

Universidade Federal do ABC
Bacharelado em Ciência e Tecnologia

Mikael Alves Monteiro

SISTEMA DE MATRÍCULAS UFABC MATRICULERO

Projeto de conclusão de disciplina

Santo André – SP

2015

Mikael Alves Monteiro

SISTEMA DE MATRÍCULAS UFABC MATRICULERO

Projeto de conclusão de disciplina

Projeto de conclusão de disciplina destinado à disciplina Banco de Dados, da Universidade Federal do ABC, ministrada por Márcio Katsumi Oikawa

São Bernardo do Campo – SP

2015

SUMÁRIO

DESCRIÇÃO	4
MODELO ENTIDADE-RELACIONAMENTO ESTENDIDO	5
MODELO LÓGICO	6
MODELO FÍSICO	7
FUNÇÕES PRINCIPAIS DO SISTEMA.....	13
CONCLUSÃO.....	15
EXEMPLOS DE TELA.....	16

DESCRIÇÃO

O UFABC Matriculero é um sistema de matrículas, que atende necessidades de coordenadores (cadastrar e remover disciplinas, turmas e aulas) e alunos (matricular-se em turmas) da Universidade Federal do ABC. O sistema poderá ser acessado por qualquer equipamento que possua acesso à internet e um navegador que suporte HTML5. Trata-se de uma aplicação escrita em PHP, e que utiliza o banco de dados MySQL.

O Matriculero realiza controle de usuários através de login, e conecta-se a uma base de dados para verificação. O software apresenta a possibilidade de múltiplos usuários ao mesmo tempo.

Ao entrar no sistema, o usuário poderá ver informações importantes que já estiverem salvas no banco de dados (o coordenador pode ver disciplinas, turmas e aulas, o aluno pode ver as turmas e aulas).

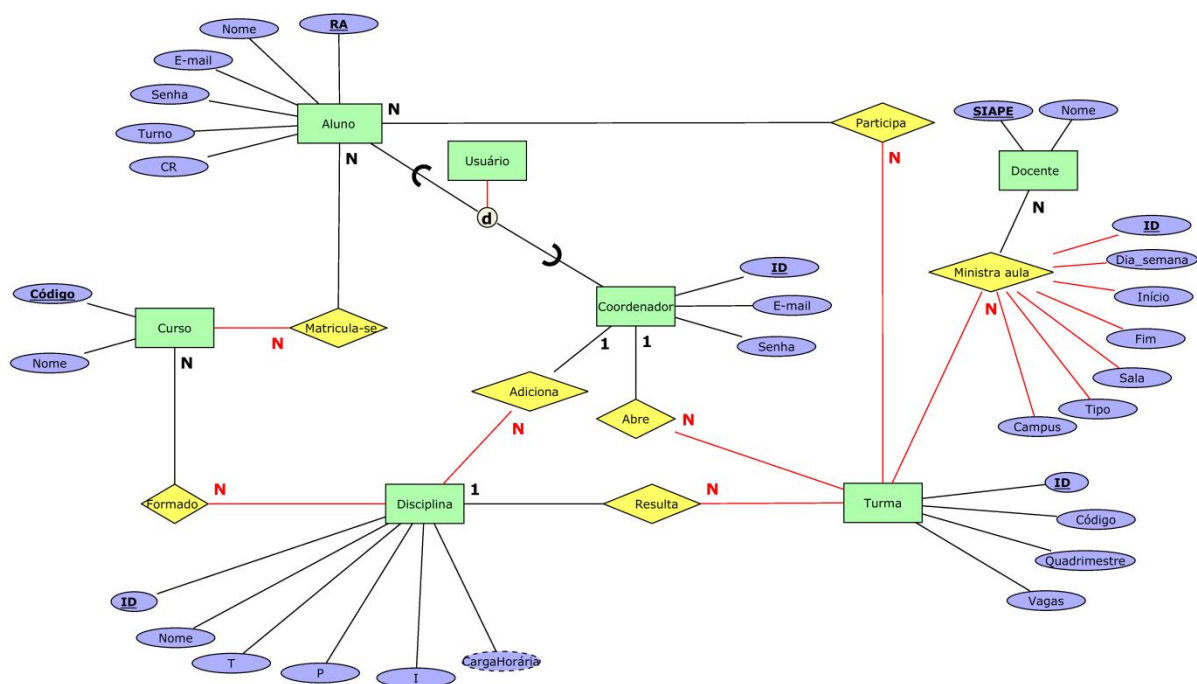
O coordenador tem à sua disposição links que permitem acesso a telas de adição e remoção de disciplinas, turmas e aulas.

