Back-End

- Swagger
- Mongo Express

Swagger

Descrição Técnica do Backend

O backend do projeto é estruturado de forma modular, com separação clara de responsabilidades entre rotas, serviços, middlewares e utilitários. A aplicação é desenvolvida utilizando **Node.js** com **Express** e toda a API está documentada de forma interativa utilizando o **Swagger UI**, acessível pelo endereço:

> “ **http://localhost:3008/api-docs/#/**

Lembre-se de ter o projeto rodando no docker para funcionar!

Estrutura de Diretórios

```
/api/
├── src/
│   ├── docs/              # Configurações e definição dos endpoints no Swagger
│   │   └── swagger.ts
│
│   ├── middlewares/       # ⚙ Middlewares diversos, incluindo o multer para uploads
│   │   └── multer.ts
│
│   ├── router/            # Definição de todas as rotas do sistema organizadas por domínio
│   │   ├── avaliador/
│   │   ├── corregedor/
│   │   ├── defensor/
│   │   ├── usuario/
│   │   └── router.ts      # Roteador principal que importa e agrupa os módulos acima
│
│   ├── services/          # Regras de negócio e funcionalidades para cada tipo de usuário
│   │   ├── avaliador/
```

```
│   │   ├── corregedor/
│   │   ├── defensor/
│   │   └── usuario/
│   │
│   ├── utils/              # 🧰 Utilitários de suporte à aplicação
│   │   ├── opt/            # Envio de OTPs e recuperação de senha
│   │   ├── pdf/            # Operações de manipulação e leitura de PDFs
│   │   ├── validations/    # Validações como token, CPF, e campos diversos
│   │   └── ...             # Outros utilitários adicionais
│   │
│   └── app.tsx             # 🚀 Ponto de entrada da aplicação e inicialização dos serviços
```

Documentação Interativa com Swagger

A API conta com documentação completa utilizando **Swagger**, onde é possível:

- Visualizar **todos os endpoints disponíveis**
- Ver os **parâmetros esperados** por cada rota
- Testar requisições diretamente pela interface
- Visualizar exemplos de **respostas de sucesso e erro**

Exemplo de rotas documentadas:

- `POST /api/defensor/registrar/triagem`
  Registra uma triagem com múltiplos arquivos PDF, processo e protocolo.
- `POST /api/avaliador/pecas/avaliar`
  Permite que um avaliador avalie peças de um defensor.
- `GET /api/usuario/email`
  Retorna as informações do email mascarado.
- `POST /api/corregedor/pecas/aprovar`
  Permite ao corregedor aprovar ou reprovar uma triagem enviada.

Essas e outras rotas estão organizadas por grupos lógicos (Defensor, Avaliador, Corregedor, Usuário) dentro do Swagger UI.

Upload de Arquivos com Multer

A aplicação utiliza o middleware **Multer** para lidar com uploads de arquivos (como PDFs), salvando-os em diretórios específicos com nomes padronizados conforme o campo da requisição.

A lógica para isso está implementada em:

```
src/middlewares/multer.ts
```

Esse middleware identifica dinamicamente o campo do arquivo (`fileRaf`, `fileProcesso_1`, etc.) e direciona o salvamento para a pasta correta, como por exemplo:

- `documentos/raf/`
- `documentos/processos/`
- `documentos/recursos/`
- `documentos/avaliacoes/`
- `documentos/relatorios`

Mongo Express

Interface Mongo Express

Além da API documentada via Swagger, o backend também conta com uma interface web para gerenciamento direto do banco de dados MongoDB utilizando o **Mongo Express** — uma ferramenta leve e funcional para visualização e execução de comandos no banco de forma gráfica.

Acesso à Interface Mongo Express

A interface está disponível no seguinte endereço (ajustável conforme o ambiente de execução):

> “ **http://localhost:8081**

Lembre-se de ter o projeto rodando no docker para funcionar!

As credenciais de acesso ao Express são as variáves: MONGO_INITDB_ROOT_USERNAME e MONGO_INITDB_ROOT_PASSWORD dos arquivos env

Através dela é possível:

- Visualizar todas as coleções disponíveis no banco
- Buscar documentos por filtros (queries MongoDB)
- Editar registros diretamente
- Remover dados manualmente (quando necessário para testes ou administração)
- Inserir novos documentos via interface visual
- Exportar dados para backup ou inspeção

Utilização no Contexto do Projeto

A interface do **Mongo Express** é extremamente útil para:

- **Testes de integração e debug** durante o desenvolvimento
- **Administração rápida de dados**, sem a necessidade de linha de comando
- **Verificação de dados persistidos** via requisições realizadas na API
- **Criação manual de documentos**, útil durante o desenvolvimento de novas rotas

---

Segurança

> ⚠ Em ambiente de produção, o Mongo Express **não deve ser exposto publicamente** sem autenticação e/ou encapsulamento por VPN, proxy ou firewall.
> No projeto atual, ele é **ativado apenas em ambientes de desenvolvimento**, sem acesso externo.

