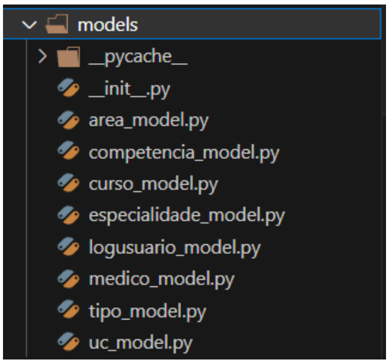
# Manual do Programador

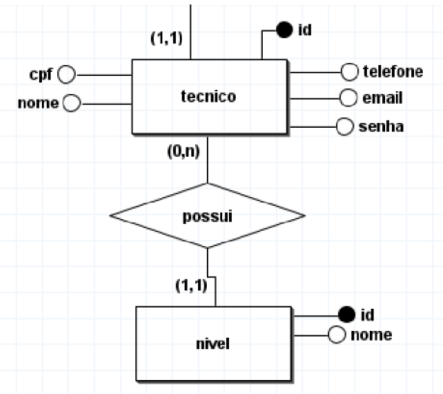
**Projeto Integrador – SENAC**

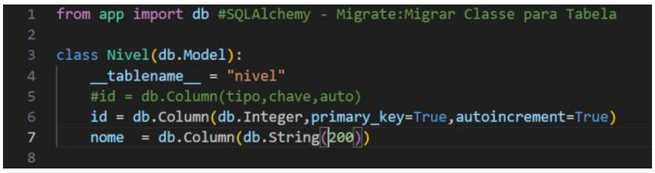
**Exemplo: Alpha InfoSystem**

## **Passo 1**. Escolha uma **entidade forte** (não tem **PK**) no diagrama

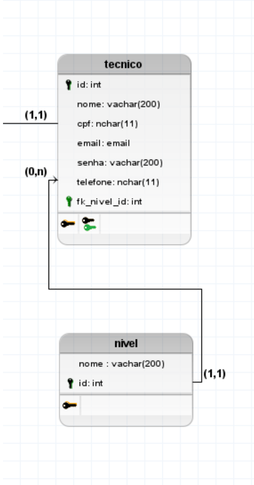


```
models
  > __pycache__
    __init__.py
    area_model.py
    competencia_model.py
    curso_model.py
    especialidade_model.py
    logusuario_model.py
    medico_model.py
    tipo_model.py
    uc_model.py
```

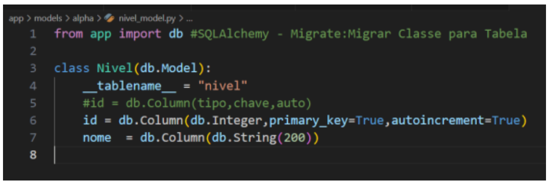




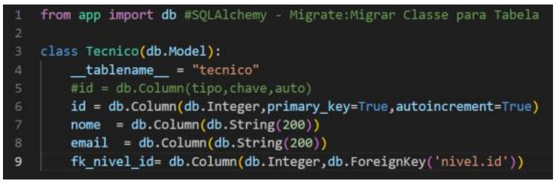
```python
from app import db #SQLAlchemy - Migrate:Migrar Classe para Tabela

class Nivel(db.Model):
    __tablename__ = "nivel"
    #id = db.Column(tipo,chave,auto)
    id = db.Column(db.Integer,primary_key=True,autoincrement=True)
    nome  = db.Column(db.String(200))

```



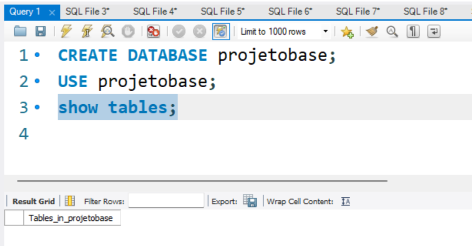
## Camada Model – OK



```python
app > models > alpha > nivel_model.py > ...
from app import db #SQLAlchemy - Migrate:Migrar Classe para Tabela

class Nivel(db.Model):
    __tablename__ = "nivel"
    #id = db.Column(tipo,chave,auto)
    id = db.Column(db.Integer,primary_key=True,autoincrement=True)
    nome  = db.Column(db.String(200))

