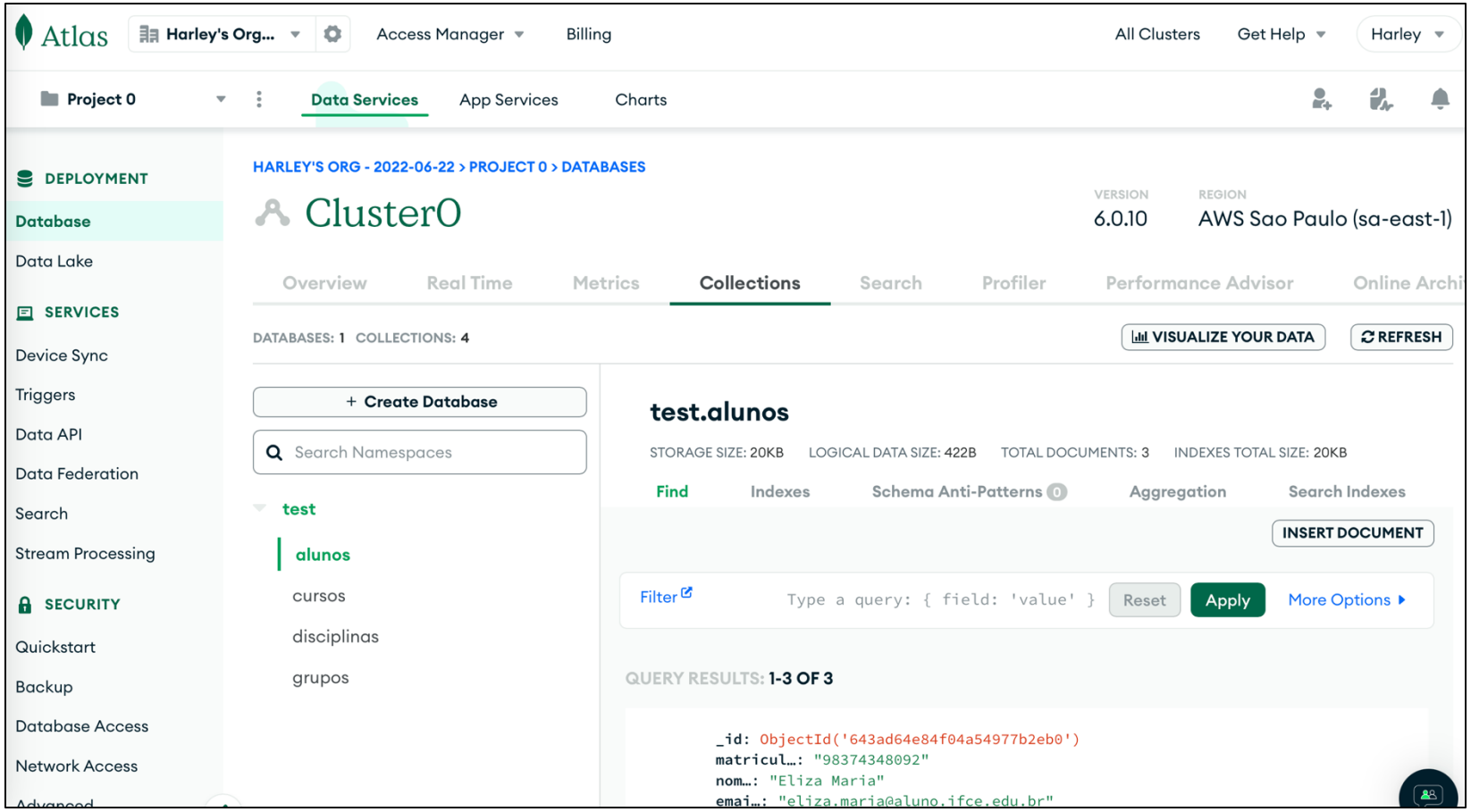
- Cluster
  - Representa um Database, com seus documentos e relacionamentos
  - Podem ter associados usuários específicos, com permissões definidas
- Documentos
  - Representam os registros
  - Semelhantes às linhas em uma tabela

