

HealthConsultationAPI

Introdução

O **HealthConsultationAPI** é uma aplicação de serviço RESTful que oferece recursos para marcação de consultas de saúde e acesso ao histórico do paciente. Este relatório apresenta uma visão geral da API, detalhes da implementação, tecnologias utilizadas e os objetivos alcançados durante o desenvolvimento.

Objetivo

O objetivo do **HealthConsultationAPI** é fornecer uma plataforma segura e eficiente para marcação de consultas médicas e acesso ao histórico dos pacientes. A API permite que os usuários realizem operações CRUD (Criar, Ler, Atualizar e Apagar) sobre os dados, disponibilizando representações de estado dos recursos em formato JSON. Além disso, a API é protegida por uma camada de autenticação e autorização, garantindo que apenas usuários autorizados possam acessar os recursos.

Tecnologias Utilizadas

- **Node.js**: Utilizado como servidor aplicacional para implementação da camada de serviços.
 - **Express.js**: Framework utilizado para implementação da camada de serviços RESTful.
 - **MySQL**: Utilizado como SGBD para armazenamento dos dados.
 - **OAuth 2.0**: Utilizado para implementação da camada de autenticação e autorização da google.
 - **Docker**: Utilizado para criação de containers para a aplicação multi-container (MySQL + Node.js).
 - **React**: Biblioteca JavaScript utilizada para a construção da interface do usuário (UI).
- Pode fazer o pull das nossas imagens em:

1. Imagem do mysql

```
docker pull inf2023dw2g25/healthapi:healthapi-mysql
```

2. Imagem do node

```
docker pull inf2023dw2g25/healthapi:healthapi-node
```

- **OpenAPI 3.0**: Utilizado para documentação da API.

Recursos Disponíveis

A API disponibiliza os seguintes recursos em localhost:3030

1. **Consulta**: Permite a marcação, visualização, atualização e exclusão de consultas médicas que tem uma relacao de muitos para um com especialista e paciente .
- Obter todas

- Criar
- Editar
- Apagar
- Obter Especifica

2. **Paciente:** Permite a visualização do histórico médico de um paciente com uma relação de 1 para 1.

- Obter todos
- Criar
- Editar
- Apagar (Este método também apagará todas as consultas e histórico do paciente)
- Obter Especifico
- Obter Consultas do Paciente
- Obter Historico do Paciente

3. **Especialista:** Permite a visualização dos especialistas disponíveis.

- Obter todos
- Criar
- Editar
- Apagar (Este método também apagará todas as consultas do especialista)
- Obter Especifico
- Obter Consultas do Especialista

4. **Especialidade:** Permite a visualização das especialidades médicas disponíveis relação de 1 para muitos com o especialista.

- Obter todas
- Criar
- Editar
- Apagar (Este método também apagará todos os especialistas pertencentes a esta especialidade)
- Obter Especialidade Especifica
- Obter Especialistas por especialidade.

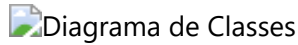
5. **Historico:** Permite a visualização do Histórico do paciente relação de um para um com o paciente o histórico é como se fosse um livro com a ficha do paciente.

- Obter todos
- Criar
- Editar
- Apagar
- Obter Especifico

Relações

- **Historico** pode ter um **Paciente** e um **Paciente** pode ter um **Historico**. 1 <-> 1
- **Paciente** pode ter várias **Consultas** e uma **Consulta** só pode ter um **Paciente**. N <-> 1 e 1 <-> 1
- **Especialista** pode ter várias **Consultas** e uma **Consulta** só pode ter um **Especialista**. N <-> 1 e 1 <-> 1

- **Especialidade** pode ter várias **Especialista** e uma **Especialista** só pode ter um **Especialidade**. N <-> 1 e 1 <-> 1



Autenticação e Autorização

A API implementa uma camada de autenticação e autorização utilizando OAuth 2.0 da google. Os usuários devem autenticar-se para acessar aos metodos (POST, PUT , DELETE) dos recursos da API , sem a autenticação só estão autorizados a ver mos metodos (GET). Além disso, o React implementa uma camada extra de autenticação no lado do cliente. Isso garante que apenas usuários autenticados possam acessar e interagir com determinadas partes da interface do usuário, proporcionando uma camada adicional de segurança e controle sobre o acesso aos recursos da aplicação.