```

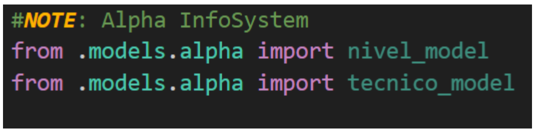


```python
from app import db #SQLAlchemy - Migrate:Migrar Classe para Tabela

class Tecnico(db.Model):
    __tablename__ = "tecnico"
    #id = db.Column(tipo,chave,auto)
    id = db.Column(db.Integer,primary_key=True,autoincrement=True)
    nome  = db.Column(db.String(200))
    email  = db.Column(db.String(200))
    fk_nivel_id= db.Column(db.Integer,db.ForeignKey('nivel.id'))
```



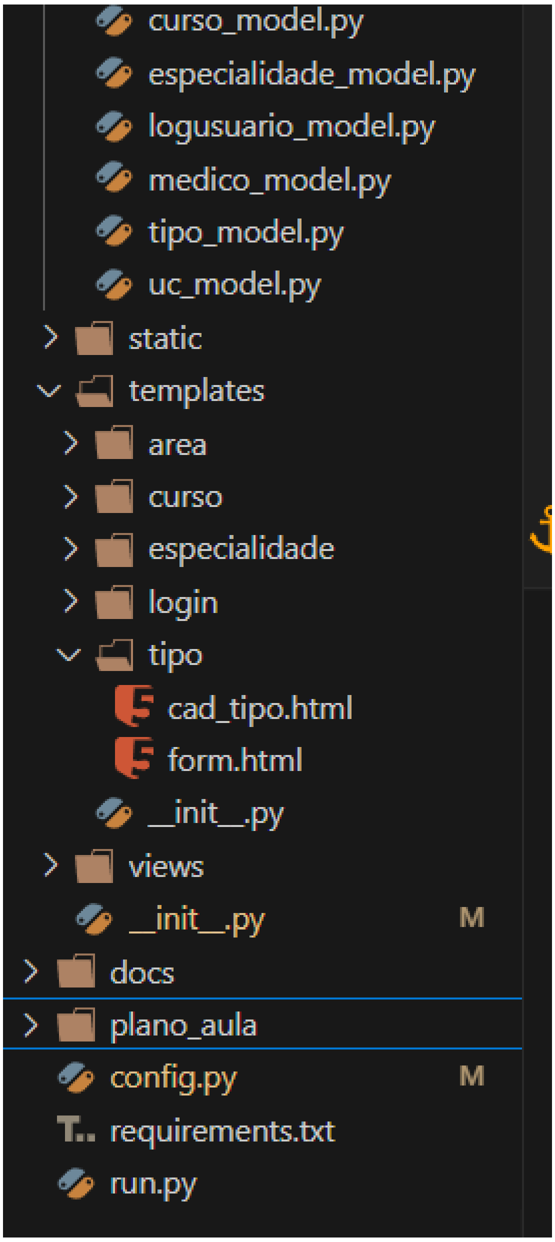
```sql
CREATE DATABASE projetobase;
USE projetobase;
show tables;
```

Tables_in_projetobase

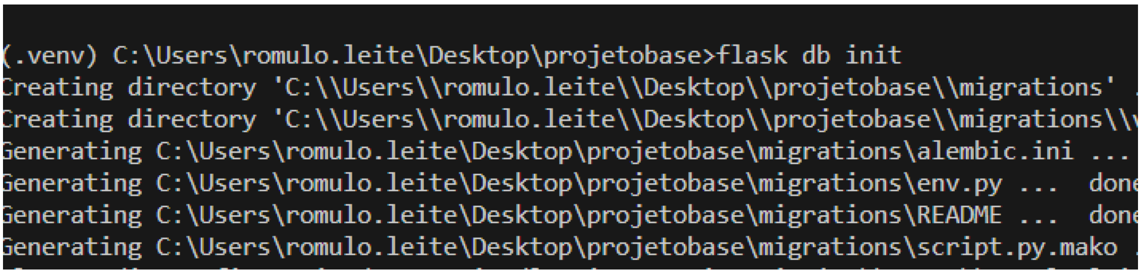


```python
#NOTE: Alpha InfoSystem
from .models.alpha import nivel_model
from .models.alpha import tecnico_model
```