MongoDB



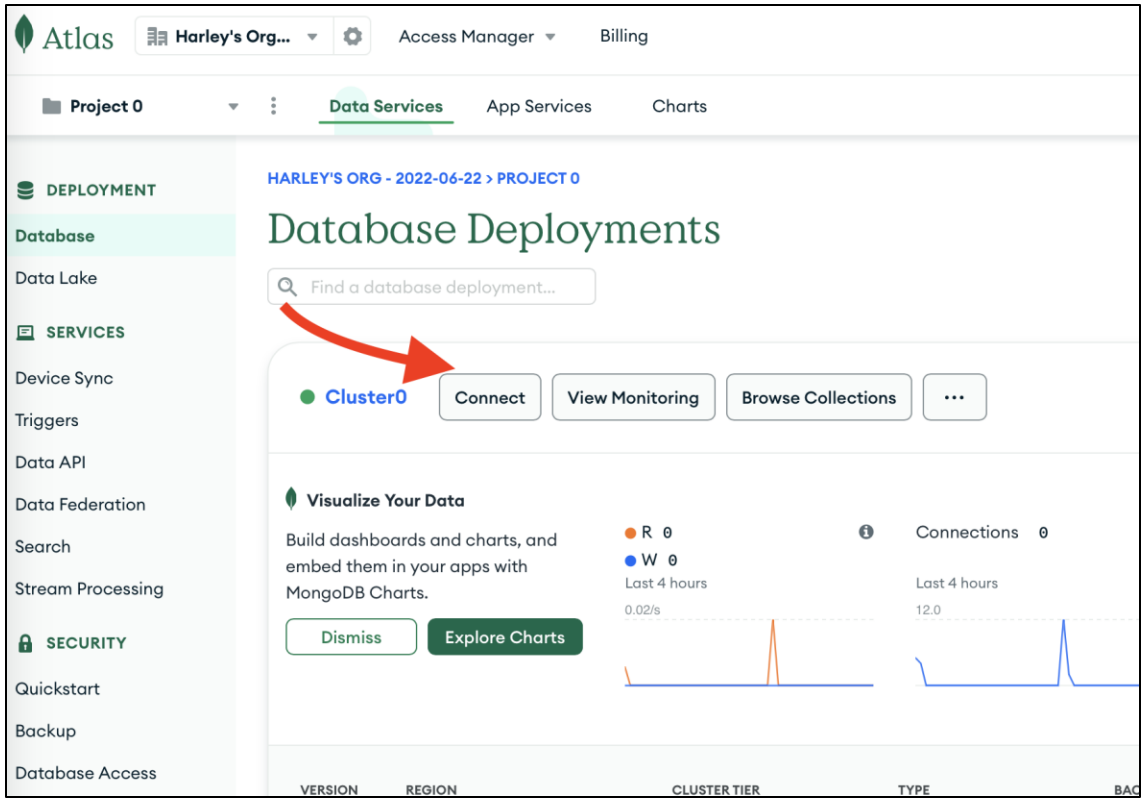
Tela raiz do Atlas MongoDB, na qual podemos visualizar projetos, usuários, times, alertas e etc.

MongoDB



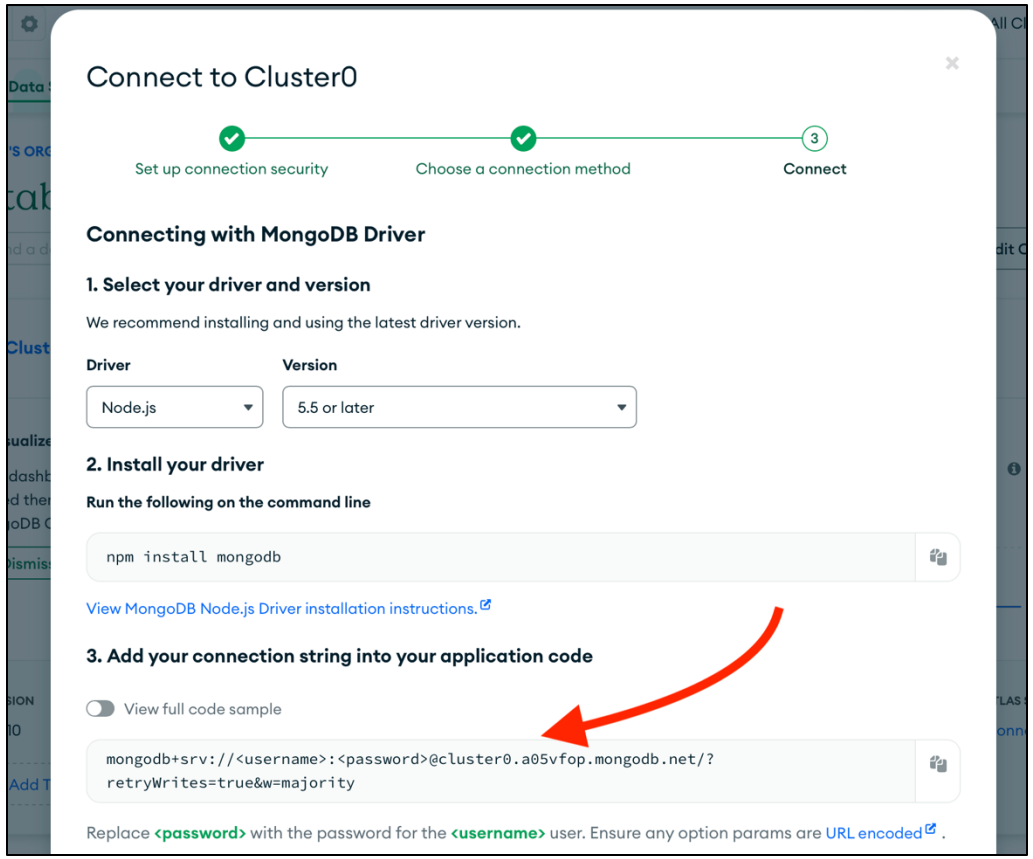
Painel que exibe as collections, como alunos e cursos.

