


	<i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i>			
Trabalho de Arquitetura de Software				
Documento de Arquitetura				
Código/Sigla:	CGS	Nome do Projeto:	CG Study - Busca de Aulas Particulares	

CG Study - Busca de Aulas Particulares

Documento de Arquitetura



Quarto Semestre / 2020

Disciplina: Arquitetura de Software
Prof. Bianca Pedrosa
Caio Mota, Gabriel Baron

	<p><i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i></p>	
<p align="center">Trabalho de Arquitetura de Software</p>		
<p align="center">Documento de Arquitetura</p>		
<p>Código/Sigla:</p>	<p>CGS</p>	<p>Nome do Projeto: CG Study - Busca de Aulas Particulares</p>

Sumário

1. Introdução	3
2. Metas e Restrições de Arquitetura	3
2.1 Metas	3
2.2 Restrições	3
3. Visão de Casos de Uso	3
4. Visão Lógica	3
5. Visão de Dados	3
6. Visão de Implementação	3
7. Referências	4

	<p><i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i></p>	
<p align="center">Trabalho de Arquitetura de Software</p>		
<p align="center">Documento de Arquitetura</p>		
<p>Código/Sigla:</p>	<p>CGS</p>	<p>Nome do Projeto: CG Study - Busca de Aulas Particulares</p>

1. Introdução

A demanda de aulas online vem crescendo bastante nesses últimos meses. E é muito perceptível a dificuldade que se é pra adaptar-se à essa nova forma de aula tanto para os alunos quanto para os professores. Dessa forma surge a oportunidade de se usar a tecnologia para facilitar o arranjo das aulas e a comunicação professor-aluno. Com a utilização do nosso Software (GC Study), professores e alunos podem organizar com mais clareza suas aulas, a fim de tornar mais fácil essa nova maneira de aprendizado.

2. Metas e Restrições de Arquitetura

2.1 Metas

2.1.1 [Meta 1] - Tempo de resposta

As interações com o sistema não deve ultrapassar mais de dez segundos.

2.1.2 [Meta 2] - Interface de fácil entendimento

As interfaces do sistema devem ser de fácil entendimento, para que os usuário não tenham dificuldades de uso, botões com funcionalidades claras.

2.1.3 [Meta 3] - Flexível



O sistema deve ser desenvolvido de uma forma que seja fácil fazer alterações, melhorias ou remoção posteriores.

2.1.4 [Meta 4] - Usar Padrões de projeto

O sistema deve ser desenvolvido utilizando 3 padrões de projetos.

2.1.5 [Meta 5] - Usar Conceitos de Programação Orientada a Objetos

O sistema deve implementado usando conceitos de programação orientada a objetos.

	Instituto Federal de São Paulo – Campus Campinas Arquitetura de Software 4º Semestre - 2019	
Trabalho de Arquitetura de Software		
Documento de Arquitetura		
Código/Sigla:	CGS	Nome do Projeto: CG Study - Busca de Aulas Particulares

2.2 Restrições

2.2.1 [Restrição 1] - Implementação

O sistema deverá ser desenvolvido utilizando o NetBeans, SceneBuilder e MySQL WorkBench.

2.2.2 [Restrição 2] - Cadastro de Usuário

Para acessar o sistema o usuário deve realizar um cadastro com as informações necessárias para cada tipo de usuário (Aluno ou Professor).

2.2.3 [Restrição 3] - Login

O sistema deve ter um login para controle de quem está acessando o sistema.

2.2.4 [Restrição 4] - Usar Padrão DAO

O sistema deve ser desenvolvido utiliza o padrão DAO para a parte de banco de dados com o Java.

2.2.5 [Restrição 5] - Usar Padrão Singleton

O sistema deve ser desenvolvido usando o padrão Singleton e mantendo os métodos com funções únicas.

2.2.6 [Restrição 6] - Usar Padrão MVC

O sistema deve ser desenvolvido usando o padrão MVC para a organização do código.



3. Visão de Casos de Uso

Ator: Usuário

- Cadastrar user
- Editar Perfil
- Verifica Aulas Marcadas

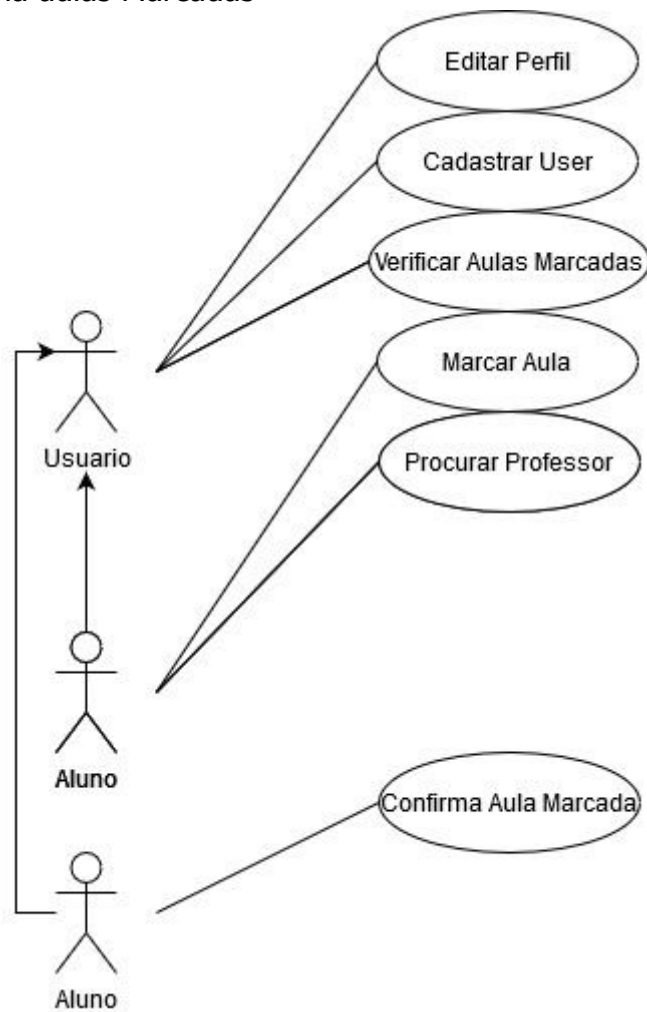
Ator: Aluno


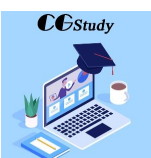
- Marcar Aula
- Procura Professor

	<p><i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i></p>	
<p align="center">Trabalho de Arquitetura de Software</p>		
<p align="center">Documento de Arquitetura</p>		
<p>Código/Sigla:</p>	<p>CGS</p>	<p>Nome do Projeto: CG Study - Busca de Aulas Particulares</p>

Ator: Professor

- Confirma aulas Marcadas



	<i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i>			
Trabalho de Arquitetura de Software				
Documento de Arquitetura				
Código/Sigla:	CGS	Nome do Projeto:	CG Study - Busca de Aulas Particulares	

4. Visão Lógica

Model: Classes modelo da aplicação usada para manipulação de dados.

View: Telas da aplicação, realizam interação com o usuário.

Controller: Classes que controlam as views e models.

Images: Pacote com todas as imagens utilizadas na aplicação.

Connection: Classes de conexão no banco de dados.

DAO: Classe que contém os comando SQL do CRUD.

Principais funcionalidades: A aplicação oferece ao usuário a pesquisa por professores, agendamento de horários.

Alunos: A classe Aluno com o AlunoDAO realiza operações CRUD e transmite para o BD para que possam ser armazenados.

Professores: A classe Professor também realiza operações CRUD e transmite para a BD.

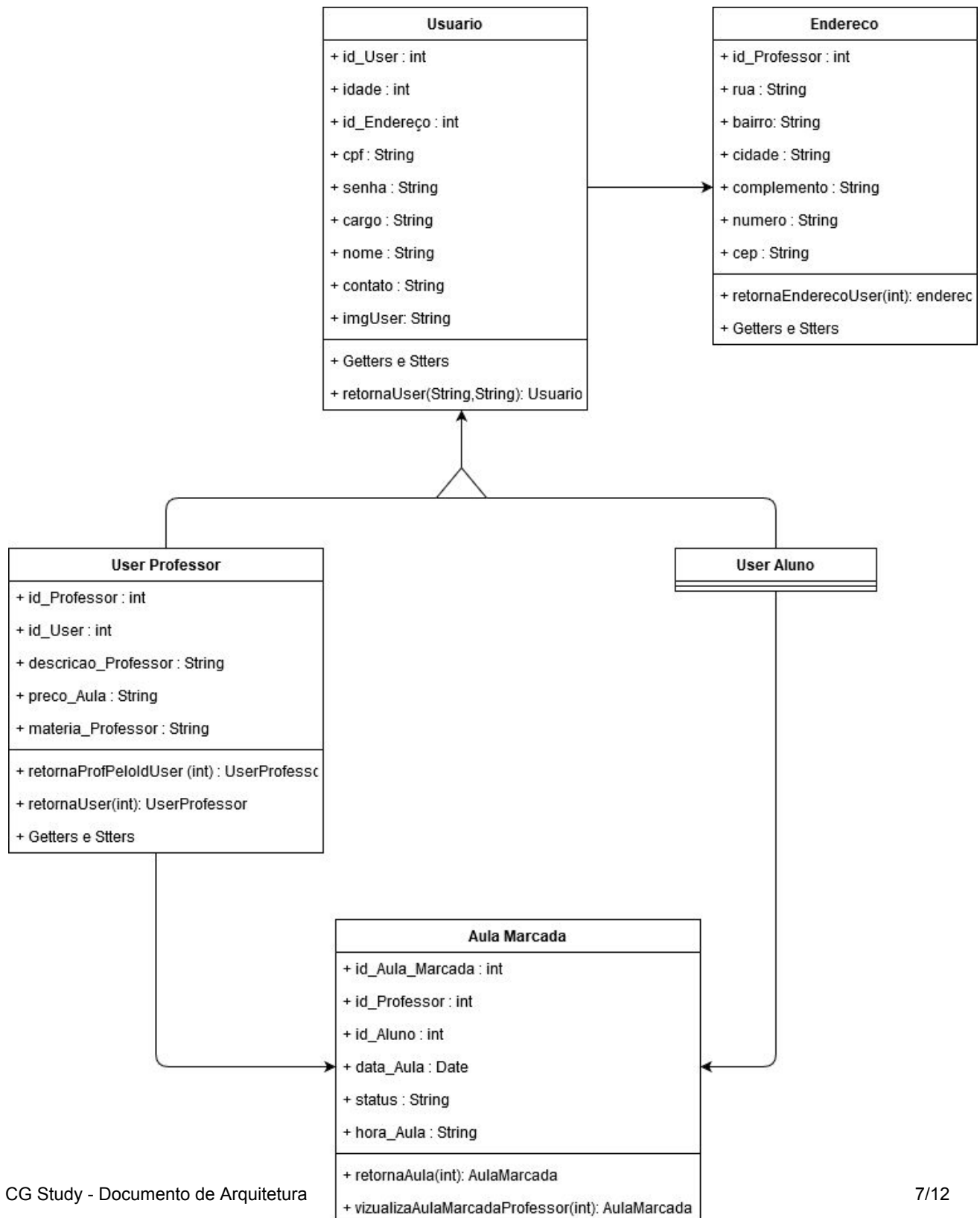
Agendamento de Horários: A classe AulaMarcada realiza operações CRUD para mostrar e gravar horários de aula e agendamentos feito por alunos.

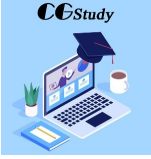


Trabalho de Arquitetura de Software

Documento de Arquitetura

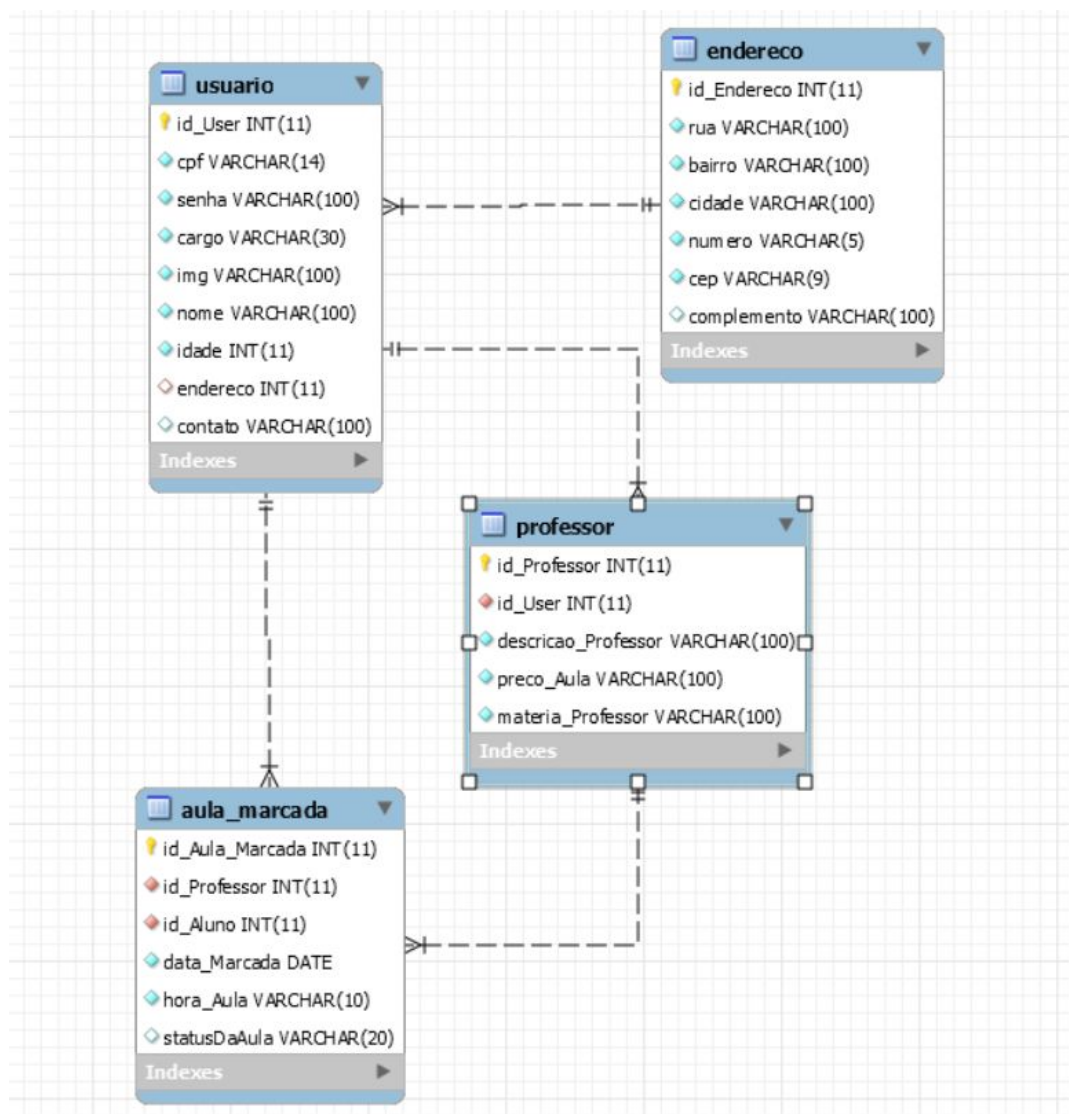
Código/Sigla: CGS Nome do Projeto: CG Study - Busca de Aulas Particulares





	<i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i>			
Trabalho de Arquitetura de Software				
Documento de Arquitetura				
Código/Sigla:	CGS	Nome do Projeto:	CG Study - Busca de Aulas Particulares	

5. Visão de Dados

O sistema terá um banco de dados relacional com 6 tabelas e com a maioria das relações de 1 para N

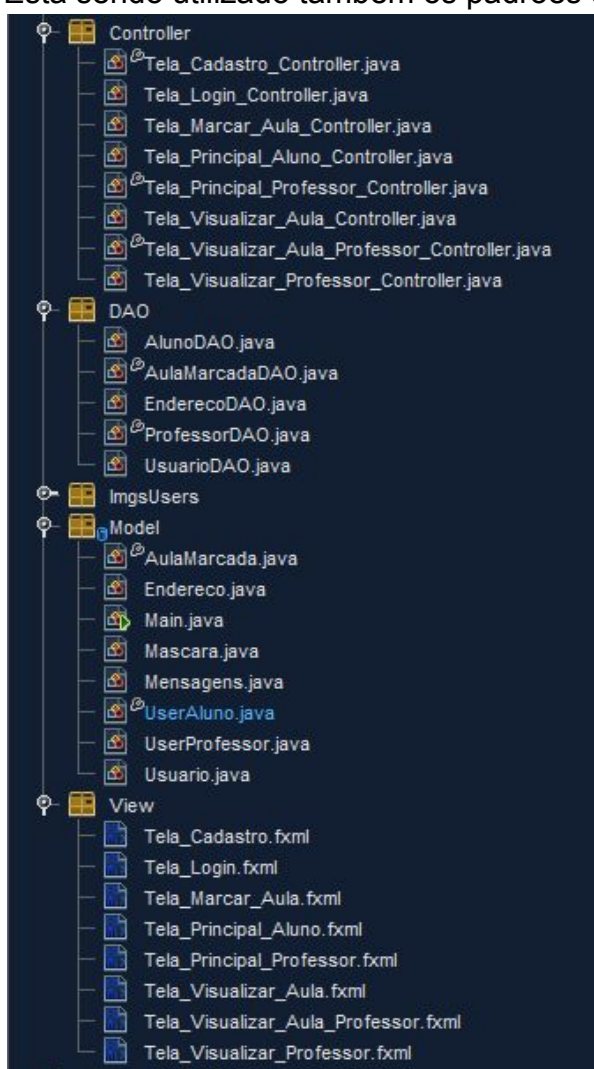


	<p><i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i></p>	
<p align="center">Trabalho de Arquitetura de Software</p>		
<p align="center">Documento de Arquitetura</p>		
<p>Código/Sigla:</p>	<p>CGS</p>	<p>Nome do Projeto: CG Study - Busca de Aulas Particulares</p>


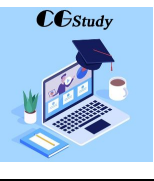
6. Visão de Implementação

O sistema está sendo implementado utilizando o modelo MVC (model, view, controller), para proporcionar melhor manutenção, gerenciamento, manutenção, organização e reuso dos códigos fontes.

Está sendo utilizado também os padrões de projeto Singleton e DAO.



DAO possibilita a separação de códigos para a comunicação com o SGBD não poluindo as classes e deixando uma organização melhor, select, delete, update e até mesmo a conexão com o SGBD estão nos arquivos do DAO.

	<p><i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i></p>	
<p align="center">Trabalho de Arquitetura de Software</p>		
<p align="center">Documento de Arquitetura</p>		
<p>Código/Sigla:</p>	<p>CGS</p>	<p>Nome do Projeto: CG Study - Busca de Aulas Particulares</p>

Está sendo utilizado também os princípios de Clean Code e SOLID, com nomes de classes, métodos , e variáveis bem claras, com métodos curtos, e métodos com funcionalidades específicas, sem ter diversas tarefas.

Deixamos as variáveis e métodos com nomes explicativos e claros.

```

public void setLoginProf(Usuario user) throws SQLException{
    lblUsername.setText(user.getNome());
    lblUsername.setAlignment(Pos.CENTER);
    Image imgUser = new Image(user.getImgUser());
    imgProfessor.setImage(imgUser);
    usuario = user;
    endereco = endereco.retornaEnderecoUser(user.getId_endereco());
    uProf = uProf.retornaProfPeloIdUser(user.getId_User());
    setPainelControle();
}

public void initTableAulasPendentes() {
    ColNomeAluno.setCellValueFactory(new PropertyValueFactory("nome"));
    ColDataAulaPendente.setCellValueFactory(new PropertyValueFactory("data_Aula"));
    ColHora.setCellValueFactory(new PropertyValueFactory("hora_Aula"));
    ColEnderecoAluno.setCellValueFactory(new PropertyValueFactory("contato"));
    tvAulasPendentes.setItems(AulasPendentes());
}

public ObservableList<AulaMarcada> AulasPendentes() {
    AulaMarcadaDAO aulaMD = new AulaMarcadaDAO();
    return FXCollections.observableArrayList(aulaMD.exibeAulasPendentesProfessor(uProf.getId_Professor()));
}

```

Estamos usando herança assim como está no diagrama de classes

```

public class UserProfessor extends Usuario {
    private int id_Professor, id_User;
    private String descricao_Professor, preco_aula, materia_Professor;
}

```

temos algumas exceções que conhecemos que estão sendo tratadas para informar melhor

	<p><i>Instituto Federal de São Paulo – Campus Campinas</i> Arquitetura de Software 4º Semestre - 2019</p>	
<p align="center">Trabalho de Arquitetura de Software</p>		
<p align="center">Documento de Arquitetura</p>		
<p>Código/Sigla:</p>	<p>CGS</p>	<p>Nome do Projeto: CG Study - Busca de Aulas Particulares</p>

o usuário

```
@FXML
public void VisualizarAulaClicado() throws SQLException, IOException{
    AulaMarcada aulaMarcada = new AulaMarcada();
    int controlaErro = 0;

    try{
        aulaMarcada = selectRowAulaMarcada();
    }
    catch (NullPointerException e){
        msg.mensagemAviso("Selecione um professor para visualizar.");
        controlaErro++;
    }
}
```

o sistema contém CRUD que foi testado, e funcionou corretamente



```
public class UsuarioDAO extends SQL{
    ResultSet rsset;

    public ResultSet Login(String cpf,String senha) throws SQLException {
        sql = "SELECT * FROM Usuario WHERE cpf = '" + cpf + "' AND senha = '" + senha + "'";
        rsset = select(sql);
        return rsset;
    }

    public void inserirUsuario(Usuario newUser)throws SQLException{
        EnderecoDAO endeDAO = new EnderecoDAO();
        ResultSet endereco = endeDAO.ultimoEndereco();
        if(endereco.next()){
            newUser.setId_endereco(endereco.getInt(1));
        }

        sql = "INSERT INTO Usuario (cpf,senha,cargo,img, nome, idade, endereco,contato) VALUES ('" + newUser.getCpf() + "', '"
            + newUser.getSenha() + "', '" + newUser.getCargo() + "', '" + newUser.getImgUser() + "', '" + newUser.getNome() + "', " + newUser.getIdade()
            + ", " + newUser.getId_endereco() + ", '" + newUser.getContato() + "')";
        update(sql);
    }

    public void atualizarUsuario(Usuario newUser)throws SQLException{
        sql = "UPDATE Usuario SET nome = '" + newUser.getNome() + "', "
            + "idade = " + newUser.getIdade() + ", "
            + "contato = '" + newUser.getContato() + "' "
            + "WHERE id_User = '" + newUser.getId_User() + "'";
        update(sql);
    }
}
```

	<i>Instituto Federal de São Paulo – Campus Campinas</i> <i>Arquitetura de Software 4º Semestre - 2019</i>	
Trabalho de Arquitetura de Software		
Documento de Arquitetura		
Código/Sigla:	CGS	Nome do Projeto: CG Study - Busca de Aulas Particulares

7. Referências

[1] - Padrões de Projeto. Matérias das aulas da matéria de Arquitetura de Software de 2020 da Professora Bianca Pedrosa.

[2] - Diagramas UML. Matérias das aulas da matéria de Engenharia de Software de 2019 da Professora Zady Salazar