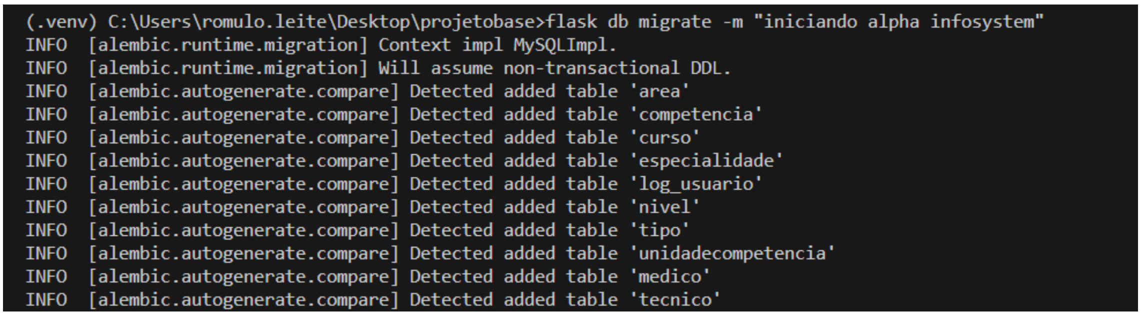
**Não tenho a pasta migrations**



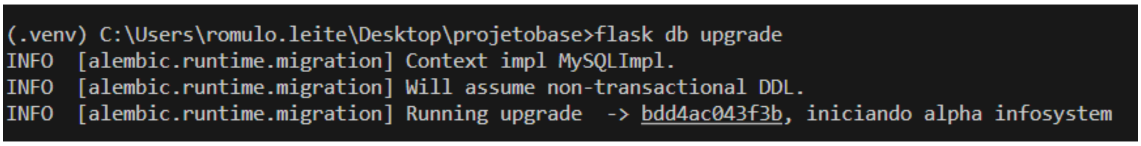
```
        curso_model.py
        especialidade_model.py
        logusuario_model.py
        medico_model.py
        tipo_model.py
        uc_model.py
  > static
  v templates
    > area
    > curso
    > especialidade
    > login
    v tipo
        cad_tipo.html
        form.html
      __init__.py
  > views
    __init__.py                 M
> docs
> plano_aula
  config.py                     M
  requirements.txt
  run.py
```



