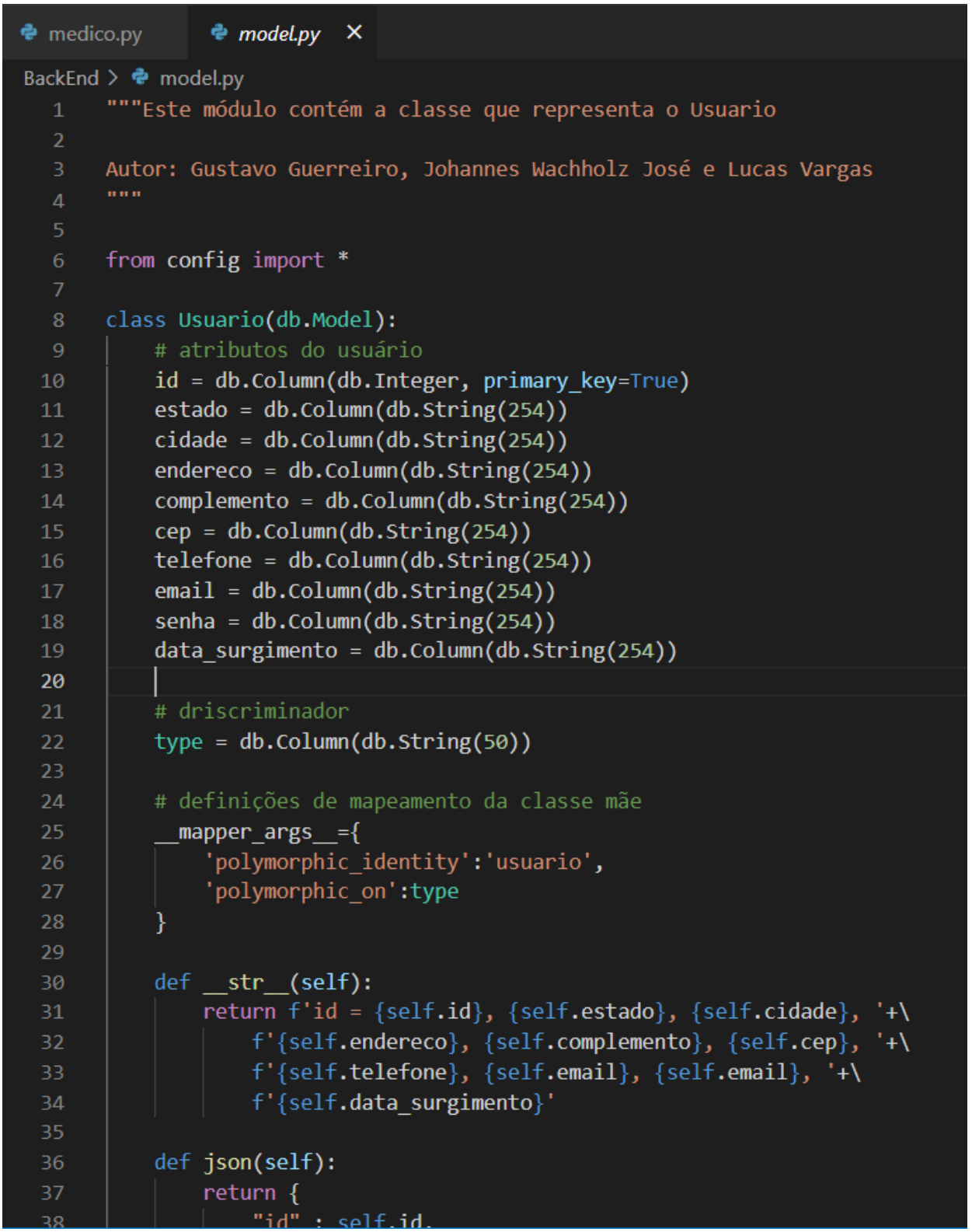
Aluno: Gustavo Guerreiro
Turma: 302

Print da definição classe do trabalho:



```python
def json(self):
    return {
        "id" : self.id,
        "estado" : self.estado,
        "cidade" : self.cidade,
        "endereco" : self.endereco,
        "complemento" : self.complemento,
        "cep" : self.cep,
        "telefone" : self.telefone,
        "email" : self.email,
        "senha" : self.senha,
        "data_surgimento" : self.data_surgimento
    }
```

