DOCUMENTAÇÃO BACKEND

Introdução

Esta documentação descreve a estrutura e a implementação do backend do aplicativo Treinar.AI, que foi desenvolvido usando Python e Flask, com um banco de dados MongoDB e hospedado na Vercel.

Tecnologias Utilizadas

Backend: Python com Flask

Banco de Dados: MongoDB

Hospedagem: Vercel

Bibliotecas: pymongo, flask

Estrutura do Projeto

O projeto está organizado da seguinte forma:

treinarAI-backend/

├── index.py

├── vercel.json

└── .idea/

  └── modules.xml

Configuração do Banco de Dados

O backend se conecta a um banco de dados MongoDB hospedado na nuvem através do MongoDB Atlas.

URI de Conexão:

"mongodb+srv://benicio:ebS3RM8Vew0hPt8y@cluster0.5g6o5p0.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0"

Configuração do Cliente MongoDB:

from pymongo.mongo_client import MongoClient

from pymongo.server_api import ServerApi

client = MongoClient(uri, server_api=ServerApi('1'))

db = client["db"]

col_users = db["users"]

Estrutura do Aplicativo Flask:

O aplicativo Flask é inicializado no arquivo index.py e define várias rotas para manipulação de usuários.

Inicialização do Flask:

from flask import Flask, request, jsonify

app = Flask(__name__)

Rota Inicial:

@app.route("/")

def index():

  return jsonify({"hello": "word"})

Rotas e Funcionalidades

1 - Login de Usuário

• Endpoint: /logar/user

• Método: POST

• Descrição: Autentica um usuário com base no email e senha fornecidos.

Exemplo de Requisição:

{

  "email": "user@example.com",

  "senha": "password123"

}

Respostas:

200: {"resp": "Bem-vindo de volta!"}

400: {"resp": "Email ou Senha Inválidos!"}

2 - Criação de Usuário

- Endpoint: /criar/user
- Método: POST
- Descrição: Cria um novo usuário com os dados fornecidos.

Exemplo de Requisição:

```
{
   "nomeCompleto": "Nome Completo",
   "username": "username",
   "email": "user@example.com",
   "senha": "password123"
}
```

Respostas:

200: {"resp": "Conta Criada com Sucesso!"}

3 - Atualização de Usuário

- Endpoint: /atualizar/user
- Método: PUT
- Descrição: Atualiza a senha de um usuário existente.

Exemplo de Requisição:

```
{
   "email": "user@example.com",
   "novaSenha": "newpassword123"
}
```

Respostas:

200: {"resp": "Senha Atualizada com Sucesso!"}

400: {"resp": "Email não Cadastrado no Banco de Dados!"}

Configuração da Hospedagem na Vercel

O arquivo vercel.json define como o backend deve ser implantado na Vercel.

Configuração de Build:

```
{
  "version": 2,
  "builds": [
    {
      "src": "./index.py",
      "use": "@vercel/python"
    }
  ],
  "routes": [
    {
      "src": "/(.*)",
      "dest": "/"
    }
  ]
}
```

Configuração do Ambiente no Android Studio

O arquivo modules.xml define a configuração do ambiente Python no Android Studio.

Configuração do Módulo:

```
<?xml version="1.0" encoding="UTF-8"?>
<module type="PYTHON_MODULE" version="4">
  <component name="NewModuleRootManager">
    <content url="file://$MODULE_DIR$">
      <excludeFolder url="file://$MODULE_DIR$/venv" />
    </content>
    <orderEntry type="jdk" jdkName="Python 3.10 (treinarAiAPI)" jdkType="Python SDK" />
    <orderEntry type="sourceFolder" forTests="false" />
  </component>
</module>
```

Conclusão

Esta documentação cobre a configuração e implementação do backend do aplicativo Treinar.AI. Certifique-se de proteger suas credenciais e de testar as rotas para garantir que tudo funcione conforme esperado. Para mais detalhes sobre a configuração e implantação, consulte a documentação oficial do Flask e da Vercel.