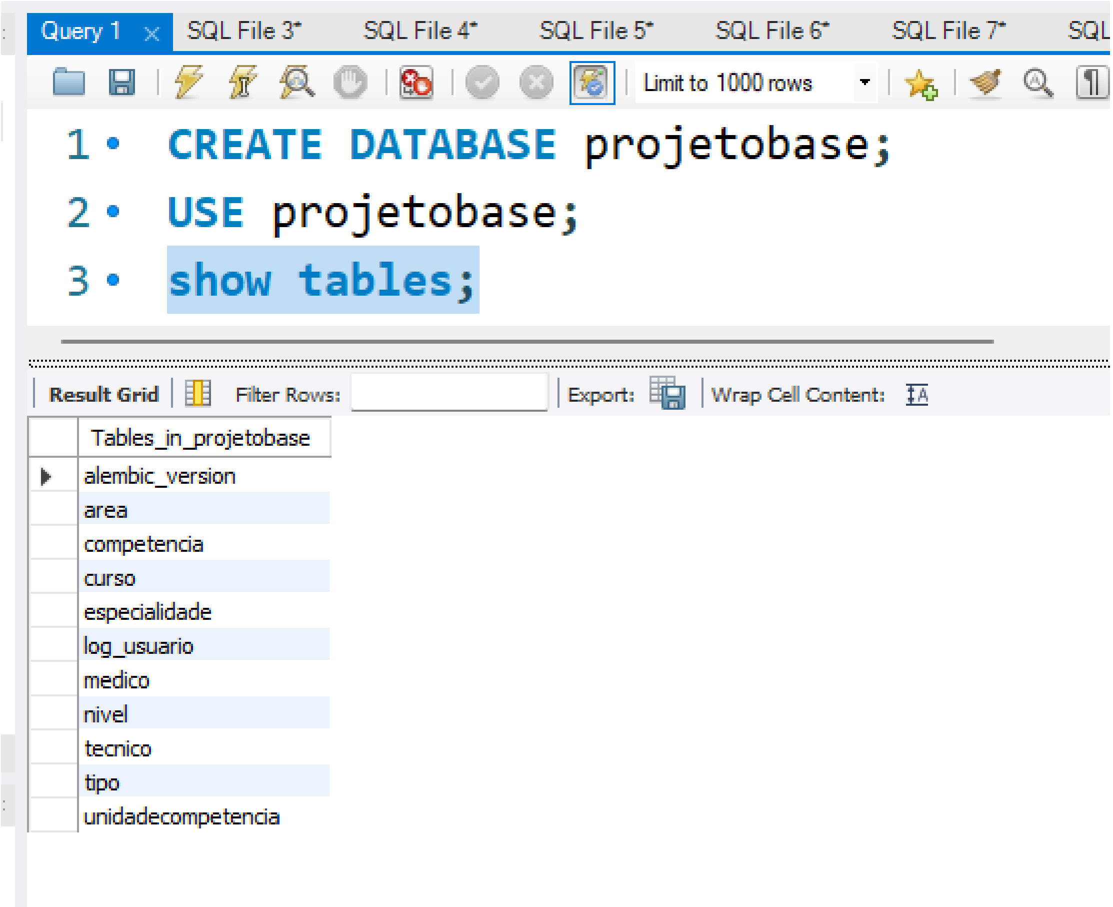
```
(.venv) C:\Users\romulo.leite\Desktop\projetobase>flask db init
Creating directory 'C:\\Users\\romulo.leite\\Desktop\\projetobase\\migrations'
Creating directory 'C:\\Users\\romulo.leite\\Desktop\\projetobase\\migrations\\
Generating C:\Users\romulo.leite\Desktop\projetobase\migrations\alembic.ini ...
Generating C:\Users\romulo.leite\Desktop\projetobase\migrations\env.py ...  done
Generating C:\Users\romulo.leite\Desktop\projetobase\migrations\README ...  done
Generating C:\Users\romulo.leite\Desktop\projetobase\migrations\script.py.mako
```



```
 (.venv) C:\Users\romulo.leite\Desktop\projetobase>flask db migrate -m "iniciando alpha infosystem"
 INFO  [alembic.runtime.migration] Context impl MySQLImpl.
 INFO  [alembic.runtime.migration] Will assume non-transactional DDL.
 INFO  [alembic.autogenerate.compare] Detected added table 'area'
 INFO  [alembic.autogenerate.compare] Detected added table 'competencia'
 INFO  [alembic.autogenerate.compare] Detected added table 'curso'
 INFO  [alembic.autogenerate.compare] Detected added table 'especialidade'
 INFO  [alembic.autogenerate.compare] Detected added table 'log_usuario'
 INFO  [alembic.autogenerate.compare] Detected added table 'nivel'
 INFO  [alembic.autogenerate.compare] Detected added table 'tipo'
 INFO  [alembic.autogenerate.compare] Detected added table 'unidadecompetencia'
 INFO  [alembic.autogenerate.compare] Detected added table 'medico'
 INFO  [alembic.autogenerate.compare] Detected added table 'tecnico'
```

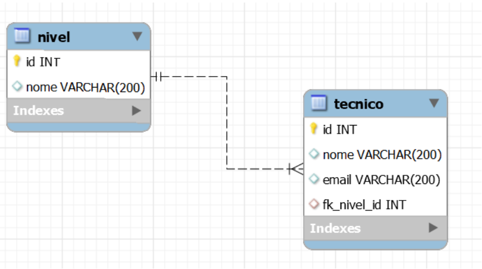


```
(.venv) C:\Users\romulo.leite\Desktop\projetobase>flask db upgrade
INFO  [alembic.runtime.migration] Context impl MySQLImpl.
INFO  [alembic.runtime.migration] Will assume non-transactional DDL.
INFO  [alembic.runtime.migration] Running upgrade  -> bdd4ac043f3b, iniciando alpha infosystem
```



```
CREATE DATABASE projetobase;
USE projetobase;
show tables;
```

| Tables_in_projetobase |
|---|
| alembic_version |
| area |
| competencia |
| curso |
| especialidade |
| log_usuario |
| medico |
| nivel |
| tecnico |
| tipo |
| unidadecompetencia |