Documentação da API

A API é documentada utilizando o formato OpenAPI 3.0. A documentação detalha todos os endpoints disponíveis, os métodos HTTP suportados, os parâmetros necessários e as respostas esperadas. na pasta docs.

Repositórios

- **GitHub:** [HealthAPI](#)
- **GitHub:** [HealthAPP](#)
- **GitHub:** [HEALTHAPPLICATION](#)
- **Docker Hub:** [HealthAPI](#)

Instruções

1. faça o clone do repositório

```
git clone https://github.com/inf23dw2g25/HEALTHAPPLICATION
```

2. entre na pasta clonada

```
cd HEALTHAPPLICATION
```

3. entre na pasta da API

```
cd HealthAPI
```

4. faça o clone do repositório

```
git clone https://github.com/inf23dw2g25/HealthAPI .
```

4. volte a root

```
cd ..
```

5. entra na pasta da APP

```
cd healthapp
```

6. faça o clone do repositório da healthapp

```
git clone https://github.com/inf23dw2g25/healthapp .
```

7. volte a root

```
cd ..
```

8. Para este passo o docker precisa de estar a correr .

```
docker compose up --build
```

Selecione os quadrados profile e email para conseguirem-se authenticar.

MARKDOWN PDF



Implementações extras e erros difíceis:

1. Encontramos este erro sempre que tentamos obter o perfil do google:

```
Erro ao obter o perfil do usuário: TypeError: Cannot read properties of undefined
(reading 'accessToken')
node_healthapi |       at Object.getPerfil (/app/services/perfilService.js:15:34)
node_healthapi |       at getProfile (/app/controllers/perfilController.js:6:43)
node_healthapi |       at Layer.handle [as handle_request]
(/app/node_modules/express/lib/router/layer.js:95:5)
node_healthapi |       at next
(/app/node_modules/express/lib/router/route.js:149:13)
node_healthapi |       at Route.dispatch
(/app/node_modules/express/lib/router/route.js:119:3)
node_healthapi |       at Layer.handle [as handle_request]
```

```
(/app/node_modules/express/lib/router/layer.js:95:5)
node_healthapi | at /app/node_modules/express/lib/router/index.js:284:15
node_healthapi | at Function.process_params
(/app/node_modules/express/lib/router/index.js:346:12)
node_healthapi | at next
(/app/node_modules/express/lib/router/index.js:280:10)
node_healthapi | at Function.handle
(/app/node_modules/express/lib/router/index.js:175:3)
node_healthapi | perfil Controller: undefined
```


Erro corrigido tivemos de optar por fazer uma modificação no middleware para usar o token Bearer após autenticação. Queríamos implementar uma parte de recursos para o profile do user, porém sem sucesso a obter do google api

2. Encontramos Varios erros de compatibilidades ao longo do projecto, porém o unico que não conseguimos resolver foi :

```
[Error] Warning: y: Support for defaultProps will be removed from function
components in a future major release. Use JavaScript default parameters instead.
y@
http://localhost:3000/static/js/vendors~main.chunk.js:43935:35
div
Login@
http://localhost:3000/static/js/main.chunk.js:2982:63
div
div
TopBar@
http://localhost:3000/static/js/main.chunk.js:2740:63
div
Router@
http://localhost:3000/static/js/vendors~main.chunk.js:47800:68
BrowserRouter@
http://localhost:3000/static/js/vendors~main.chunk.js:45743:54
AuthProvider@
http://localhost:3000/static/js/main.chunk.js:566:5
App@
http://localhost:3000/static/js/main.chunk.js:205:58
  printWarning (vendors~main.chunk.js:16143)
  error (vendors~main.chunk.js:16120)
  validateFunctionComponentInDev (vendors~main.chunk.js:33609)
  mountIndeterminateComponent (vendors~main.chunk.js:33578)
  beginWork$1 (vendors~main.chunk.js:39770)
  performUnitOfWork (vendors~main.chunk.js:39040)
  workLoopSync (vendors~main.chunk.js:38963)
  renderRootSync (vendors~main.chunk.js:38936)
  performConcurrentWorkOnRoot (vendors~main.chunk.js:38331:93)
  workLoop (vendors~main.chunk.js:52855)
  flushWork (vendors~main.chunk.js:52833)
  performWorkUntilDeadline (vendors~main.chunk.js:53070)
  run (vendors~main.chunk.js:53272)
```

```
runIfPresent (vendors~main.chunk.js:53300)
onGlobalMessage (vendors~main.chunk.js:53337)
```