Print do código de teste da classe do trabalho:

```python
# para testar
if __name__ == "__main__":
    # comando para remover arquivo banco de dados caso já exista
    if os.path.exists(arquivobd):
        os.remove(arquivobd)

    # comando para criar tabela (classe usuario)
    db.create_all()

    # criar objetos (na memória, sem persistência)
    p1 = Usuario(estado="SC", cidade="Blumenau", endereco="endereçoteste",
    complemento="Alto", cep="89037-255", telefone="992922070",
    email="lucasv@email.com", senha="123", data_surgimento="10/10/2020")
    p2 = Usuario(estado="PR", cidade="Itaporobo", endereco="endereçoteste",
    complemento="vermelho", cep="32076-454", telefone="85920132",
    email="sabrino@email.com", senha="543", data_surgimento="21/40/2019")

    # para tornar os objetos persistentes
    db.session.add(p1)
    db.session.add(p2)
    db.session.commit()

    # exibir pessoa
    print(p1.json())
    print(p2)
```

Prints do código da definição da classe nova:

medico.py

BackEnd > medico.py

```python
"""Este módulo contém a classe que representa um médico

No arquivo encontra-se a implementação de uma classe: Medico.

Autor: Gustavo Guerreiro.
"""
from config import *
from model import Usuario
from entidade import Entidade

class Medico(Usuario):
    """Classe que representa um médico no site
    """
    # atributos exclusivos do médico
    id = db.Column(db.Integer, db.ForeignKey('usuario.id'), primary_key=True)
    nome_medico = db.Column(db.String(254))
    area_atuacao = db.Column(db.String(254))
    cpf_medico = db.Column(db.String(254))
    sexo_medico = db.Column(db.String(254))

    # atributo de chave estrangeira
    id_entidade = db.Column(db.Integer, db.ForeignKey('entidade.id'), nullable=False)

    # atributo de relacionamento
    entidade = db.relationship("Entidade", foreign_keys=[id_entidade])

    # definindo indentidade polimórfica que ficará armazenada na classe pai
    # no campo type
    __mapper_args__= {
        'polymorphic_identity' : 'medico',
    }

    def __slr__(self):
        """Feita para fazer o print imprimir o dicionario contido em Medico

        Returns:
            (str): string adicionando a area de atuacao do medico ao dicionario
        """
```

```python
    def __str__(self):
        """Feita para fazer o print imprimir o dicionario contido em Medico

        Returns:
            (str): string adicionando a area de atuacao do medico ao dicionario
        """
        return super().__str__() + f', {self.nome_medico}, ' +\
        f'{self.cpf_medico}, {self.sexo_medico}, ' +\
        f'{self.area_atuacao}, {self.id_entidade}, {self.entidade}'

    def json(self):
        """Adiciona area_atuacao ao return do json do Usuario

        Returns:
            (dict): dicionário json com a area de atuação inclusa
        """
        return super().json() | {
            "nome_medico" : self.nome_medico,
            "area_atuacao" : self.area_atuacao,
            "cpf_medico" : self.cpf_medico,
            "sexo_medico" : self.sexo_medico,
            "id_entidade" : self.id_entidade,
            "entidade" : self.entidade.json(),
        }
```

Print do código do teste:

```python

# parte de teste da classe
if __name__ == "__main__":
    # se um banco de dados já exister, o comando a seguir o apaga
    if os.path.exists(arquivobd):
        os.remove(arquivobd)

    # comando cria uma tabela
    db.create_all()

    # criando instancia de entidade para fazer o teste do médico
    entidade_teste = Entidade(estado="SC", cidade="Blumenau",
    endereco="Alguma Casa", complemento="Alto", cep="89037-255",
    telefone="992922070", email="lucasv@email.com", senha="123",
    data_surgimento="10/10/2020", nome_fantasia="bananinha",
    razao_social="coca-cola", numero_funcionarios="40",
    tipo_instituicao="médica", cnpj="32323232")

    # comandos para testar o módulo (criar objetos na memória sem persistência)
    test1 = Medico(estado="RS", cidade="Porto Triste",
    endereco="Algum Apartamento", complemento="Baixo", cep="55555-955",
    telefone="47888888888", email="gustavog@email.com", senha="321",
    data_surgimento="11/11/2011", nome_medico="Jorge", area_atuacao="Urologia",
    sexo_medico="Masculino", cpf_medico="989.654.258-89", id_entidade="1")

    # torna os objetos persistentes
    db.session.add(entidade_teste)
    db.session.add(test1)
    db.session.commit()

    # exibe o médico
    print(test1.json())
    print("-----------------------------------------------")
    # acessa o nome fantasia da entidade por meio do médico
    print(test1.entidade.nome_fantasia)
```

