

# Programação orientada a objetos

Instituto Politécnico do Cávado e do Ave  
Escola Técnica Superior Profissional  
CTeSP – Tecnologia e Inovação Informática  
Ano Letivo: 2021/2022



## Relatório Trabalho Prático

Ludgero Miguel Simões – 23135

Diogo Manuel Marques - 23000

João Gabriel Albergaria - 23013

Joana Freitas Pimenta – 22999

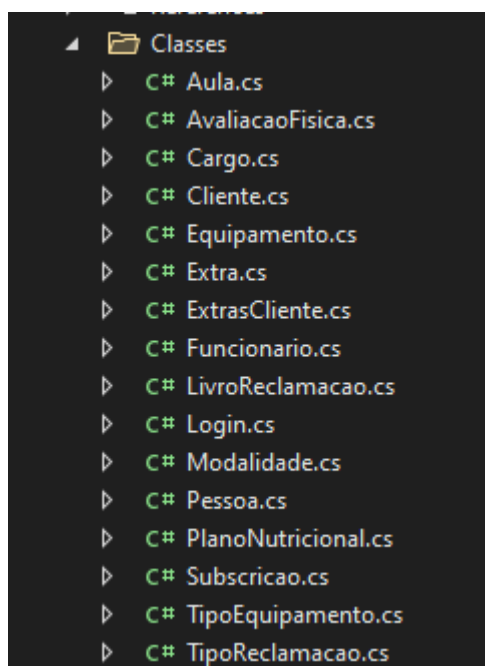
## INTRODUÇÃO

Neste trabalho prático, foi nos proposto realizar um projeto para a gestão do ginásio aonde seria conciliado com a outra disciplina de análise e modelação de software para implementar as soluções e as funcionalidades do nosso projeto.

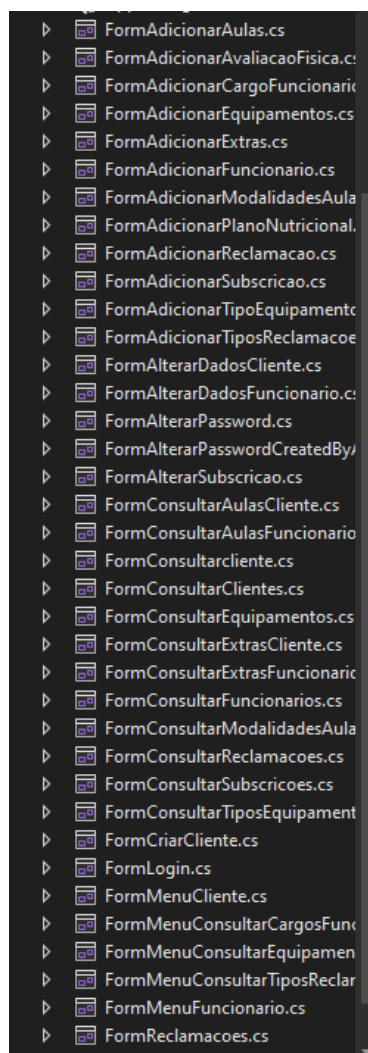
## EXPLICAÇÃO DO PROJETO

No nosso projeto temos dois tipos de cargos, o de funcionário e o de cliente e conforme o login, através do username conseguimos determinar qual o seu cargo e apresentar o menu correspondente.

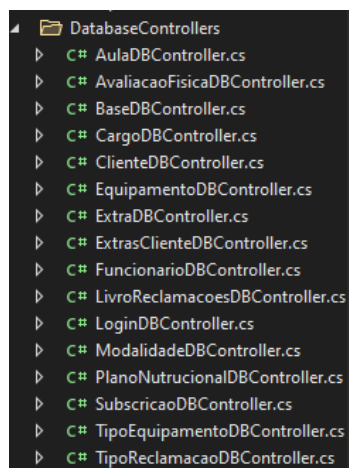
As nossas classes:



Os nossos formulários:



Controladores da database:



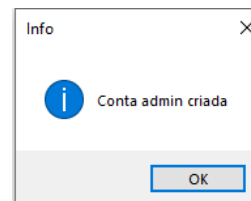
O programa começa com a apresentação do menu de Login e caso não exista nenhum cargo de recursos humanos ele vai criar uma conta com esse cargo. Na primeira vez que é executado o programa, este vai criar um novo funcionário e consequentemente a conta de login do mesmo, como é demonstrado no código abaixo:

```
login = new Login(username, password, "funcionario_" + funcionario.id, 1, 1);  
login.password = login.encryptPassword(password);  
if (!login.inserir()) return false;  
MessageBox.Show("Conta " + username + " criada", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);
```

Login default ( 1ª execução ):

Username: admin

Password: passwordAdmin



Depois de as credenciais do login serem aceites, o programa vai verificar se a conta foi criada por um admin, e dependendo do tipo do login é redirecionado para o formulário de funcionário ou de cliente. No caso 'createdbyAdmin == 1' será pedido uma nova password, redirecionado posteriormente para o menu de funcionário. Todas as senhas são encriptadas através da função "HashPassword".

```

if (login.createdByAdmin == 1) {
    this.Hide();
    FormAlterarPasswordCreatedByAdmin formAlterarPasswordCreatedByAdmin = new FormAlterarPasswordCreatedByAdmin();
    formAlterarPasswordCreatedByAdmin.Closed += (s, args) => this.Close();
    formAlterarPasswordCreatedByAdmin.Show();
} else if (Program.tipoConta == "cliente") {
    this.Hide();
    FormMenuCliente formMenuCliente = new FormMenuCliente();
    formMenuCliente.Closed += (s, args) => this.Close();
    formMenuCliente.Show();
} else if (Program.tipoConta == "funcionario") {
    this.Hide();
    FormMenuFuncionario formMenuFuncionario = new FormMenuFuncionario();
    formMenuFuncionario.Closed += (s, args) => this.Close();
    formMenuFuncionario.Show();
}

```

## Alterar Password

Devido a tua conta ser criada por um funcionario tens de alterar a password

Nova password

Repetir password

Menu de funcionário( caso a conta de login esteja associada a um funcionário ou a conta for criada por um admin )

## Menu Funcionário

Consultar Clientes	Consultar Funcionarios	Consultar Tipos de Equipamento	Consultar Aulas
Consultar Subscrições	Consultar Cargos Funcionario	Consultar Equipamentos	Consultar Modalidades Aula
Consultar Reclamações	Consultar Tipos Reclamação	Consultar Extras	Alterar Password
Alterar Dados	Terminar Sessão		

**Consulta de clientes:**

✕

Consultar Clientes

Consultar

Adicionar

Adicionar Avaliação física

Adicionar Plano Nutricional

	Id	Primeiro Nome	Ultimo Nome	Data de Nascimento	Nif	Genero	Telefone	Email
--	----	---------------	-------------	--------------------	-----	--------	----------	-------

< >

Voltar

Caso não haja nenhuma subscrição não será possível criar nenhum cliente:

Erro

✕

Não é possível abrir o menu de criar cliente pois não existe nenhum plano

OK

Caso exista pelo menos uma subscrição será submetido para a ficha da criação do cliente:

✕

## Consultar Clientes

Consultar

Adicionar

Adicionar Avaliação física

Adicionar Plano Nutricional

	Id	Primeiro Nome	Ultimo Nome	Data de Nascimento	Nif	Genero	Telefone	Email
--	----	---------------	-------------	--------------------	-----	--------	----------	-------

<

>

Voltar

✕

## Criar Cliente

Dados do Cliente

Primeiro Nome

Ludgero

Ultimo Nome

Simões

Data de Nascimento

27 de abril de 2000

NIF

250951053

Género

☒ Masculino ☐ Femenino

Telefone

9143914

Email

lmcms20

Morada

Rua dom

Foto

C:\Users\

Subscrição

Id: 1-Nom

Fim Subscrição

25 de agosto de 2022

Sucesso

✕

Conta criada com sucesso

OK

Dados do Login

Username

LuD

Password

••••

Voltar

Adicionar

Posteriormente podemos consultar o cliente:

## Consultar Cliente

**Dados Cliente**

ID: 1  
 Nome: Ludgero Simões  
 Data Nascimento: 27/4/2000  
 NIF: 250951053  
 Genero: Masculino  
 Telefone: 914391401  
 Email: lmcms2000@gmail.com  
 Morada: Rua dom pedro V, 33

**Dados Subscrição**

ID: 1  
 Nome: Livre  
 Preço: 40

**Aulas:**

Modalidade	Nº Sala	Maximos de alunos	Dia da semana

**Avaliações Física**

Id	Peso	Tamanho	Gordura

**Extras**

Nome	Preço	Quantidade

**Plano Nutricional**

Dia Semana	Pequeno Almoço	Lanche da Manhã	Almoço

Podemos atribuir uma avaliação física a um cliente selecionando e clicando no botão de "Adicionar Avaliação Física":

## Avaliação Física

Peso

75

Tamanho

170

Gordura

20

Massa Muscular

40

Voltar

Adicionar

Também podemos atribuir um plano nutricional clicando no botão de "Adicionar Plano Nutricional":

## Plano Nutricional

Dia Semana

Segunda

Pequeno Almoço

Ovos e pão

Lanche da Manhã

Fruta

Almoço

Frango assado

Lanche da Tarde

iogurte

Jantar

Peixe grelhado

Ceia

Sementes

Voltar

Adicionar



## Consulta de Funcionários:

Cela

✕

Consultar Funcionarios

Adicionar

	Id	Primeiro Nome	Ultimo Nome	Data de Nascimento	Nif	Genero	Telefone	Email
▶	1	Admin	System	10/10/2003	999999999	Masculino	999999999	admin

< >

Voltar

Formulário da adição de um novo funcionário:

✕

Adicionar Funcionário

Primeiro Nome

Morada

Último nome

Cargo

Data Nascimento

25 de junho de 2022

Fim do contrato

25 de junho de 2022

NIF

Início do turno

:

Género

☒ Masculino ☐ Femenino

Fim do Turno

:

Telefone

Salário

Email

Foto

default

Dados do Login

Username

Password

Voltar

Adicionar

**Consultar Tipos de Equipamento:**

Na adição de um novo equipamento temos o seguinte formulário:

Adicionar tipo de equipamento

Nome do tipo de equipamento

Informação

×

Tipo de equipamento criado com sucesso!

OK

**Consultar Aulas:**

Na inserção de uma nova aula:

Adicionar Aulas

Modalidade

Máximo de alunos

Professor

Dia de semana

Número de sala

Hora  :

***Consultar Subscrições:***

Inserção de uma nova subscrição:

Adicionar Subscrição

Nome

Preço

Voltar

Adicionar

***Consultar Cargos Funcionário:***

Inserção de um novo cargo de funcionário:

Adicionar Cargo Funcionario

Nome

Voltar

Adicionar

## Consultar Equipamentos:

Inserção de um novo equipamento:

### Adicionar equipamentos

Nome

Quantidade

Tipo de equipamento

Funcionario

Remoção de um equipamento é preciso o selecionar e depois clicar no botão de remover:

### Consultar Equipamentos

	Id	Nome	Quantidade	Tipo Equipamento	Funcionario
▶	1	Supino com barra...	2	Supino com barra	Admin System

Informação

Equipamento removido com sucesso

### **Consultar Modalidades Aula:**

Na inserção de uma nova modalidade:

Consultar aulas

Adicionar Modalidades de Aulas

Nome MMA

Voltar

Adicionar

Informação

Modalidade introduzida com sucesso

OK

### **Consultar Reclamações:**

Aqui podemos ver as reclamações criadas pelo cliente.

### **Consultar Tipos Reclamação:**

Na inserção de um novo tipo de reclamação:

Adicioar tipo Reclamação

Nome Limpezas

Voltar

Adicionar

### **Consultar Extras:**

Aqui podemos consultar as reclamações introduzidas pelo cliente.

