

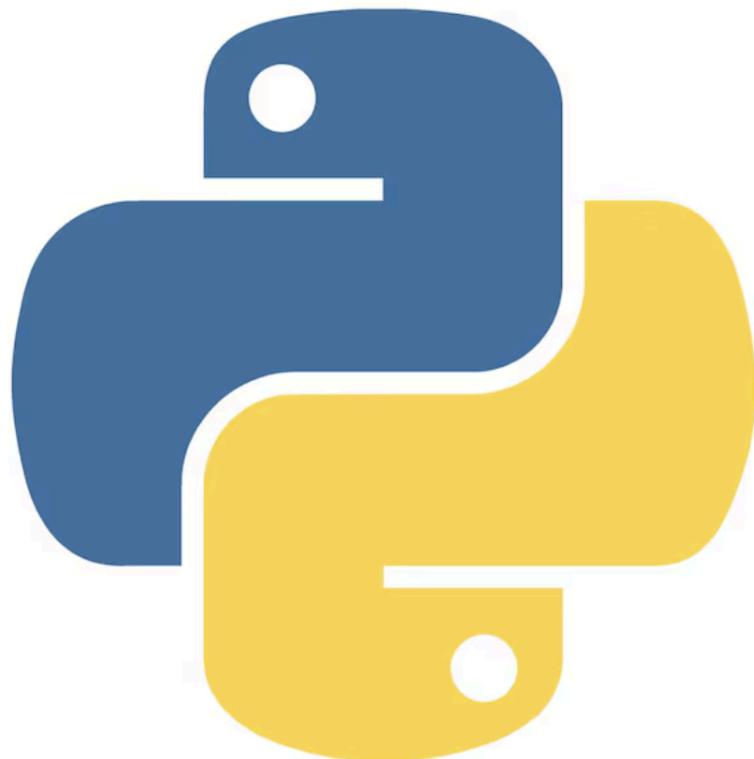


INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL, I.P.
CENTRO DE EMPREGO E FORMAÇÃO PROFISSIONAL DE BRAGA

EFA T – PROGRAMADORA DE INFORMÁTICA

UFCD 10794

Python Avançado



Leonardo Souza, 06.

Trabalho sobre: Plataforma Web para uma

Clínica Veterinária.

Formador: Marcos Alvarães



EFA T – PROGRAMADORA DE INFORMÁTICA

Leonardo Souza, 06.....	1
Resumo do Projeto.....	3
Sistema de Clínica Veterinária.....	3
Este projeto consiste no desenvolvimento de um sistema web para gestão de uma clínica veterinária, com o objetivo de facilitar o controle de utilizadores, clientes, animais e consultas, centralizando todas as informações em uma única plataforma.....	3
O sistema foi desenvolvido utilizando o framework Flask (Python), seguindo o padrão MVC (Model–View–Controller), onde:.....	3
• o backend é responsável pelo processamento das regras de negócio,.....	3
• o frontend apresenta as interfaces ao utilizador através de templates HTML,.....	3
• e o banco de dados MySQL armazena todas as informações do sistema.....	3
A aplicação possui um sistema de autenticação e controle de acesso, permitindo que apenas utilizadores autenticados accessem áreas restritas. Os utilizadores são organizados por perfis (admin, staff e cliente), onde cada perfil possui permissões específicas dentro do sistema.....	3
Entre as principais funcionalidades do sistema destacam-se:.....	3
• Registo e autenticação de utilizadores.....	3
• Gestão de clientes.....	3
• Cadastro e listagem de animais associados a cada cliente.....	3
• Registo e consulta de consultas veterinárias.....	3
• Área exclusiva do cliente (“Minha Área”), onde é possível visualizar seus dados, seus animais e suas consultas.....	4
• Proteção de rotas utilizando sessões para garantir a segurança da aplicação.....	4
O projeto tem como foco organização, segurança e usabilidade, permitindo que a clínica tenha um controle eficiente das informações, reduzindo erros manuais e melhorando o atendimento aos clientes.....	4
Documentação das Funções - Clínica Veterinária.....	4
Função: ligar_bd.....	4
Função: executar_query.....	5
Função: coleta_user_role.....	6
Função: base.....	6
Função: login.....	7
Função: logout.....	8
Função: home.....	8
Função: tabela_utilizadores.....	10
Função: tabela_animais.....	11
Função: tabela_consultas.....	12
Função: registrar_novo_cliente.....	13
Função: registrar_novo_utilizador.....	15
Função: editar_users.....	16
Função: nova_consulta.....	19
Função: deleta_cliente.....	20
Função: deleta_utilizador.....	21



INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL, I.P.
CENTRO DE EMPREGO E FORMAÇÃO PROFISSIONAL DE BRAGA

EFA T – PROGRAMADORA DE INFORMÁTICA

Função: deleta_consulta.....	22
Função: editar_cliente.....	22
Função: editar_animal.....	24
Função: editar_consulta.....	25
Função: editar_users.....	26
Função: trocar_password.....	27



Resumo do Projeto

Sistema de Clínica Veterinária

Este projeto consiste no desenvolvimento de um **sistema web para gestão de uma clínica veterinária**, com o objetivo de facilitar o controle de utilizadores, clientes, animais e consultas, centralizando todas as informações em uma única plataforma.

O sistema foi desenvolvido utilizando o **framework Flask (Python)**, seguindo o padrão **MVC (Model-View-Controller)**, onde:

- o **backend** é responsável pelo processamento das regras de negócio,
- o **frontend** apresenta as interfaces ao utilizador através de templates HTML,
- e o **banco de dados MySQL** armazena todas as informações do sistema.

A aplicação possui um sistema de **autenticação e controle de acesso**, permitindo que apenas utilizadores autenticados accessem áreas restritas. Os utilizadores são organizados por **perfis (admin, staff e cliente)**, onde cada perfil possui permissões específicas dentro do sistema.

Entre as principais funcionalidades do sistema destacam-se:

- Registo e autenticação de utilizadores
- Gestão de clientes
- Cadastro e listagem de animais associados a cada cliente
- Registo e consulta de consultas veterinárias



EFA T – PROGRAMADORA DE INFORMÁTICA

- Área exclusiva do cliente (“Minha Área”), onde é possível visualizar seus dados, seus animais e suas consultas
- Proteção de rotas utilizando sessões para garantir a segurança da aplicação

O projeto tem como foco **organização, segurança e usabilidade**, permitindo que a clínica tenha um controle eficiente das informações, reduzindo erros manuais e melhorando o atendimento aos clientes.

Documentação das Funções - Clínica Veterinária

Função: [ligar_bd](#)

Descrição: Realiza a conexão com o banco de dados MySQL.

Print da função:

```
# -----
# FUNÇÃO PARA CONEXÃO COM MYSQL
# -----
def ligar_bd():
    """
    Cria e retorna a conexão com o servidor MySQL
    """
    return mysql.connector.connect(
        host="62.28.39.135",
        user="efa0125",
        password="123.Abc",
        database="efa0125_06_Leonardo_clinica_veterinaria"
    )
```



Função: `executar_query`

Descrição: Executa comandos SQL de forma segura.

Print da função:

```
# -----
# HELPER DE EXECUÇÃO SEGURA (TRY / EXCEPT)
# -----
def executar_query(query, params=None, fetchone=False, fetchall=False, commit=False):
    cnx = None
    cursor = None
    try:
        cnx = ligar_bd()
        cursor = cnx.cursor(dictionary=True)
        cursor.execute(query, params or ())
        if fetchone:
            return cursor.fetchone()
        if fetchall:
            return cursor.fetchall()

        if commit:
            cnx.commit()

    except mysql.connector.Error as err:
        if cnx:
            cnx.rollback()
        print("ERRO BD:", err)
        raise err

    finally:
        if cursor:
            cursor.close()
        if cnx:
            cnx.close()
```



Função: coleta_user_role

Descrição: Disponibiliza a role do utilizador nos templates.