Print do código rodando no terminal:

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE                                Python

Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Experimente a nova plataforma cruzada PowerShell https://aka.ms/pscore6

PS C:\Users\Usuário\Documents\GitHub\SitePlataformaMedica> & C:/Users/Usuário/AppData/Local/Programs/Python/Python39/python.exe c:/Users/Usuário/Documents/GitHub/SitePlataformaMedica/BackEnd/medico.py
{'id': 2, 'estado': 'RS', 'cidade': 'Porto Triste', 'endereco': 'Algum Apartamento', 'complemento': 'Baixo', 'cep': '55555-955', 'telefone': '47888888888', 'email': 'gustavog@email.com', 'senha': '321', 'data_surgimento': '11/11/2011', 'nome_medico': 'Jorge', 'area_atuacao': 'Urologia', 'cpf_medico': '989.654.258-89', 'sexo_medico': 'Masculino', 'id_entidade': 1, 'entidade': {'id': 1, 'estado': 'SC', 'cidade': 'Blumenau', 'endereco': 'Alguma Casa', 'complemento': 'Alto', 'cep': '89037-255', 'telefone': '992922070', 'email': 'lucasv@email.com', 'senha': '123', 'data_surgimento': '10/10/2020', 'nome_fantasia': 'bananinha', 'razao_social': 'coca-cola', 'numero_funcionarios': '40', 'tipo_instituicao': 'médica', 'cnpj': 32323232}}
-----------------------------------------------
bananinha
PS C:\Users\Usuário\Documents\GitHub\SitePlataformaMedica>
```

Print código rodando json:

```
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Usuário>curl localhost:5000/listar_usuarios
[
  {
    "cep": "89037-255",
    "cidade": "Blumenau",
    "cnpj": 32323232,
    "complemento": "Alto",
    "data_surgimento": "10/10/2020",
    "email": "lucasv@email.com",
    "endereco": "Alguma Casa",
    "estado": "SC",
    "id": 1,
    "nome_fantasia": "bananinha",
    "numero_funcionarios": "40",
    "razao_social": "coca-cola",
    "senha": "123",
    "telefone": "992922070",
    "tipo_instituicao": "m\u00e9dica"
  },
  {
    "area_atuacao": "Urologia",
    "cep": "55555-955",
    "cidade": "Porto Triste",
    "complemento": "Baixo",
    "cpf_medico": "989.654.258-89",
    "data_surgimento": "11/11/2011",
    "email": "gustavog@email.com",
    "endereco": "Algum Apartamento",
    "entidade": {
      "cep": "89037-255",
      "cidade": "Blumenau",
      "cnpj": 32323232,
      "complemento": "Alto",
      "data_surgimento": "10/10/2020",
      "email": "lucasv@email.com",
      "endereco": "Alguma Casa",
      "estado": "SC",
      "id": 1,
      "nome_fantasia": "bananinha",
      "numero_funcionarios": "40",
      "razao_social": "coca-cola",
      "senha": "123",
      "telefone": "992922070",
      "tipo_instituicao": "m\u00e9dica"
    },
    "estado": "RS",
    "id": 2,
    "id_entidade": 1,
    "nome_medico": "Jorge",
    "senha": "321",
    "sexo_medico": "Masculino",
    "telefone": "47888888888"
  }
]
```

Link para o github: https://github.com/GuerreiroG/SitePlataformaMedica

O código fonte encontra-se na pasta “BackEnd” o arquivo da classe nova criada se chama “medico.py” e a classe do trabalho é “model.py”