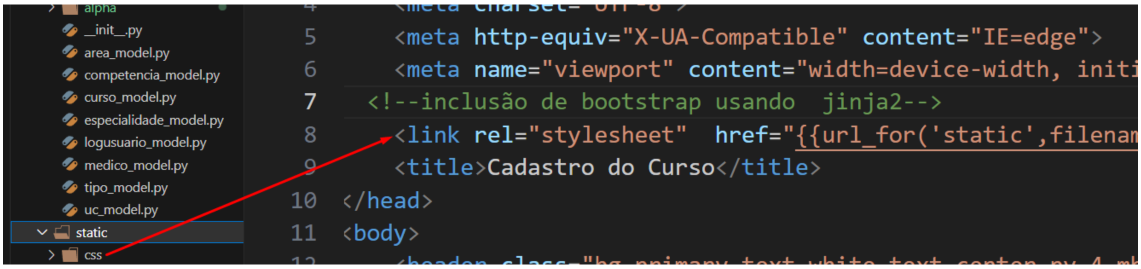


## *Etapa Model Finalizada*

## *Front-End*

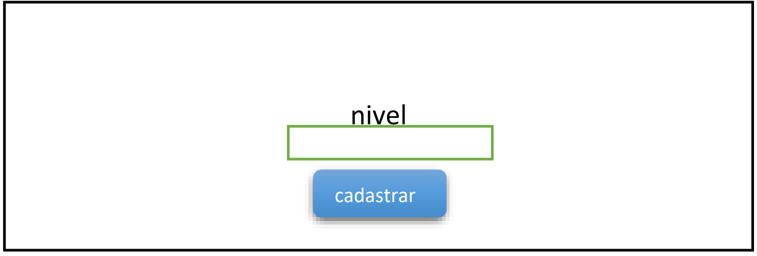
- ***HTML***
- ***CSS***
- ***Java Script***

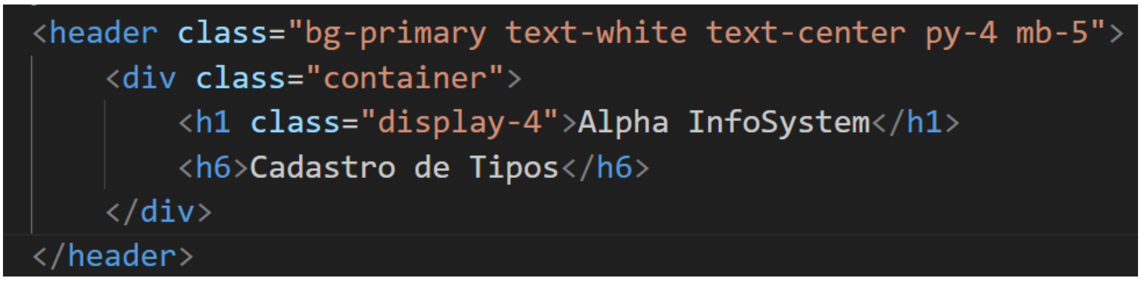
***HTML+CSS+JavaScript = Bootstrap***



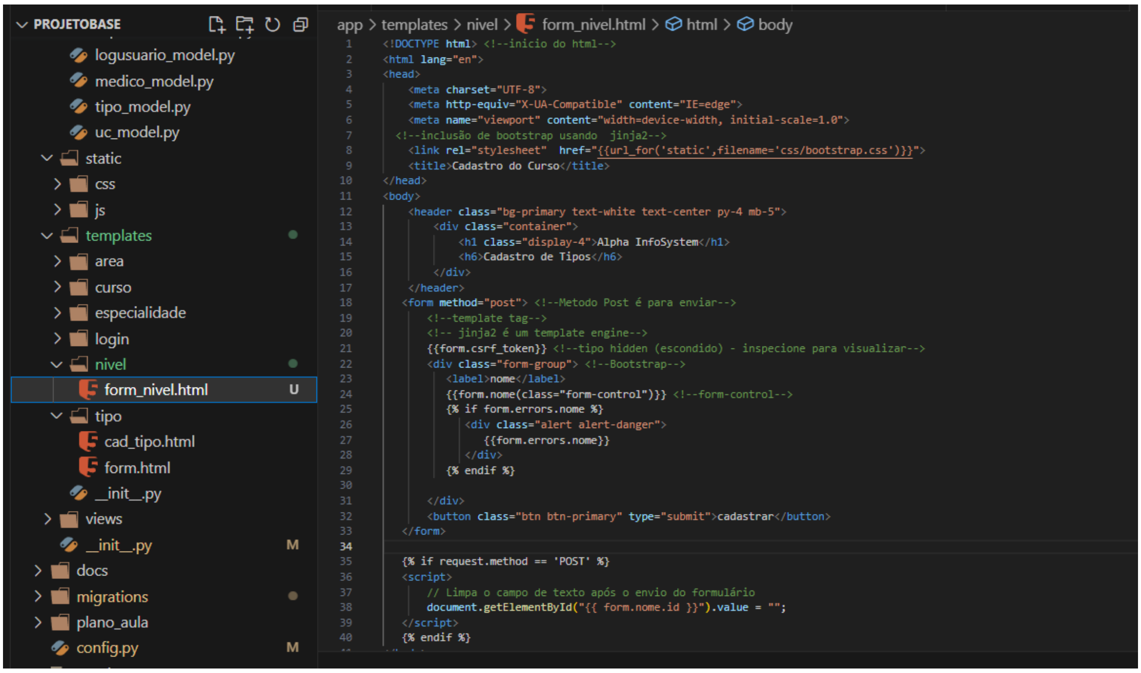
```
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initi
  <!--inclusão de bootstrap usando  jinja2-->
    <link rel="stylesheet"  href="{{url_for('static',filenam
    <title>Cadastro do Curso</title>
</head>
<body>
    <header class="bg-primary text-white text-center py-4 mb
```