Print da função:

```
# -----
# CONTEXT PROCESSOR
#
@app.context_processor
def coleta_user_role():
    """
    Disponibiliza a role do usuário logado para todos os templates
    """
    return dict(user_role=session.get("user_role"))
```

Função: base

Descrição: Página inicial do sistema.

Print da função:

```
# -----
# ROTA BASE
#
@app.route("/")
def base():
    """
    Página inicial: redireciona para login se não estiver logado
    """
    if "user_id" not in session:
        return redirect(url_for("login"))
    return render_template("base.html")
```



Função: login

Descrição: Autenticação de utilizadores.

Print da função:

```
# -----
# LOGIN
# -----
@app.route("/login", methods=["GET", "POST"])
def login():
    """
    Login de usuários
    """
    if request.method == "POST":
        username = request.form["username"]
        password = request.form["password"]

        try:
            user = executar_query(
                "SELECT id, username, password, role FROM users WHERE username=%s",
                (username,),
                fetchone=True
            )
        except:
            flash("Erro ao acessar a base de dados.")
            return redirect(url_for("login"))

        if user and user["password"] == password:
            session["user_id"] = user["id"]
            session["username"] = user["username"]
            session["user_role"] = user["role"]
            session["password"] = user["password"]
            return redirect(url_for("base"))

        flash("Username ou password incorretos.")
        return redirect(url_for("login"))

    return render_template("users/login.html")
```



EFA T – PROGRAMADORA DE INFORMÁTICA

Função: **logout**

Descrição: Finaliza a sessão do utilizador.

Print da função:

```
# -----
# LOGOUT
#
@app.route("/logout")
def logout():
    """
    Limpa sessão e redireciona para login
    """
    session.clear()
    return redirect(url_for("login"))
```

Função: **home**

Descrição: Página principal após login.

Print da função:

```
# -----
# HOME
#
@app.route("/home")
def home():
    if "user_id" not in session:
        flash("Faça login primeiro!")
        return redirect(url_for("login"))
    return render_template("global/home.html")
```

Função: **minha_area**

Descrição: Área do cliente com dados, animais e consultas.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
-----  
MINHA ÁREA (CLIENTE)  
-----  
  
pp.route("/minha_area")  
f minha_area():  
    """  
    Mostra área do cliente  
    """  
    if "user_id" not in session:  
        flash("Faça login primeiro!")  
        return redirect(url_for("login"))  
  
    try:  
        user = executar_query(  
            "SELECT * FROM users WHERE id=%s",  
            (session["user_id"],),  
            fetchone=True  
        )  
  
        cliente = executar_query(  
            "SELECT * FROM clientes WHERE id=%s",  
            (user["cliente_id"],),  
            fetchone=True  
        )  
  
        animais = executar_query(  
            "SELECT * FROM animais WHERE cliente_id=%s",  
            (cliente["id"],),  
            fetchall=True  
        )  
  
        consultas = executar_query("""  
            SELECT c.*, a.nome AS animal_nome  
            FROM consultas c  
            JOIN animais a ON c.animal_id = a.id  
            WHERE a.cliente_id=%s  
        """ , (cliente["id"],), fetchall=True)  
  
    except:  
        flash("Erro ao carregar dados.")  
        return redirect(url_for("login"))  
  
    return render_template(  
        "cliente/minha_area.html",  
        user=user,  
        cliente=cliente,  
        animais=animais,  
        consultas=consultas  
    )
```



Função: **tabela_utilizadores**

Descrição: Lista todos os utilizadores.

Print da função:

```
# -----
# TABELAS
# -----
@app.route("/tabela_utilizadores")
def tabela_utilizadores():
    if "user_id" not in session:
        return redirect(url_for("login"))

    try:
        utilizadores = executar_query(
            "SELECT id, username, role, created_at FROM users ORDER BY id ASC",
            fetchall=True
        )
    except:
        flash("Erro ao carregar utilizadores.")
        return redirect(url_for("base"))

    return render_template("admin/tabela_utilizadores.html", utilizadores=utilizadores)
```



EFA T – PROGRAMADORA DE INFORMÁTICA

Função: **tabela_clientes**

Descrição: Lista todos os clientes.

Print da função:

```
@app.route("/tabela_clientes")
def tabela_clientes():
    """
    Lista todos os clientes
    """

    if "user_id" not in session:
        return redirect(url_for("login"))

    try:
        clientes = executar_query(
            "SELECT id, nome, email, telefone, morada, created_at FROM clientes ORDER BY id ASC"
            "fetchall=True"
        )
    except:
        flash("Erro ao carregar clientes.")
        return redirect(url_for("base"))

    return render_template("staff/tabela_clientes.html", clientes=clientes)
```

Função: **tabela_animais**

Descrição: Lista todos os animais.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
@app.route("/tabela_animais")
def tabela_animais():
    """
    Lista todos os animais
    """

    if "user_id" not in session:
        return redirect(url_for("login"))

    try:
        animais = executar_query(
            "SELECT id, nome, especie, raca, data_nascimento FROM animais ORDER BY id ASC",
            fetchall=True
        )
    except:
        flash("Erro ao carregar animais.")
        return redirect(url_for("base"))

    return render_template("staff/tabela_animais.html", animais=animais)
```

Função: **tabela_consultas**

Descrição: Lista todas as consultas.

Print da função:

```
@app.route("/tabela_consultas")
def tabela_consultas():
    """
    Lista todas as consultas
    """

    if "user_id" not in session:
        return redirect(url_for("login"))

    try:
        consultas = executar_query(
            "SELECT id, data_hora, motivo, notas, created_at FROM consultas ORDER BY id ASC",
            fetchall=True
        )
    except:
        flash("Erro ao carregar consultas.")
        return redirect(url_for("base"))

    return render_template("staff/tabela_consultas.html", consultas=consultas)
```



INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL, I.P.
CENTRO DE EMPREGO E FORMAÇÃO PROFISSIONAL DE BRAGA

EFA T – PROGRAMADORA DE INFORMÁTICA

Função: registrar_novo_cliente

Descrição: Insere um novo cliente na base de dados.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
# -----
# REGISTRAR NOVO CLIENTE
# -----
@app.route("/registrar_novo_cliente", methods=["GET", "POST"])
def registrar_novo_cliente():
    """
    Registra novo cliente e cria usuário automaticamente
    """

    if request.method == "POST":
        nome = request.form.get("nome")
        email = request.form.get("email")
        telefone = request.form.get("telefone")
        morada = request.form.get("morada")

        if not nome or not email or not telefone or not morada:
            flash("Preencha todos os campos!")
            return redirect(url_for("registrar_novo_cliente"))

    try:
        existe = executar_query(
            "SELECT id FROM clientes WHERE email=%s",
            (email,),
            fetchone=True
        )
        if existe:
            flash("Este cliente já está cadastrado!")
            return redirect(url_for("registrar_novo_cliente"))

        executar_query(
            "INSERT INTO clientes (nome, email, telefone, morada) VALUES (%s,%s,%s,%s)",
            (nome,email,telefone,morada),
            commit=True
    )
```



EFA T – PROGRAMADORA DE INFORMÁTICA

```
cliente = executar_query(  
    "SELECT id FROM clientes WHERE email=%s",  
    (email,),  
    fetchone=True  
)  
  
executar_query(  
    "INSERT INTO users (username, password, role, cliente_id) VALUES (%s,%s,%s,%s)",  
    (email, "123", "cliente", cliente["id"]),  
    commit=True  
)  
  
except:  
    flash("Erro ao registrar cliente.")  
    return redirect(url_for("registrar_novo_cliente"))  
  
    flash("Cliente e login criados com sucesso!")  
    return redirect(url_for("tabela_clientes"))  
  