MongoDB



Para obter a String de conexão, clique no botão ‘Connect’, em seguida em ‘Drive’.

MongoDB



Obtendo string de conexão para seu App acessar o DB.

MongoDB

Arquivo .env em seu projeto local, com dados para acessar o MongoDB Atlas.

Relacionamentos

- Modelos mais usados
  - Modelo de documento embutido: Documento pai ter uma coleção de documentos filhos
  - Modelo de referência: Documentos filhos têm uma referência ao documento pai

Relacionamentos

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

const aluno = mongoose.Schema({
    matricula: String,
    nome: String,
    email: String,
    cursoNome: String
})

module.exports = mongoose.model('Aluno', aluno)
```

Esquema aluno com o atributo ‘cursoNome’, que irá relacionar este documento com curso.

Relacionamentos

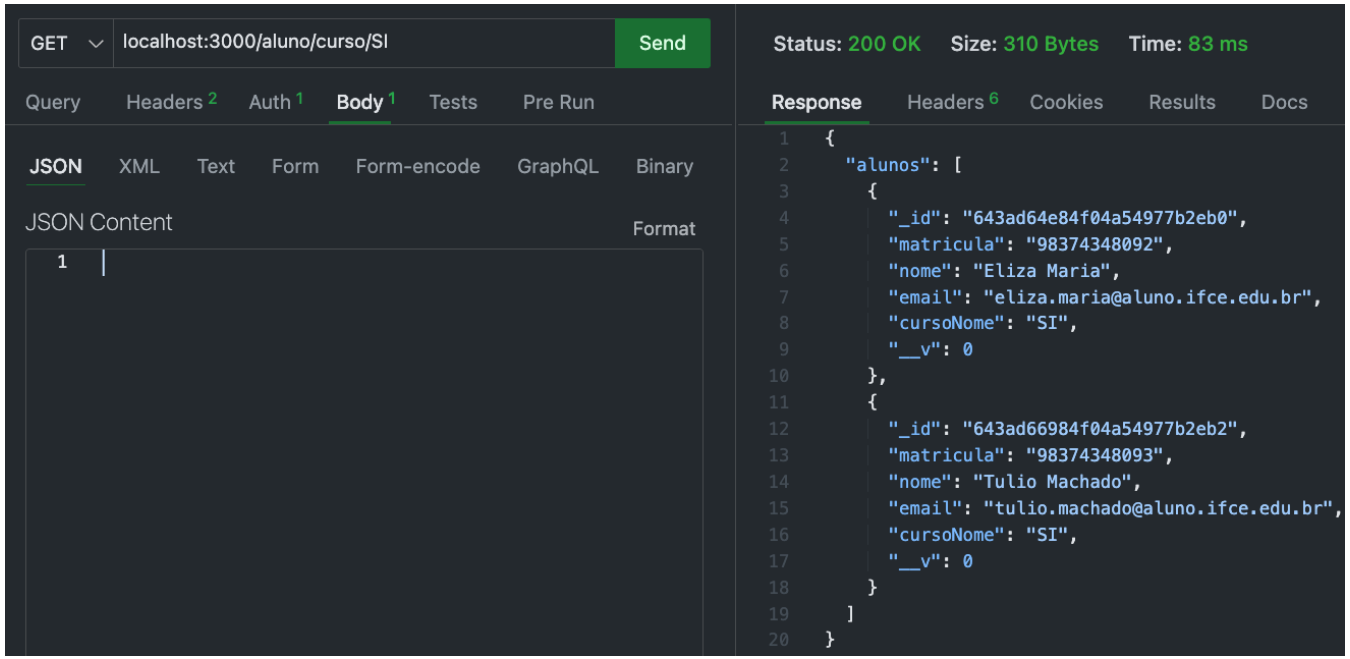
```
const mongoose = require('mongoose')