Integração com a API

Durante o desenvolvimento, a interface Mongo Express serve como um complemento ao Swagger:

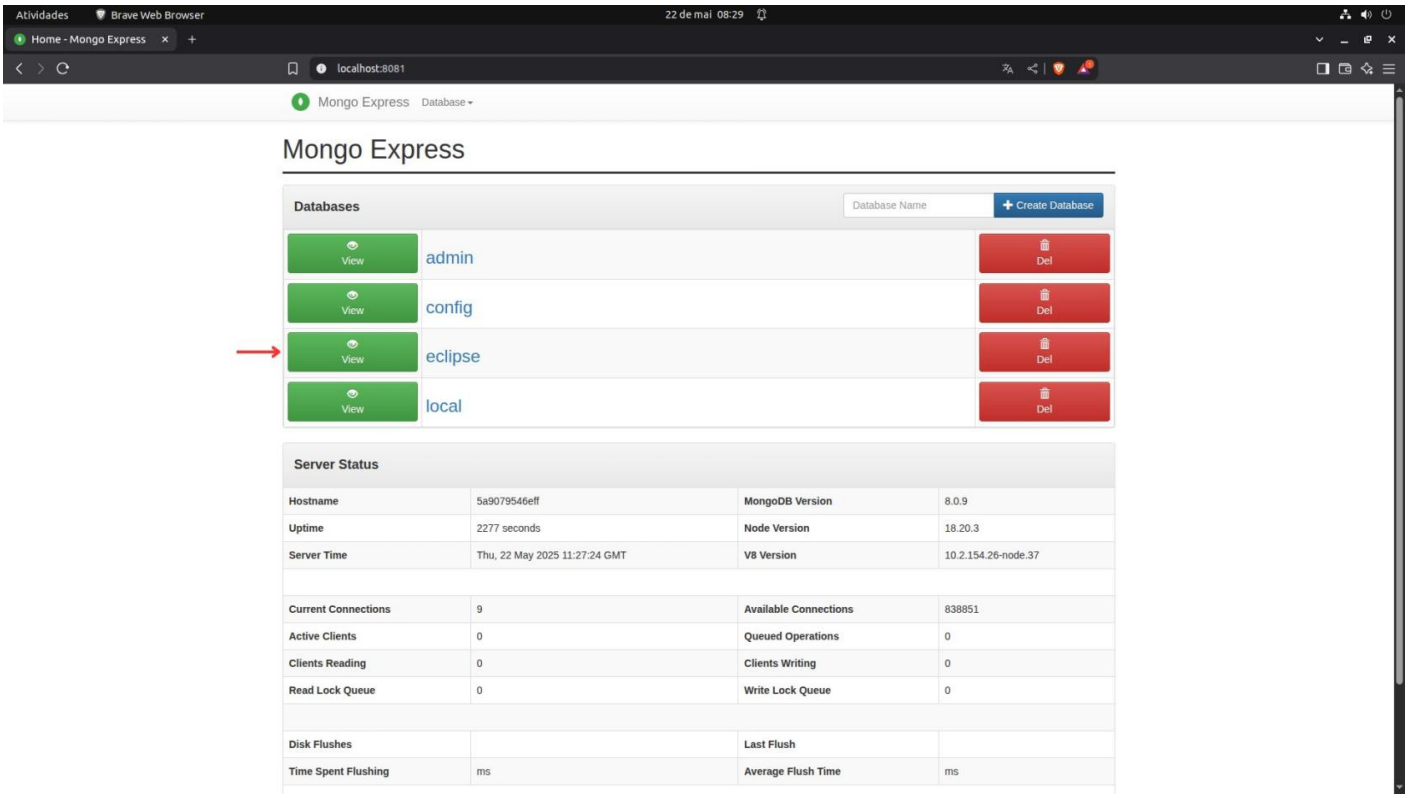
- Você faz uma requisição no **Swagger UI** (como POST `/api/avaliador/pecas/avaliar`)
- E pode **ver o resultado diretamente no Mongo Express**, conferindo como o documento foi persistido em `coleção: Avaliacao`