return render_template("staff/registrar_novo_cliente.html")
```

Função: **registrar_novo_utilizador**

Descrição: Insere um novo utilizador a base de dados.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
# -----
# REGISTRAR NOVO USUÁRIO
# -----
@app.route("/registrar_novo_utilizador", methods=["GET", "POST"])
def registrar_novo_utilizador():
    .....
    Registra novo usuário (admin/staff)
    .....

    if request.method == "POST":
        username = request.form.get("username")
        password = request.form.get("password")
        role = request.form.get("role")

        if not username or not password or not role:
            flash("Preencha todos os campos!")
            return redirect(url_for("registrar_novo_utilizador"))

    try:
        existe = executar_query(
            "SELECT id FROM users WHERE username=%s",
            (username,),
            fetchone=True
        )
        if existe:
            flash("Este nome de usuário já está em uso!")
            return redirect(url_for("registrar_novo_utilizador"))

        executar_query(
            "INSERT INTO users (username, password, role) VALUES (%s,%s,%s)",
            (username,password,role),
            commit=True
        )

    except:
        flash("Erro ao registrar utilizador.")
        return redirect(url_for("registrar_novo_utilizador"))

    flash("Conta criada com sucesso!")
    return redirect(url_for("base"))

    return render_template("admin/registrar_novo_utilizador.html")
```

Função: editar_users

Descrição: Edita dados de um utilizador.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
# -----
# EDITAR UTILIZADOR
# -----
@app.route("/editar_utilizador/<int:id>", methods=["GET","POST"])
def editar_users(id):
    .....
    Edita dados de um usuário
    .....
    if "user_id" not in session:
        return redirect(url_for("login"))

    if request.method == "POST":
        try:
            executar_query(
                "UPDATE users SET username=%s, role=%s WHERE id=%s",
                (request.form["username"], request.form["role"], id),
                commit=True
            )
        except:
            flash("Erro ao atualizar utilizador.")
            return redirect(url_for("tabela_utilizadores"))

        return redirect(url_for("tabela_utilizadores"))

    try:
        usuarios = executar_query(
            "SELECT id, username, role FROM users WHERE id=%s",
            (id,),
            fetchone=True
        )
    except:
        flash("Erro ao carregar utilizador.")
        return redirect(url_for("tabela_utilizadores"))

    return render_template("admin/editar_users.html", usuarios=usuarios)
```



EFA T – PROGRAMADORA DE INFORMÁTICA

Função: novo_animal

Descrição: Insere um novo animal.

Print da função:

```
# -----
# NOVO ANIMAL
# -----
@app.route("/clientes/<int:cliente_id>/novo_animal", methods=["GET","POST"])
def novo_animal(cliente_id):
    """
    Permite registrar um novo animal para o cliente
    """
    try:
        cliente = executar_query(
            "SELECT * FROM clientes WHERE id=%s",
            (cliente_id,),
            fetchone=True
        )
        if not cliente:
            flash("Cliente não encontrado!")
            return redirect(url_for("tabela_clientes"))

        if request.method == "POST":
            executar_query(
                "INSERT INTO animais (cliente_id, nome, especie, raca, data_nascimento) VALUES (%s,%s,%s,%s,%s)",
                (
                    cliente_id,
                    request.form.get("nome"),
                    request.form.get("especie"),
                    request.form.get("raca"),
                    request.form.get("data_nascimento")
                ),
                commit=True
            )
            flash("Animal registrado com sucesso!")
            return redirect(url_for("tabela_animais"))

    except:
        flash("Erro ao registrar animal.")
        return redirect(url_for("tabela_animais"))

    return render_template("cliente/novo_animal.html", cliente=cliente)
```



Função: nova_consulta

Descrição: Insere uma nova consulta.

Print da função:

```
# -----
# NOVA CONSULTA
# -----
@app.route("/clientes/<int:cliente_id>/nova_consulta", methods=["GET","POST"])
def nova_consulta(cliente_id):
    """
    Permite registrar nova consulta para os animais do cliente
    """

    try:
        animais = executar_query(
            "SELECT * FROM animais WHERE cliente_id=%s",
            (cliente_id,),
            fetchall=True
        )
        if not animais:
            flash("Cliente não possui animais cadastrados!")
            return redirect(url_for("tabela_clientes"))

        if request.method == "POST":
            executar_query(
                "INSERT INTO consultas (animal_id, data_hora, motivo) VALUES (%s,%s,%s)",
                (
                    request.form.get("animal_id"),
                    request.form.get("data_hora"),
                    request.form.get("motivo")
                ),
                commit=True
            )
            flash("Consulta marcada com sucesso!")
            return redirect(url_for("tabela_consultas"))
    
```



Função: `deleta_cliente`

Descrição: Remove um cliente do sistema..

Print da função:

```
# -----
# DELETAR REGISTROS
# -----
@app.route("/deleta_cliente/<int:id>", methods=["POST"])
def deleta_cliente(id):
    """
    Deleta um cliente pelo ID
    """

    if "user_id" not in session:
        return redirect(url_for("login"))

    try:
        executar_query(
            "DELETE FROM clientes WHERE id=%s",
            (id,),
            commit=True
        )
    except:
        flash("Erro ao apagar cliente.")
        return redirect(url_for("tabela_clientes"))

    return redirect(url_for("tabela_clientes"))
```



Função: **deleta_utilizador**

Descrição: Remove um utilizador do sistema.

Print da função:

```
# -----
# DELETAR UTILIZADOR
# -----
@app.route("/deleta_utilizador/<int:id>", methods=["POST"])
def deleta_utilizador(id):
    if "user_id" not in session:
        return redirect(url_for("login"))

    try:
        executar_query(
            "DELETE FROM users WHERE id=%s",
            (id,),
            commit=True
        )
    except:
        flash("Erro ao apagar utilizador.")
        return redirect(url_for("tabela_utilizadores"))

    flash("Utilizador apagado com sucesso.")
    return redirect(url_for("tabela_utilizadores"))

# -----
```



Função: deleta_consulta

Descrição: Remove uma consulta do sistema.

Print da função:

```
@app.route("/deleta_consulta/<int:id>", methods=["POST"])
def deleta_consulta(id):
    """
    Deleta uma consulta pelo ID (somente admin)
    """

    if "user_id" not in session or session.get("user_role") != "admin":
        flash("Acesso não autorizado.")
        return redirect(url_for("login"))

    try:
        executar_query(
            "DELETE FROM consultas WHERE id=%s",
            (id,),
            commit=True
        )
        flash("Consulta apagada com sucesso.")
    except:
        flash("Erro ao apagar consulta.")
        return redirect(url_for("tabela_consultas"))

    return redirect(url_for("tabela_consultas"))
```

Função: editar_cliente

Descrição: Edita os dados de um cliente no sistema.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
# -----
# EDITAR DADOS
# -----
@app.route("/editar_cliente<int:id>", methods=["GET","POST"])
def editar_cliente(id):
    """
    Edita dados do cliente
    """

    if "user_id" not in session:
        return redirect(url_for("login"))

    if request.method == "POST":
        try:
            executar_query(
                "UPDATE clientes SET nome=%s,email=%s,telefone=%s,morada=%s WHERE id=%s",
                (
                    request.form["nome"],
                    request.form["email"],
                    request.form["telefone"],
                    request.form["morada"],
                    id
                ),
                commit=True
            )
        except:
            flash("Erro ao atualizar cliente.")
            return redirect("/")
    else:
        cliente = executar_query(
            "SELECT id,nome,email,telefone,morada FROM clientes WHERE id=%s",
            (id,),
            fetchone=True
        )
        except:
            flash("Erro ao carregar cliente.")
            return redirect("/")

    return render_template("admin/editar_clientes.html", cliente=cliente)
```



INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL, I.P.
CENTRO DE EMPREGO E FORMAÇÃO PROFISSIONAL DE BRAGA

EFA T – PROGRAMADORA DE INFORMÁTICA

Função: editar_animal

Descrição: Edita um animal no sistema.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
@app.route("/editar_animal/<int:id>", methods=["GET","POST"])
def editar_animal(id):
    """
    Edita dados de um animal
    """
    if "user_id" not in session:
        return redirect(url_for("login"))

    if request.method == "POST":
        try:
            executar_query(
                "UPDATE animais SET nome=%s,especie=%s,raca=%s,data_nascimento=%s WHERE id=%s",
                (
                    request.form["nome"],
                    request.form["especie"],
                    request.form["raca"],
                    request.form["data_nascimento"],
                    id
                ),
                commit=True
            )
        except:
            flash("Erro ao atualizar animal.")
            return redirect("/tabela_animais")

    return redirect("/tabela_animais")
```

```
try:
    animal = executar_query(
        "SELECT id,nome,especie,raca,data_nascimento FROM animais WHERE id=%s",
        (id,),
        fetchone=True
    )
except:
    flash("Erro ao carregar animal.")
    return redirect("/tabela_animais")

return render_template("staff/editar_animais.html", animal=animal)
```

Função: **editar_consulta**

Descrição: Edita uma consulta no sistema.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
@app.route("/editar_consulta/<int:id>", methods=["GET", "POST"])
def editar_consulta(id):
    """
    Edita dados de uma consulta
    """
    if "user_id" not in session:
        return redirect(url_for("login"))

    if request.method == "POST":
        try:
            executar_query(
                "UPDATE consultas SET data_hora=%s,motivo=%s,notas=%s WHERE id=%s",
                (
                    request.form["data_hora"],
                    request.form["motivo"],
                    request.form["notas"],
                    id
                ),
                commit=True
            )
        except:
            flash("Erro ao atualizar consulta.")
            return redirect("/tabela_consultas")

    return redirect("/tabela_consultas")
```

```
try:
    consulta = executar_query(
        "SELECT id,data_hora,motivo,notas FROM consultas WHERE id=%s",
        (id,),
        fetchone=True
    )
except:
    flash("Erro ao carregar consulta.")
    return redirect("/tabela_consultas")

return render_template("staff/editar_consulta.html", consulta=consulta)
```

Função: [editar_users](#)

Descrição: Edita um utilizador no sistema.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
@app.route("/editar_utilizador<int:id>", methods=["GET","POST"])
def editar_users(id):
    """
    Edita dados de um usuário
    """
    if "user_id" not in session:
        return redirect(url_for("login"))

    if request.method == "POST":
        try:
            executar_query(
                "UPDATE users SET username=%s, role=%s WHERE id=%s",
                (
                    request.form["username"],
                    request.form["role"],
                    id
                ),
                commit=True
            )
        except:
            flash("Erro ao atualizar usuário.")
            return redirect("/tabela_utilizadores")

    return redirect("/tabela_utilizadores")

    try:
        usuarios = executar_query(
            "SELECT id,username,role FROM users WHERE id=%s",
            (id,),
            fetchone=True
        )
    except:
        flash("Erro ao carregar usuário.")
        return redirect("/tabela_utilizadores")

    return render_template("admin/editar_users.html", usuarios=usuarios)
```

Função: [trocar_password](#)

Descrição: Altera a senha do utilizador.

Print da função:



EFA T – PROGRAMADORA DE INFORMÁTICA

```
@app.route("/trocar_password/<int:id>", methods=["GET","POST"])
def trocar_password(id):
    """
    Permite trocar senha do usuário
    """

    if "user_id" not in session:
        return redirect(url_for("login"))

    if request.method == "POST":
        try:
            executar_query(
                "UPDATE users SET username=%s,password=%s,role=%s WHERE id=%s",
                (
                    request.form["username"],
                    request.form["password"],
                    request.form["role"],
                    id
                ),
                commit=True
            )
        except:
            flash("Erro ao atualizar usuário.")
            return redirect("/")
        return redirect("/")
    try:
        usuarios = executar_query(
            "SELECT id,username,role FROM users WHERE id=%s",
            (id,),
            fetchone=True
        )
    except:
        flash("Erro ao carregar usuário.")
        return redirect("/")
    return render_template("users/trocar_password.html", usuarios=usuarios)
```