const curso = mongoose.Schema({
    nome: String,
    tipo: String,
    duracao: Number
})

module.exports = mongoose.model('Curso', curso)
```

Esquema curso, que não faz referência a aluno.

Relacionamentos



Request para recuperar dados de relacionamento, trazendo apenas alunos que pertencem ao curso de SI.

Mongoose

- Modelagem de dados de objeto(ODM)
- Biblioteca que abstrai o uso do MongoDB
- Tradutor de objetos do sistema para objetos no DB
- Auxilia a projeção de esquemas para o sistema

Mongoose

- Função mongoose.Schema({schema})
  - Define uma entidade do sistema
  - O esquema não obriga que todos os objetos tenham a mestra estrutura
- Função mongoose.connect(string_con)
  - Cria uma nova conexão com um cluster
  - Após um período sem uso, a conexão é fechada pelo servidor

Mongoose

- Comandos para manipulação de dados
  - Create
    - Criação de um novo registro(document)
    - Model.create({atributo: valor})
  - Find
    - Busca de registros
    - Model.find({atributo: valor})
    - Os parâmetros servem como filtro da busca

Mongoose

- Comandos para manipulação de dados
  - DeleteOne
    - Exclui um documento da coleção
    - Model.DeleteOne({filtro})
  - FindOneAndUpdate
    - Atualiza um documento da coleção
    - Model.FindOneandUpdate({filtro}, {novosDados})

Mongoose

```
5   //Rota GET para obter todos os alunos
6   alunoRouter.get('/aluno/todos', async (req, res) => {
7       try {
8           await mongoose.connect(process.env.BD_STR_CON)
9           const alunosBuscados = await aluno.find()
10          res.json({alunos: alunosBuscados})
11      } catch (error) {
12          res.json({erro: true, mensagem: 'Erro durante consulta'})
13      }
14  })
15
16  //Rota GET para obter 1 alunos, pela matrícula
17  alunoRouter.get('/aluno/matricula/:matricula', async (req, res) => {
18      try {
19          await mongoose.connect(process.env.BD_STR_CON)
20          const alunoBuscado = await aluno.findOne({matricula: req.params.matricula})
21          res.json({aluno: alunoBuscado})
22      } catch (error) {
23          res.json({erro: true, mensagem: 'Erro durante consulta'})
24      }
25  })
```

Funções find e findOne em uso.

Mongoose

```
57  //Rota PUT para alterar um aluno, pela matrícula
58  alunoRouter.put('/aluno', async (req, res) => {
59      try {
60          await mongoose.connect(process.env.BD_STR_CON)
61          await aluno.findOneAndUpdate(
62              {
63                  "matricula": req.body.matricula
64              },
65              {
66                  "matricula": req.body.matricula,
67                  "nome": req.body.nome,
68                  "email": req.body.email,
69                  "cursoNome": req.body.cursoNome
70              }
71          )
72          res.json({mensagem: 'Aluno atualizado com sucesso'})
73      } catch (error) {
74          res.json({erro: true, mensagem: 'Erro durante consulta'})
75      }
76  })
```

Função put em uso.

Mongoose

```
78  //Rota DELETE para excluir um aluno, pela matrícula
79  alunoRouter.delete('/aluno', async (req, res) => {
80      try {
81          await mongoose.connect(process.env.BD_STR_CON)
82          await aluno.deleteOne({matricula: req.body.matricula})
83          res.json({mensagem: 'Aluno removido com sucesso'})
84      } catch (error) {
85          res.json({erro: true, mensagem: 'Erro durante consulta'})
86      }
87  })
88
89  module.exports = alunoRouter
```

Função delete em uso.