POSTMAN COLECTION

 Readme PDF

```
{
  "info": {
    "_postman_id": "d9a4549d-5e56-4562-b4cb-179af336a9a7",
    "name": "HealthApi",
    "schema": "https://schema.getpostman.com/json/collection/v2.1.0/collection.json",
    "_exporter_id": "30230440"
  },
  "item": [
    {
      "name": "Get Consultas",
      "request": {
        "method": "GET",
        "header": [],
        "url": {
          "raw": "localhost:3000/consultas",
          "host": [
            "localhost"
          ],
          "port": "3000",
          "path": [
            "consultas"
          ]
        }
      },
      "response": []
    },
    {
      "name": "Get consulta Especifica",
      "request": {
        "method": "GET",
        "header": [],
        "url": {
          "raw": "localhost:3000/consultas/7",
          "host": [
            "localhost"
          ],
          "port": "3000",
          "path": [
            "consultas",
            "7"
          ]
        }
      }
    }
  ]
}
```

```
    "response": []
  },
  {
    "name": "Get Perfil",
    "request": {
      "method": "GET",
      "header": [],
      "url": {
        "raw": "localhost:3000/perfil",
        "host": [
          "localhost"
        ],
        "port": "3000",
        "path": [
          "perfil"
        ]
      }
    },
    "response": []
  },
  {
    "name": "Post Consulta",
    "request": {
      "method": "POST",
      "header": [],
      "body": {
        "mode": "raw",
        "raw": "{\n  \"data_e_hora\": \"2024-05-09 01:43\",\n  \"paciente_id\": 3,\n  \"especialista_id\": 4,\n  \"observacoes\": \"Ola isto e um teste\\n\\n\"}"
      },
      "options": {
        "raw": {
          "language": "json"
        }
      }
    },
    "url": {
      "raw": "localhost:3000/consultas",
      "host": [
        "localhost"
      ],
      "port": "3000",
      "path": [
        "consultas"
      ]
    }
  },
  "response": []
},
{
  "name": "Delete Consulta",
  "request": {
    "method": "DELETE",
    "header": [],
```

```
        "url": {
          "raw": "localhost:3000/consultas/61",
          "host": [
            "localhost"
          ],
          "port": "3000",
          "path": [
            "consultas",
            "61"
          ]
        },
      },
      "response": []
    },
    {
      "name": "Edit Consulta",
      "request": {
        "method": "PUT",
        "header": [],
        "url": {
          "raw": "localhost:3000/consultas/1",
          "host": [
            "localhost"
          ],
          "port": "3000",
          "path": [
            "consultas",
            "1"
          ]
        }
      },
      "response": []
    }
  ],
  "auth": {
    "type": "oauth2",
    "oauth2": [
      {
        "key": "client_authentication",
        "value": "header",
        "type": "string"
      },
      {
        "key": "scope",
        "value": "https://www.googleapis.com/auth/userinfo.profile",
        "type": "string"
      },
      {
        "key": "tokenName",
        "value": "HealthApi token",
        "type": "string"
      },
      {
        "key": "accessTokenUrl",
```



```

        "value": "https://accounts.google.com/o/oauth2/token",
        "type": "string"
    },
    {
        "key": "authUrl",
        "value": "https://accounts.google.com/o/oauth2/auth",
        "type": "string"
    },
    {
        "key": "useBrowser",
        "value": true,
        "type": "boolean"
    },
    {
        "key": "redirect_uri",
        "value": "http://localhost:3000/google/callback",
        "type": "string"
    },
    {
        "key": "grant_type",
        "value": "authorization_code",
        "type": "string"
    },
    {
        "key": "clientSecret",
        "value": "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",
        "type": "string"
    },
    {
        "key": "clientId",
        "value": "860907029773-
f1k556igmjvjdm8lpg0cja24o1m2mhnf.apps.googleusercontent.com",
        "type": "string"
    },
    {
        "key": "addTokenTo",
        "value": "header",
        "type": "string"
    }
]
},
"event": [
    {
        "listen": "prerequest",
        "script": {
            "type": "text/javascript",
            "packages": {},
            "exec": [
                ""
            ]
        }
    },
    {
        "listen": "test",

```

```
    "script": {
      "type": "text/javascript",
      "packages": {},
      "exec": [
        ""
      ]
    }
  ]
}
```