

## Exercício Avaliativo – S202 – L2

- Class Database

```
1 import pymongo
2
3
4 class Database:
5
6     def __init__(self, database, collection):
7         self.connect(database, collection)
8
9     def connect(self, database, collection):
10        try:
11            connectionString = "localhost:27017"
12            self.clusterConnection = pymongo.MongoClient(
13                connectionString,
14                tlsAllowInvalidCertificates=True
15            )
16            self.db = self.clusterConnection[database]
17            self.collection = self.db[collection]
18            print("Database connected successfully!")
19        except Exception as e:
20            print(e)
21
22    def discoconnect(self):
23        try:
24            self.db.drop_collection(self.collection)
25            print("Database reseted successfully!")
26        except Exception as e:
27            print(e)
28
```

- Class ZoologicoDAO

```
1 from Database import Database
2
3
4 class ZoologicoDAO:
5
6     def __init__(self):
7         self.db = Database('S202', 'Animal')
8
9     def createAnimal(self, animal):
10        novoAnimal = {
11            "id": animal.id,
12            "name": animal.nome,
13            "especie": animal.especie,
14            "idade": animal.idade,
15            "habitat": [{
16                'id': animal.habitat.id,
17                'nome': animal.habitat.nome,
18                'tipoAmbiente': animal.habitat.tipoAmbiente,
19                'cuidador': {
20                    'id': animal.habitat.cuidador.id,
21                    'nome': animal.habitat.cuidador.nome,
22                    'documento': animal.habitat.cuidador.documento
23                }
24            }]
25        }
26        self.db.collection.insert_one(novoAnimal)
27
```

```
main.py x zoologicoCLI.py x animal.py x habitat.py x cuidador.py x zoologicoDAO.py x Database.py x
24     }}
25 }
26 self.db.collection.insert_one(novoAnimal)
27
28 def readAnimal(self, id):
29     animal = self.db.collection.find({'id': id})
30     return animal
31
32 def upadteAnimal(self, id, nome, idade):
33     result = self.db.collection.update_one(
34         {'id': id},
35         {"$set": {"name": 'aaa'}}
36     )
37     if (result.acknowledged):
38         print('Atualizada com sucesso!')
39     else:
40         print('Erro ao atualizar!')
41
42 def deleteAnimal(self, id):
43     result = self.db.collection.delete_one({'id': id})
44     if (result.acknowledged):
45         print('Deletado com sucesso!')
46     else:
47         print('Erro ao deletar!')
48
ZoologicoDAO > upadteAnimal() > else
```

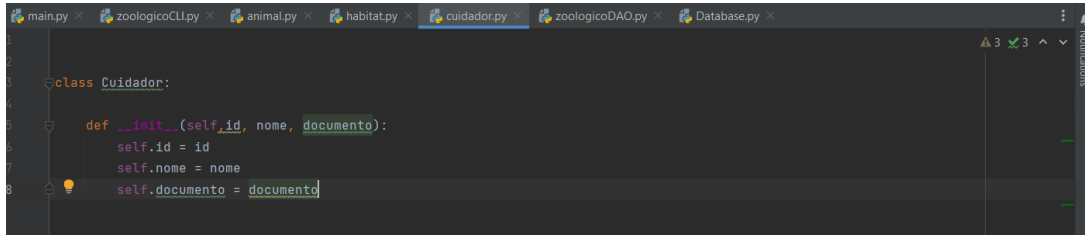
- Class Animal

```
main.py x zoologicoCLI.py x animal.py x habitat.py x cuidador.py x zoologicoDAO.py x Database.py x
1
2 class Animal:
3
4     def __init__(self, id, nome, especie, idade, habitat):
5         self.id = id
6         self.nome = nome
7         self.especie = especie
8         self.idade = idade
9         self.habitat = habitat
```

- Classe Habitat

```
main.py x zoologicoCLI.py x animal.py x habitat.py x cuidador.py x zoologicoDAO.py x Database.py x
1
2 class Habitat:
3
4     def __init__(self, id, nome, tipoAmbiente, cuidador):
5         self.id = id
6         self.nome = nome
7         self.tipoAmbiente = tipoAmbiente
8         self.cuidador = cuidador
```

- Classe Cuidador



```

1  class Cuidador:
2
3      def __init__(self, id, nome, documento):
4          self.id = id
5          self.nome = nome
6          self.documento = documento

```


- Classe Animal Compass

```

_id: ObjectId('644320e250a4d6305ec96e52')
id: "1"
name: "Juca"
especie: "Baleia"
idade: "5"
▼ habitat: Array
  ▼ 0: Object
    id: "1"
    nome: "Mar"
    tipoAmbiente: "Molhado"
  ▼ cuidador: Object
    id: "1"
    nome: "Joao"
    documento: "12345678"