***Inserção dos extras:***

### Adicionar extras

Nome

Preço

***Alterar Password:***

### Alterar Password

Nova password

Repetir password

***Alterar Dados:***

### Alterar dados Funcionario

Primeiro Nome	<input type="text" value="Admin"/>	Morada	<input type="text" value="System"/>
Último nome	<input type="text" value="System"/>	Email	<input type="text" value="admin@system.com"/>
Data Nascimento	<input type="text" value="10 de outubro de 2003"/>	Género	<input checked="" type="radio"/> Masculino <input type="radio"/> Femenino
NIF	<input type="text" value="999999999"/>	Telefone	<input type="text" value="999999999"/>

[illegible]

## Consultar Extras:

×

Extras

	Id	Nome	Preço

Voltar

Quantidade extra

Contratar Extra

## Alterar Subscrições:

×

Alterar Subscrição

Subscrição

Fim Subscrição

Voltar

Alterar

## Reclamações:

Na criação de uma nova reclamação:

×

Adicionar Reclamação

Descrição

Tipo Reclamação

Voltar

Adicionar



## Consultar Aulas:

×

Consultar aulas

Id	Modalidade	Nº Sala	Maximo Alunos	Dia semana	Hora	Professor

Inscriver Aula

Desinscrever Aula

Voltar

## Alterar Dados:

×

Alterar Dados Cliente

Primeiro Nome

Ludgero

Ultimo Nome

Simões

Data de Nascimento

27 de abril de 2000

NIF

250951053

Género

☒ Masculino ☐ Femenino

Telefone

914391401

Email

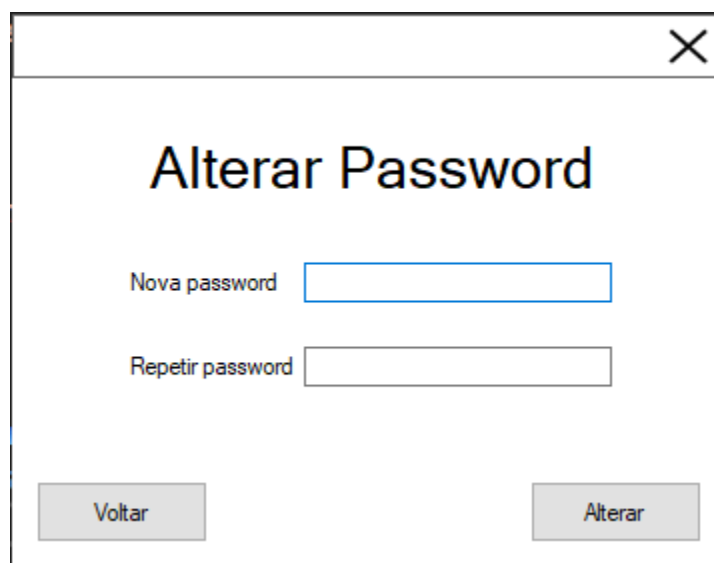
lmcms2000@gmail.com

Morada

Rua dom pedro V, 33

Voltar

Alterar

**Alterar Password:**

A screenshot of a web application dialog box titled "Alterar Password". The dialog box has a close button (X) in the top right corner. Inside the dialog, the title "Alterar Password" is centered. Below the title, there are two input fields: "Nova password" and "Repetir password". At the bottom of the dialog, there are two buttons: "Voltar" (Back) and "Alterar" (Change).

**Terminar sessão:**

É perguntado se tem mesmo a certeza se pretende sair da conta, e clicando no "Yes", o utilizador é redirecionado para o menu de Login.

## Porções de código explicativas do programa:

Poderia mostrar todas as classes, mas vou apresentar apenas a **classe de cargo**, que o resto é mais do mesmo e ficaria chato.

```
namespace Ginasio.Classes {
    26 references
    internal class Cargo {
        private int _id;
        private string _nome;
        private string _nomeSistema;
        private Funcionario[] _funcionarios;

        2 references
        public Cargo(string nome) {
            this._nome = nome;
            this._nomeSistema = nome.ToLower().Replace(" ", "_");
        }

        3 references
        public Cargo(int id, string nome, string nomeSistema) {
            this._id = id;
            this._nome = nome;
            this._nomeSistema = nomeSistema;
        }

        4 references
        public int id {
            get { return this._id; }
        }

        4 references
        public string nome {
            get { return this._nome; }
        }
    }
    7 references
}
```

```
0 references
public Funcionario[] funcionarios {
    get { return this._funcionarios; }
}

0 references
public bool getFuncionariosCargo() {
    bool status = true;

    try {
        this._funcionarios = new FuncionarioDBController().getFuncionariosCargo(this._id);
    } catch {
        status = false;
    }

    return status;
}

2 references
public bool inserir() {
    int id = new CargoDBController().inserir(this);

    if (id == -1) return false;

    this._id = id;

    return true;
}

0 references
public bool remover() {
    return new CargoDBController().remover(this._id);
}
```