***localhost/cadnivel***





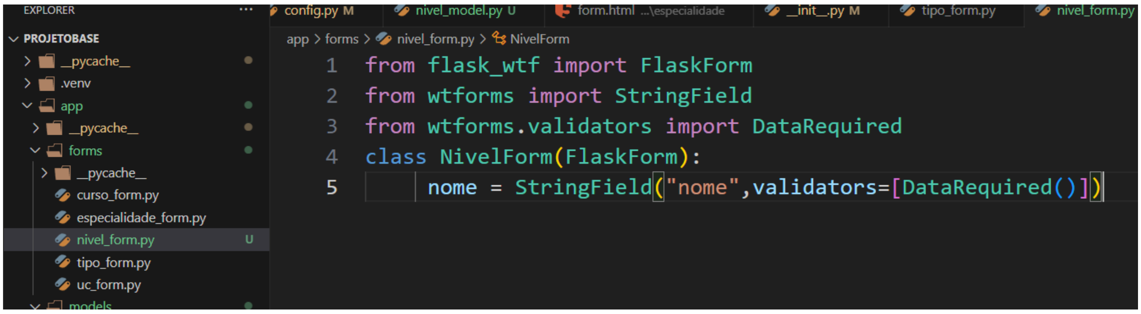
```
<header class="bg-primary text-white text-center py-4 mb-5">
    <div class="container">
        <h1 class="display-4">Alpha InfoSystem</h1>
        <h6>Cadastro de Tipos</h6>
    </div>
</header>
```



```html
<!DOCTYPE html> <!--inicio do html-->
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <!--inclusão de bootstrap usando  jinja2-->
    <link rel="stylesheet"  href="{{url_for('static',filename='css/bootstrap.css')}}">
    <title>Cadastro do Curso</title>
</head>
<body>
    <header class="bg-primary text-white text-center py-4 mb-5">
        <div class="container">
            <h1 class="display-4">Alpha InfoSystem</h1>
            <h6>Cadastro de Tipos</h6>
        </div>
    </header>
   <form method="post"> <!--Metodo Post é para enviar-->
       <!--template tag-->
       <!-- jinja2 é um template engine-->
       {{form.csrf_token}} <!--tipo hidden (escondido) - inspecione para visualizar-->
       <div class="form-group"> <!--Bootstrap-->
          <label>nome</label>
          {{form.nome(class="form-control")}} <!--form-control-->
          {% if form.errors.nome %}
             <div class="alert alert-danger">
                {{form.errors.nome}}
             </div>
          {% endif %}

       </div>
       <button class="btn btn-primary" type="submit">cadastrar</button>
   </form>

   {% if request.method == 'POST' %}
   <script>
       // Limpa o campo de texto após o envio do formulário
       document.getElementById("{{ form.nome.id }}").value = "";
   </script>
   {% endif %}
```