```

- Zoologico CLI



```

1  from animal import Animal
2  from cuidador import Cuidador
3  from habitat import Habitat
4  from zoologicoDAO import ZoologicoDAO
5  from pprint import pprint
6
7
8  class ZoologicoCLI:
9
10     def menu(self):
11         print("0 que deseja fazer?")
12         print("1-Criar um animal")
13         print("2 - Ler um animal")
14         print("3 - Atualizar o nome do animal")
15         print("4 - Deletar um animal")
16         option = input('Entre com a sua opcao: ')
17         return option
18

```

```
main.py x zoologicoCLI.py x animal.py x habitat.py x cuidador.py x zoologicoDAO.py x Database.py x
16 option = input('Entre com a sua opcao: ')
17 return option
18
19 def createAnimal(self):
20     ZooBd = ZoologicoDAO()
21     idCuidador = input("Id do cuidador: ")
22     nomeCuidador = input("Nome do cuidador: ")
23     documentoCuidador = input("Documento do cuidador: ")
24     novoCuidador = Cuidador(idCuidador, nomeCuidador, documentoCuidador)
25
26     print(novoCuidador.id, type(novoCuidador))
27
28     idHabitat = input("Id do Habitat: ")
29     nomeHabitat = input("Nome do Habitat: ")
30     tipoHabitat = input("Tipo do Ambiente: ")
31     novoHabitat = Habitat(idHabitat, nomeHabitat, tipoHabitat, novoCuidador)
32
33     print(novoHabitat.cuidador, type(novoHabitat))
34
35     idAnimal = input("Id do Animal: ")
36     nomeAnimal = input("Nome do Animal: ")
37     especieAnimal = input("Especie do Animal: ")
38     idadeAnimal = input("Idade do Animal: ")
39     novoAnimal = Animal(idAnimal, nomeAnimal, especieAnimal, idadeAnimal, novoHabitat)
40
41     ZooBd.createAnimal(novoAnimal)
```

```
main.py x zoologicoCLI.py x animal.py x habitat.py x cuidador.py x zoologicoDAO.py x Database.py x
37 especieAnimal = input("Especie do Animal: ")
38 idadeAnimal = input("Idade do Animal: ")
39 novoAnimal = Animal(idAnimal, nomeAnimal, especieAnimal, idadeAnimal, novoHabitat)
40
41 ZooBd.createAnimal(novoAnimal)
42
43 def readAnimal(self):
44     ZooBd = ZoologicoDAO()
45     id = input('Entre com o id: ')
46     response = ZooBd.readAnimal(id)
47     for aux in response:
48         pprint(aux)
49
50 def updateAnimal(self):
51     ZooBd = ZoologicoDAO()
52     id = input('Entre com o id do animal: ')
53     novoNome = input('Entre com o novo nome do animal: ')
54     novaIdade = input('Entre com a nova idade do animal: ')
55     ZooBd.upadteAnimal(novoNome, novaIdade, id)
56
57 def deleteAnimal(self):
58     ZooBd = ZoologicoDAO()
59     id = input('Entre com o id para deletar o animal: ')
60     ZooBd.deleteAnimal(id)
61
```

- Schema de Animal

S202.Animal

0 1  
DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

```
1 {
2   $jsonSchema: {
3     bsonType: 'object',
4     required: [
5       'name',
6       'id',
7       'especie',
8       'idade',
9       'habitat'
10    ],
11    properties: {
12      id: {
13        bsonType: 'string',
14        description: 'deve ser uma string'
15      },
16      name: {
17        bsonType: 'string',
18        description: 'deve ser uma string'
19      },
20      especie: {
21        bsonType: 'string',
22        description: 'deve ser um string'
23      },
24      idade: {
25        bsonType: 'string',
26        description: 'deve ser uma string'
27      },
28      habitat: {
```

S202.Animal

0 1  
DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

```
27 },
28 habitat: {
29   bsonType: [
30     'object'
31   ],
32   items: {
33     bsonType: 'object',
34     required: [
35       'nome',
36       'tipoAmbiente',
37       'cuidador'
38     ],
39     properties: {
40       nome: {
41         bsonType: 'string',
42         description: 'deve ser um string'
43       },
44       tipoAmbiente: {
45         bsonType: 'string',
46         description: 'deve ser um string'
47       },
48       cuidador: {
49         bsonType: 'object',
50         required: [
51           'id',
52           'nome',
53           'documento'
54         ],
55         properties: {
56           id: {
57             bsonType: 'string',
58             description: 'deve ser um string'
59           },
60           nome: {
61             bsonType: 'string',
```

Documents Aggregations Schema Explain Plan Indexes Validation

```
48  //
49  cuidador: {
50    bsonType: 'object',
51    required: [
52      'id',
53      'nome',
54      'documento'
55    ],
56    properties: {
57      id: {
58        bsonType: 'string',
59        description: 'deve ser um string'
60      },
61      nome: {
62        bsonType: 'string',
63        description: 'deve ser um string'
64      },
65      documento: {
66        bsonType: 'string',
67        description: 'deve ser um string'
68      }
69    }
70  },
71  }
72  }
73  }
74  }
75  }
```