Na classe cargo temos os atributos dele como o id, nome e um array da classe do tipo funcionário em "private". Também temos construtores para atribuir à sua classe os seus atributos através da passagem dos parâmetros.

Também temos métodos para consultar, inserção e remoção os funcionários por cargo chamando o controller da database (CargoDBController).

### Controller Base

```
abstract class BaseDBController {
    protected MySqlConnection connection;
    protected MySqlDataReader reader;
    protected MySqlCommand command;
    protected MySqlDataAdapter adapter;
    protected string sql;

    88 references
    protected MySqlConnection DBConn() {
        connection = new MySqlConnection("Server=localhost;Database=ginasio;Uid=root;Pwd=;");
        return connection;
    }

    99+ references
    protected void closeDB() {
        connection.Close();
        connection = null;
        command = null;
    }
}
```

```
31 references
protected int getNumRegistrosDB(string tableName) {
    int nRegistros;

    try {
        connection = DBConn();

        sql = "select count(1) as nRegistros from " + tableName;

        command = new MySqlCommand(sql, connection);

        connection.Open();

        reader = command.ExecuteReader();

        if (reader != null && reader.HasRows && reader.Read()) {
            nRegistros = Convert.ToInt32(reader["nRegistros"]);
        } else {
            nRegistros = 0;
        }

    } catch {
        nRegistros = 0;
    } finally {
        closeDB();
    }

    return nRegistros;
}
```

Todos os outros controladores vão derivar deste controlador "pai", com as definições para a manipulação da nossa base de dados.

## Controller para a database na parte do cargo:

```

namespace Ginasio.DatabaseControllers {
    7 references
    internal class CargoDBController : BaseDBController {
        1 reference
        public int inserir(Cargo cargo) {
            int id;

            try {
                connection = DBConn();

                sql = "INSERT INTO cargo(nome, nomeSistema) VALUES(@nome, @nomeSistema)";

                command = new MySqlCommand(sql, connection);
                command.Parameters.AddWithValue("@nome", cargo.nome);
                command.Parameters.AddWithValue("@nomeSistema", cargo.nomeSistema);

                connection.Open();

                if (command.ExecuteNonQuery() == 1) {
                    command = null;

                    sql = "SELECT LAST_INSERT_ID() as 'id' FROM 'cargo' LIMIT 1";

                    command = new MySqlCommand(sql, connection);

                    reader = command.ExecuteReader();

                    if (reader != null && reader.HasRows && reader.Read()) {
                        id = Convert.ToInt32(reader["id"]);
                    } else {
                        id = -1;
                    }
                } else {
                    id = -1;
                }
            } catch {
                id = -1;
            } finally {
                closeDB();
            }

            return id;
        }

        1 reference
        public bool remover(int idCargo) {
            bool status;

            try {
                connection = DBConn();

                sql = "DELETE FROM cargo WHERE id = @id";

                command = new MySqlCommand(sql, connection);
                command.Parameters.AddWithValue("@id", idCargo);

                connection.Open();

                if (command.ExecuteNonQuery() == 1) status = true;
                else status = false;
            } catch {
                status = false;
            } finally {
                closeDB();
            }

            return status;
        }
    }
}

```

```

public Cargo getById(int idCargo) {
    Cargo cargo = null;

    try {
        connection = DBConn();

        sql = "SELECT * FROM cargo WHERE id = @id";

        command = new MySqlCommand(sql, connection);
        command.Parameters.AddWithValue("@id", idCargo);

        connection.Open();

        reader = command.ExecuteReader();

        if (reader != null && reader.HasRows && reader.Read()) {
            int id;
            string nome, nomeSistema;

            id = Convert.ToInt32(reader["id"]);
            nome = Convert.ToString(reader["nome"]);
            nomeSistema = Convert.ToString(reader["nomeSistema"]);

            cargo = new Cargo(id, nome, nomeSistema);
        }
    } catch (Exception ex) {
        closeDB();
        throw ex;
    } finally {
        closeDB();
    }

    return cargo;
}

```

```

public Cargo[] getAll() {
    Cargo[] cargos = null;
    int nRows = getNumRegistrosDB("cargo"), i = 0;

    try {
        connection = DBConn();

        sql = "SELECT * FROM cargo";

        command = new MySqlCommand(sql, connection);

        connection.Open();

        reader = command.ExecuteReader();

        cargos = new Cargo[nRows];

        if (reader.HasRows) {
            while (reader.Read()) {
                int id;
                string nome, nomeSistema;

                id = Convert.ToInt32(reader["id"]);
                nome = Convert.ToString(reader["nome"]);
                nomeSistema = Convert.ToString(reader["nomeSistema"]);

                cargos[i] = new Cargo(id, nome, nomeSistema);
                i++;
            }
        }
    } catch (Exception ex) {
        closeDB();
        throw ex;
    } finally {
        closeDB();
    }
}

```

```

        return cargos;
    }

    2 references
    public Cargo[] searchByNomeSistema(string nomeSistema) {
        Cargo[] cargos = null;
        int nRows = getNumRegistrosDB("cargo"), i = 0;

        try {
            connection = DBConn();

            sql = "SELECT * FROM cargo WHERE nomeSistema LIKE @nomeSistema";

            command = new MySqlCommand(sql, connection);
            command.Parameters.AddWithValue("@nomeSistema", nomeSistema+"%");

            connection.Open();

            reader = command.ExecuteReader();

            cargos = new Cargo[nRows];

            if (reader.HasRows) {
                while (reader.Read()) {
                    int id;
                    string nome, nomeSistema;

                    id = Convert.ToInt32(reader["id"]);
                    nome = Convert.ToString(reader["nome"]);
                    nomeSistema = Convert.ToString(reader["nomeSistema"]);

                    cargos[i] = new Cargo(id, nome, nomeSistema);
                    i++;
                }
            }
            catch (Exception ex) {
                closeDB();
                throw ex;
            }
            finally {
                closeDB();
            }
        }

        return cargos;
    }

```

Através deste controller conseguimos manipular a parte correspondente aos cargos, através de vários métodos que interagem diretamente com a base de dados.

### Formulário para a consulta dos cargos dos funcionários:

```

1 reference
private void FormMenuConsultarCargosFuncionario_Load(object sender, EventArgs e) {
    Cargo[] cargos = null;

    try {
        cargos = new CargoDBController().getAll();
    } catch {
        MessageBox.Show("Ocorreu algum erro, tenta novamente mais tarde", "Erro", MessageBoxButtons.OK);
    }

    dgvCargos.Columns.Add("id", "Id");
    dgvCargos.Columns.Add("nome", "Nome");

    foreach (Cargo cargo in cargos) {
        dgvCargos.Rows.Add(cargo.id, cargo.nome);
    }
}

```

## Formulário para a criação de um novo cargo:

```
1 reference
private void btnAdicionar_Click(object sender, EventArgs e) {
    if (txtNome.Text == String.Empty) {
        MessageBox.Show("Tens de fornecer o nome do cargo", "Aviso", MessageBoxButtons.OK);
        txtNome.Focus();
        return;
    }

    Cargo cargo = new Cargo(txtNome.Text);
    Cargo[] cargosDB = null;

    try {
        cargosDB = new CargoDBController().searchByNomeSistema(cargo.nomeSistema);
    } catch {
        MessageBox.Show("Ocorreu algum erro, tenta novamente mais tarde", "Erro", MessageBoxButtons.OK);
        return;
    }

    if (cargosDB != null && cargosDB.Length > 0 && cargosDB[0] != null) {
        MessageBox.Show("Ja existe um cargo com esse nome!", "Erro", MessageBoxButtons.OK);
        txtNome.Text = String.Empty;
        txtNome.Focus();
        return;
    }

    if (cargo.inserir()) {
        MessageBox.Show("Cargo criado com sucesso", "Informação", MessageBoxButtons.OK);
        txtNome.Text = String.Empty;
    } else {
        MessageBox.Show("Ocorreu algum erro a criar o cargo, tenta novamente mais tarde", "Erro", MessageBoxButtons.OK);
    }
}
```

```
public bool inserir() {
    int id = new CargoDBController().inserir(this);

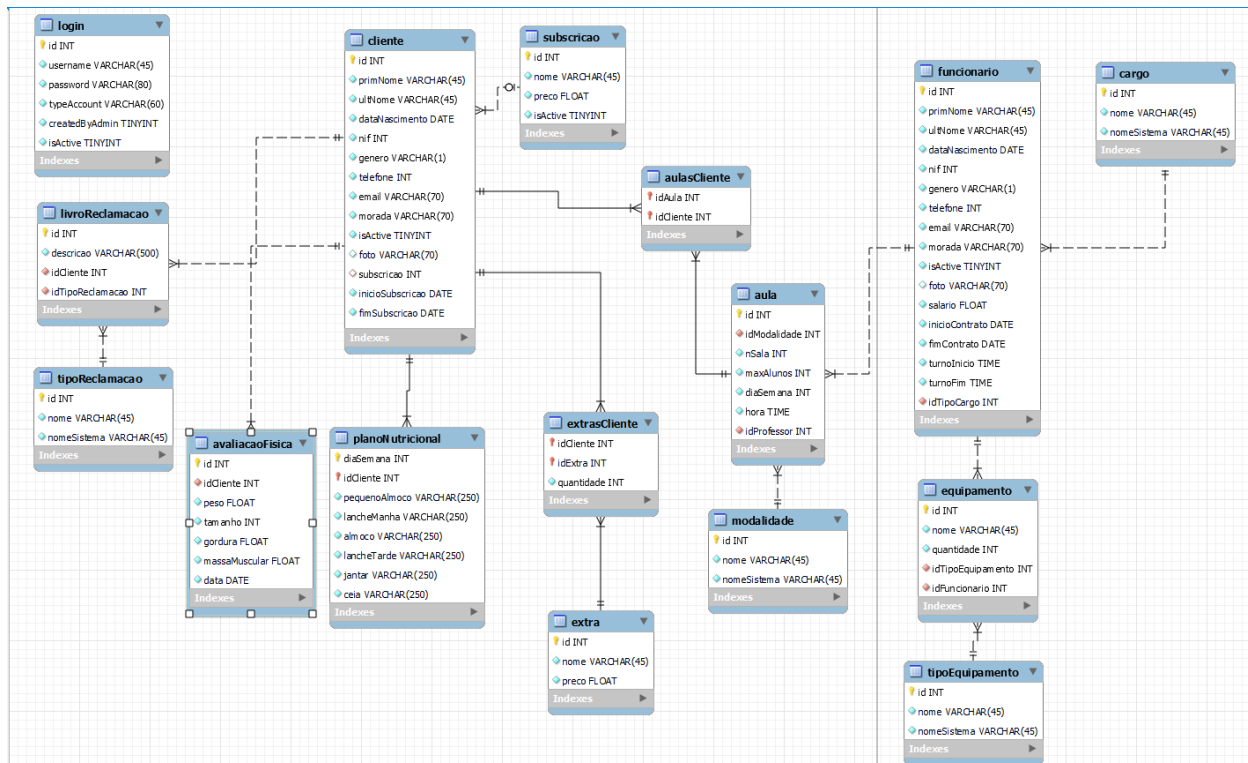
    if (id == -1) return false;

    this._id = id;

    return true;
}
```



## ESQUEMA DE SQL:



## CONCLUSÃO

Penso que alcançamos todos os objetivos pretendidos neste projeto. Apesar da complexidade do mesmo procuramos desde início aproximar o trabalho o mais possível de um caso real. Ao longo do projeto, fomos desenvolvendo as nossas capacidades, conseguindo pôr em prática todos os conhecimentos adquiridos em contexto de aula finalizando com um projeto que nos orgulhamos e conseguimos retirar aprendizagens e um software com bastante qualidade.

**Bem-hajam!**