Modelagem de dados

DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

AGENDA

- Entendendo alguns comandos do Flask Migrate
- Modelagem de dados

Entendendo alguns comandos do Flask Migrate

- **flask run db init** - Tem a responsabilidade de criar a pasta migrations, que irá conter a parte de versionamento do banco de dados através do Flask.
- **flask run db migrate** - Se já houver a pasta migrations criada, ao executar esse comando irá atualizar a versão, baseada nas informações das tabelas disponíveis.
- **flask run db upgrade** - Irá atualizar o banco de dados conectado com a última versão encontrada na pasta migrations, e refletir o modelo de dados da aplicação.

Modelagem de dados - Technology

```
from flask_sqlalchemy import Model
from src.app import DB, MA
class Technology(DB.Model):
  __tablename__ = 'technologies'
  id = DB.Column(DB.Integer, autoincrement=True, primary_key=True)
  name = DB.Column(DB.String(84), nullable=False)
  def __init__(self, name):
    self.name = name
class TechnologySchema(MA.Schema):
  class Meta:
    fields = ('id', 'name')
technology_share_schema = TechnologySchema()
technologies_share_schema = TechnologySchema(many=True)
```

Modelagem de dados - Country

```
from src.app import DB, MA

class Country(DB.Model):

  __tablename__ = 'countries'

  id = DB.Column(DB.Integer, autoincrement = True, primary_key = True)

  name = DB.Column(DB.String(84), nullable = False)

  language = DB.Column(DB.String(84), nullable = False)
```

Modelagem de dados - Country

```
  def __init__(self, name, language):
    self.name = name
    self.language = language
class CountrySchema(MA.Schema):
  class Meta:
    fields = ('id', 'name', 'language')
country_share_schema = CountrySchema()
countries_share_schema = CountrySchema(many = True)
```

Modelagem de dados - State

```python
from src.app import DB, MA
from src.app.models.country import Country
class State(DB.Model):
  __tablename__ = "states"
  id = DB.Column(DB.Integer, autoincrement = True, primary_key = True)
  country_id = DB.Column(DB.Integer, DB.ForeignKey(Country.id), nullable = False)
  name = DB.Column(DB.String(84), nullable = False)
  initials = DB.Column(DB.String(2), nullable = False)
  def __init__(self, country_id, name, initials):
    self.country_id = country_id
    self.name = name
    self.initials = initials
```

Modelagem de dados - State

```
class StateSchema(MA.Schema):
  class Meta:
    fields = ('id', 'country_id', 'name', 'initials')
state_share_schema = StateSchema()
states_share_schema = StateSchema(many = True)
```

Modelagem de dados - City

```python
from src.app import DB, MA

from src.app.models.state import State

class City(DB.Model):

  __tablename__ = 'cities'

  id = DB.Column(DB.Integer, autoincrement = True, primary_key = True)

  state_id = DB.Column(DB.Integer, DB.ForeignKey(State.id), nullable = False)

  name = DB.Column(DB.String(84), nullable = False)

  def __init__(self, state_id, name):

    self.state_id = state_id

    self.name = name
```

Modelagem de dados - City

```
class CitySchema(MA.Schema):
  class Meta:
    fields = ('id', 'state_id', 'name')
city_share_schema = CitySchema()
cities_share_schema = CitySchema(many = True)
```

Modelagem de dados - User

```python
from src.app import DB, MA
from src.app.models.city import City
class User(DB.Model):
  __tablename__ = 'users'
  id = DB.Column(DB.Integer, autoincrement = True, primary_key = True)
  city_id = DB.Column(DB.Integer, DB.ForeignKey(City.id), nullable = False)
  name = DB.Column(DB.String(84), nullable = False)
  age = DB.Column(DB.Integer, nullable = False)
  email = DB.Column(DB.String(84), nullable = False)
  password = DB.Column(DB.String(84), nullable = False)
```

Modelagem de dados - User

```
  def __init__(self, city_id, name, age, email, password):
    self.city_id = city_id
    self.name = name
    self.age = age
    self.email = email
    self.password = password
class UserSchema(MA.Schema):
  class Meta:
    fields = ('id', 'city_id', 'name', 'age', 'email', 'password')
user_share_schema = UserSchema()
users_share_schema = UserSchema(many = True)
```

Modelagem de dados - Developer

```python
from src.app import DB, MA

from src.app.models.user import User

class Developer(DB.Model):

  __tablename__ = "developers"

  id = DB.Column(DB.Integer, autoincrement=True, primary_key=True)

  months_experience = DB.Column(DB.Integer, nullable = False)

  accepted_remote_work = DB.Column(DB.Boolean, nullable = False, default = True)

  user_id = DB.Column(DB.Integer, DB.ForeignKey(User.id), nullable = True)
```

Modelagem de dados - Developer

```python
  def __init__(self, months_experience, accepted_remote_work, user_id):

    self.months_experience = months_experience

    self.accepted_remote_work = accepted_remote_work

    self.user_id = user_id

class DeveloperSchema(MA.Schema):

  class Meta:

    fields = ('id', 'months_experience', 'accepted_remote_work', 'user_id')

developer_share_schema = DeveloperSchema()

developers_share_schema = DeveloperSchema(many = True)
```

DEVinHouse

Modelagem de dados - Developer_technology

```python
from src.app import DB, MA

from src.app.models.developer import Developer

from src.app.models.technology import Technology

class DeveloperTechnology(DB.Model):

  __tablename__ = 'developer_technologies'

  id = DB.Column(DB.Integer, autoincrement = True, primary_key = True)

  technology_id = DB.Column(DB.Integer, DB.ForeignKey(Technology.id), nullable =

False)

  developer_id = DB.Column(DB.Integer, DB.ForeignKey(Developer.id), nullable = False)

  is_main_tech = DB.Column(DB.Boolean, nullable = False, default = False)
```

Modelagem de dados - Developer_technology

```
def __init__(self, technology_id, developer_id, is_main_tech):
    self.technology_id = technology_id
    self.developer_id = developer_id
    self.is_main_tech = is_main_tech
class DeveloperTechnologySchema(MA.Schema):
  class Meta:
    fields = ('id', 'technology_id', 'developer_id', 'is_main_tech')
developer_technology_schema = DeveloperTechnologySchema()
developer_technologies_schema = DeveloperTechnologySchema(many = True)
```

DEVinHouse

Parcerias para desenvolver a sua carreira

OBRIGADO!

SENAI <LAB365>