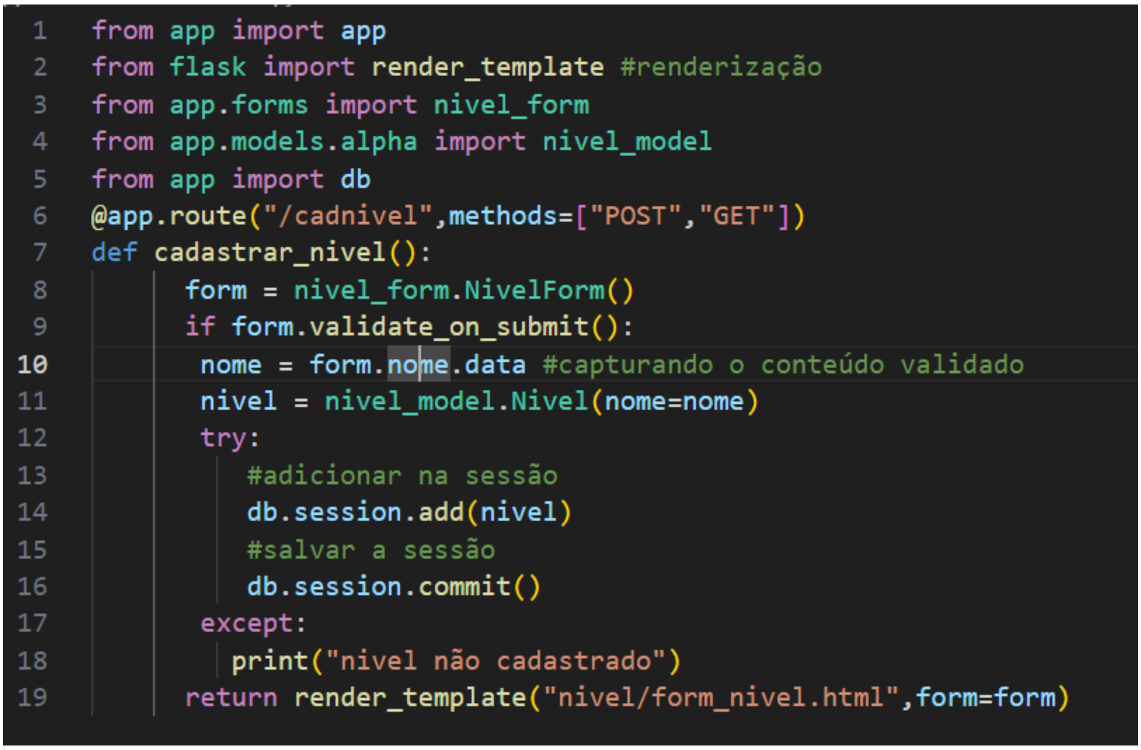
## ***Formulário com Jinja 2***

## ***Camada form (validação de formulários)***

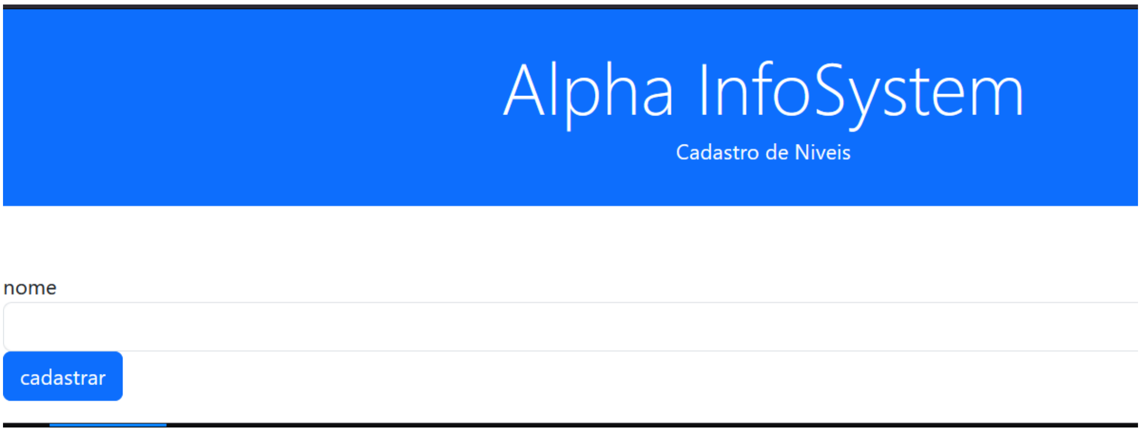


```python
from flask_wtf import FlaskForm
from wtforms import StringField
from wtforms.validators import DataRequired
class NivelForm(FlaskForm):
    nome = StringField("nome",validators=[DataRequired()])
```