Exemplos de coleções visualizáveis:

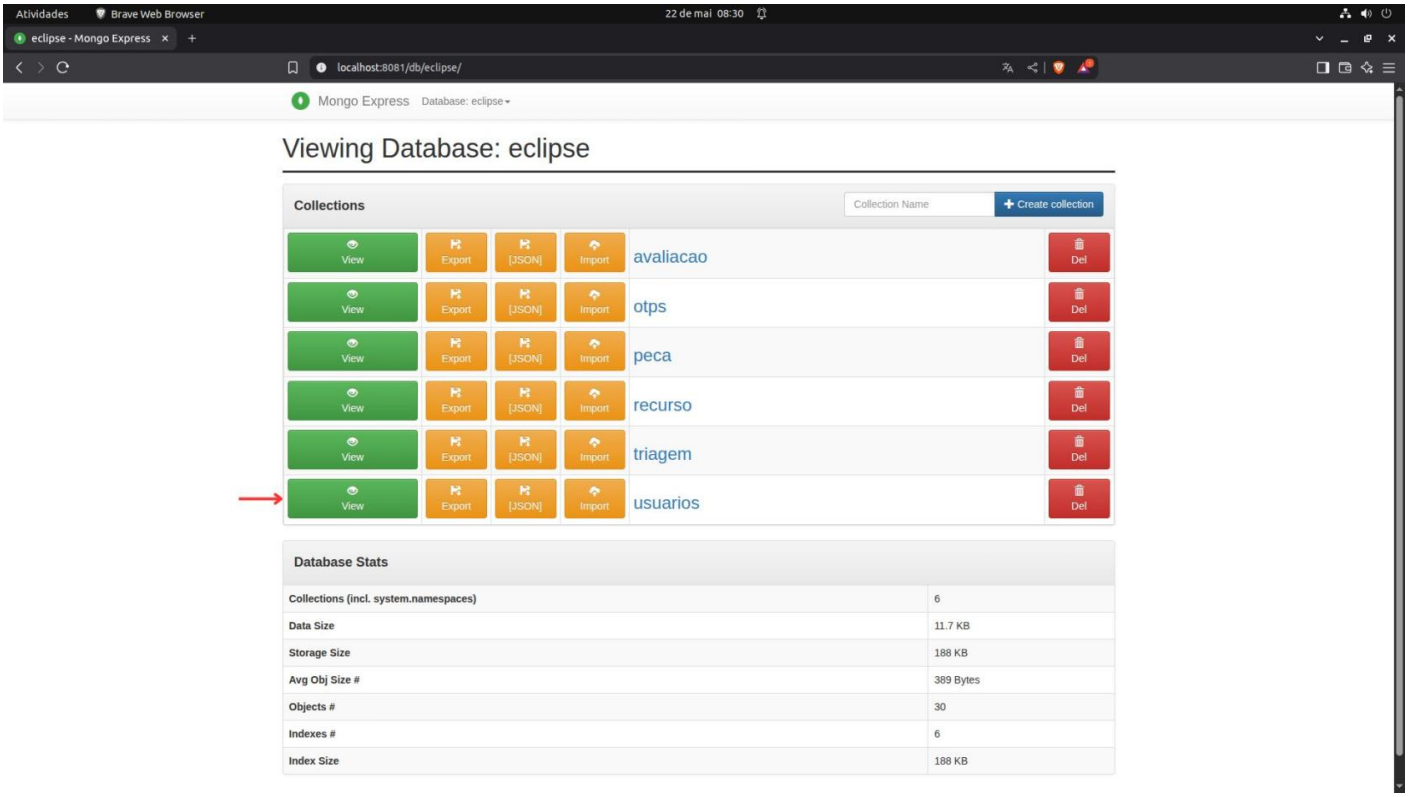
- `Usuarios`
- `Triagem`
- `Peca`
- `Avaliacao`
- `Recurso`
- `Otps`

Passo a Passo de Uso

O mongo express criar algumas tabelas por padrão, mas a que estará sendo usada, é a gerada pelo docker com o nome ***eclipse,*** como indicado na seta vermelha abaixo, basta clicar em ***View*** ou no próprio nome do database:

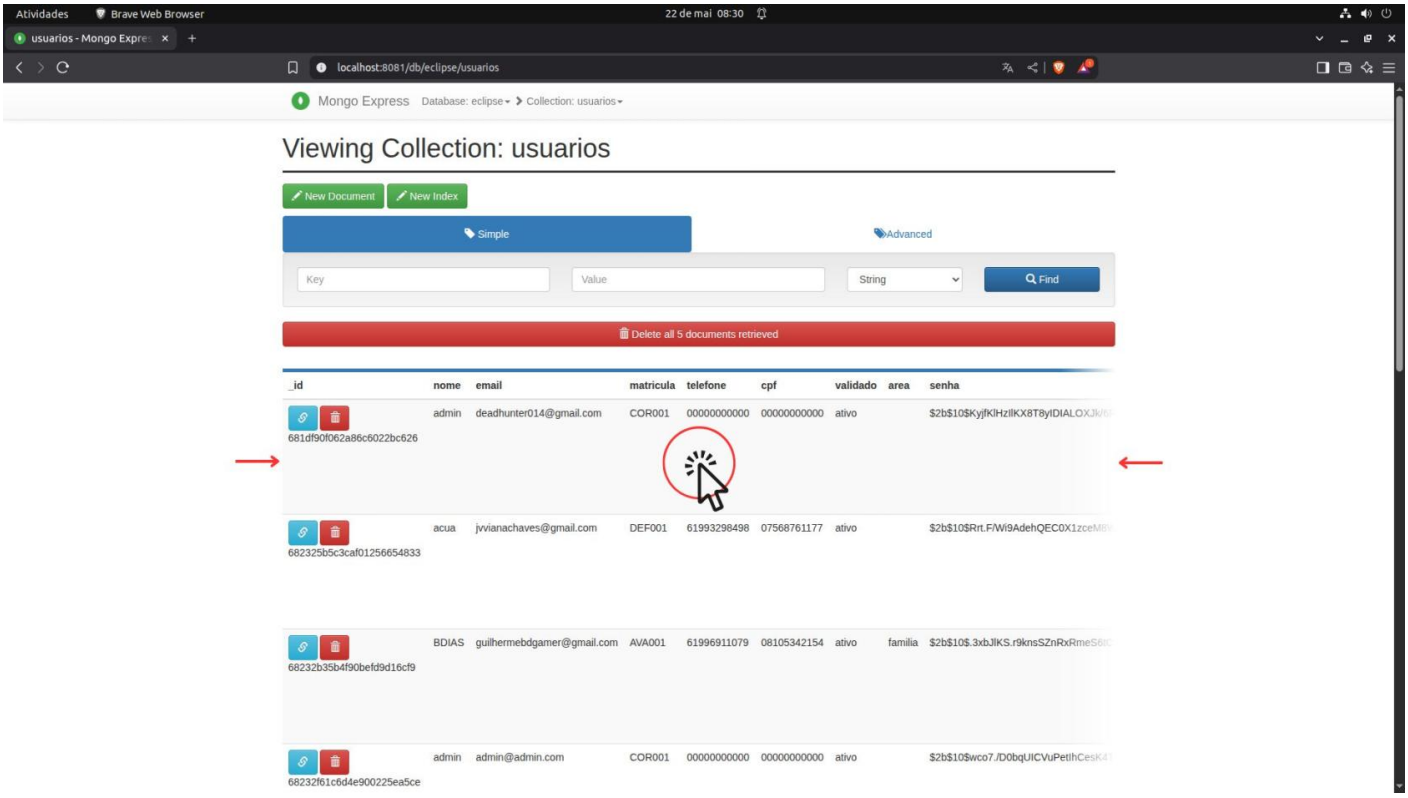


Apos clicar no database **eclipse**, temos acesso as *coleções* (entidades do banco), onde cada respectivo dado se encontra, no exemplo abaixo estaremos mostrando a coleção de ***usuários***, mas você pode repetir o mesmo procedimento para as demais entidades:



Depois de abrir a coleção de ***usuários***, você verá todos os dados cadastrados no banco referentes a essa entidade com todos os campos declarados nas ***Models*** da pasta **api**, como demonstrado

abaixo, você pode clicar em cada um dos dados da coleção para abri-la e/ou edita-la:



Aqui você poderá ver todos os atributos e dados do documento do qual clicou, bem como o nome dos campos e valores. Aqui você pode clicar em qualquer campo e alterar seu valor **(Não se esqueça de clicar em *Save* após as alterações)**, e clique em **Back** para voltar para a sessão de coleções: