**GUIA DE USO API EX ENTERPRISE**

Guia inicial:

* Npm i
* verificar o arquivo .env, alterar a porta do servidor mysql
* Ex: DATABASE\_URL="mysql://root@localhost:3307/ex\_enterprise" (mudar de :3307 para a sua porta)
* Migrar tabela do prisma para o mysql com o comando:  
  npx prisma migrate dev --name init
* Obx: todas as requisições serão separadas por rotas  
   OBS: o token está sendo salvo nos cookies

ROTAS:

Todas as rotas são subsequentes de “/usuario”

* Rota: “/”
* tipo: GET
* função: retornar todos os registros de usuário
* Oq Devo passar?: Nada
* O que me retorna? Json com os usuários encontrados
* Rota: “/login”
* tipo: POST
* função: Verificar se existe esse usuário, validar e gerar token.
* Oq Devo passar?: {email, senha} via body
* O que me retorna? Json com objeto “message” contendo uma mensagem e “token” contendo um token valido após login concluído.
* Rota: “/cadastrar”
* tipo: POST
* função: Cadastrar um novo usuário e gerar token.
* Oq Devo passar?: { nome, sobrenome, email, telefone, senha } via body
* O que me retorna? Json com objeto “message” contendo uma mensagem e “token” contendo um token valido após cadastro concluído.
* Rota: “/:id”
* tipo: PUT
* função: Atualizar um usuário se o token for valido.
* Oq Devo passar?: { “dado a ser atualizado”} via body
* O que me retorna? Json com objeto “message” contendo uma mensagem.
* Rota: “/:id”
* tipo: DELETE
* função: Deletar um usuário se o token for valido.
* Oq Devo passar?: { “dado a ser atualizado”} via body
* O que me retorna? Json com objeto “message” contendo uma mensagem.

ROTA: “/verificarToken”

* Rota: “/”
* tipo: GET
* função: Validar do front se o token armazenado nos cookies é válido
* Oq Devo passar?: nada.
* O que me retorna? Json com objeto “message” contendo uma mensagem “Token valido” ou “Token inválido ou expirado”.

Obs: Todas as rotas com ressalva do Login e cadastro irá precisa do token armazenado em cookies

ROTA: “/pdf”

* Rota: “/gerar”
* tipo: POST
* função: Gerar o contrato preenchido e em PDF
* Oq Devo passar?: nomeLocador, estadoCivilLocador, profissaoLocador, rgLocador, cpfLocador, locadouroLocador, logradouroNumeroLocador, bairroLocador, cidadeLocador, cepLocador, nomeLocatario, estadoCivilLocatario, profissaoLocatario, rgLocatario, cpfLocatario, logradouroImovel, numeroImovel, complementoImovel, cidadeImovel, cepImovel, prazoLocacao, valorMulta, valorMultaExtenso, valorLocacao, valorLocacaoExtenso, diaPagamentoLocacao, porcentagemMulta, porcentagemMultaExtenso, jurosMora, jurosMoraExtenso, limiteDiasAtrasoPagamento, prazoServicos, serviços
* O que me retorna? Um arquivo PDF preenchido. Também irá salvar na pasta PUBLIC do back-end

ROTA: “/imoveis”

* Rota: “/listar”
* tipo: GET
* função: mostrar todos os imóveis cadastrados
* Oq Devo passar?: nada, porem o token precisa ser válido
* O que me retorna? Um json contendo todos os imóveis cadastrados
* Rota: “/listar/:id”
* tipo: POST
* função: mostrar somente 1 imóvel
* Oq Devo passar?: passar o id pelo parâmetro
* O que me retorna? Um json contendo um imovel
* Rota: “/cadastrar”
* tipo: POST
* função: cadastrar um novo imovel
* Oq Devo passar?: nome, endereco, disponibilidade, valorAlocacao, valorCondominio, valorIPTU, os outros dados não são obrigatórios no cadastro
* O que me retorna? Um json “message” contendo a mensagem Imovel adicionado com sucesso
* Rota: “/:id”
* tipo: PUT
* função: Atualizar os dados de um imovel
* Oq Devo passar?: nome, endereco, disponibilidade, valorAlocacao, valorCondominio, valorIPTU, os outros dados não são obrigatórios.
* O que me retorna? Um json “message” contendo a mensagem Imovel atualziado com sucesso
* Rota: “/:id”
* tipo: DELETE
* função: Deletar os um imovel do sistemas
* Oq Devo passar?: id por aprametro
* O que me retorna? Um json “message” contendo a mensagem Imóvel deletado com sucesso