***<u>Rota – camada view</u>***



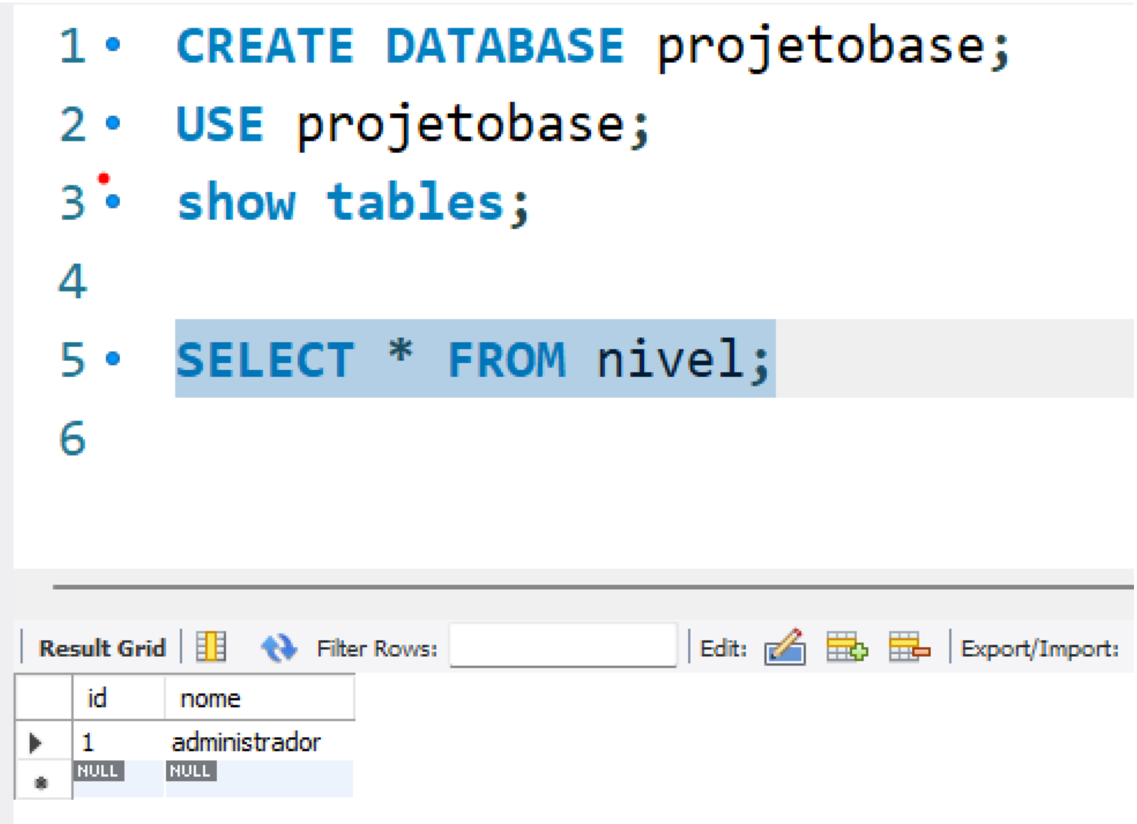
```python
from app import app
from flask import render_template #renderização
from app.forms import nivel_form
from app.models.alpha import nivel_model
from app import db
@app.route("/cadnivel",methods=["POST","GET"])
def cadastrar_nivel():
      form = nivel_form.NivelForm()
      if form.validate_on_submit():
       nome = form.nome.data #capturando o conteúdo validado
       nivel = nivel_model.Nivel(nome=nome)
       try:
          #adicionar na sessão
          db.session.add(nivel)
          #salvar a sessão
          db.session.commit()
       except:
         print("nivel não cadastrado")
      return render_template("nivel/form_nivel.html",form=form)
```



Alpha InfoSystem

Cadastro de Niveis

nome



```
CREATE DATABASE projetobase;
USE projetobase;
show tables;

SELECT * FROM nivel;

```

| id | nome |
|---|---|
| 1 | administrador |
| NULL | NULL |