O aluno poderá ver todas as turmas disponíveis, realizar matrículas e se desmatricular.

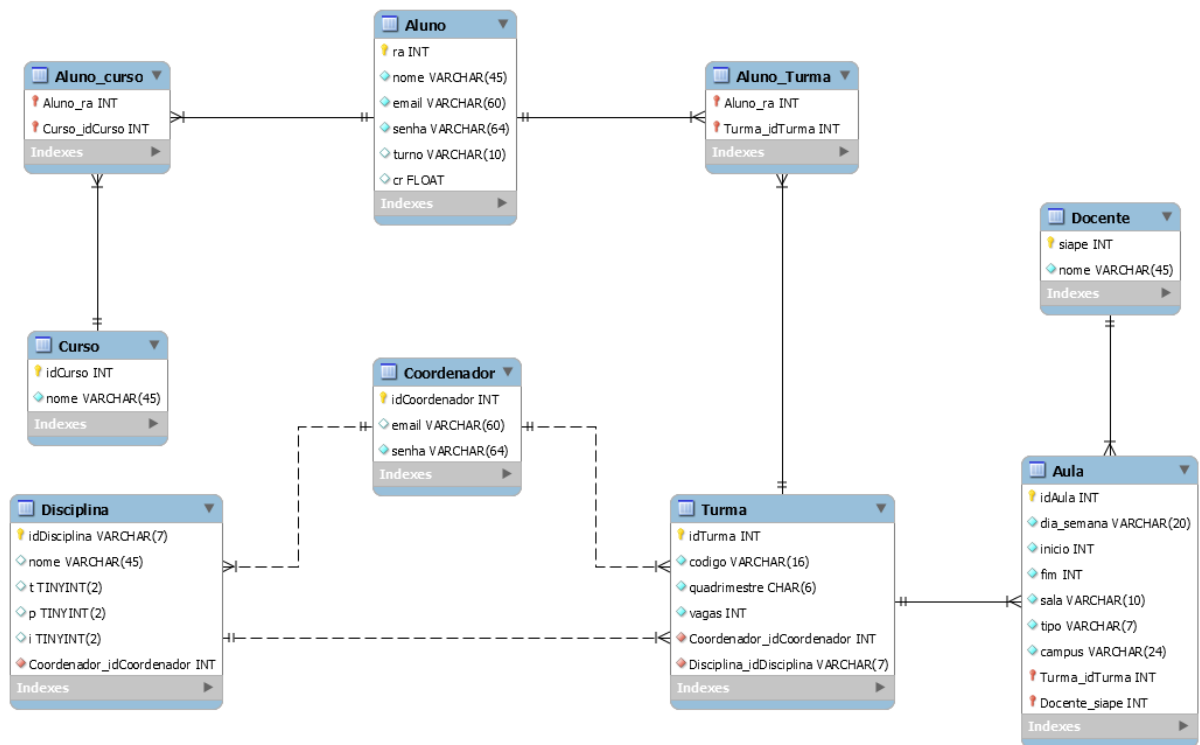
O sistema ainda possui algumas restrições de funcionalidade:

1. Ainda não existe limite de professores por turma;
2. Ainda não existe o mecanismo de análise de T-P-I para cadastro;
3. Ainda não existe o sistema de análise de conflito de horários para matrícula;
4. Aluno ainda não recebe aviso de deferimento ou indeferimento;
5. Ainda não existe o mecanismo de acerto de vagas por requisição.

MODELO ENTIDADE-RELACIONAMENTO ESTENDIDO



MODELO LÓGICO



MODELO FÍSICO

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
```

```
CREATE SCHEMA IF NOT EXISTS `matricula` DEFAULT CHARACTER SET utf8 COLLATE
utf8_general_ci ;
USE `matricula` ;
```

Tabela “Aluno”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Aluno` (
  `ra` INT NOT NULL COMMENT '',
  `nome` VARCHAR(45) NOT NULL COMMENT '',
  `email` VARCHAR(60) NOT NULL COMMENT '',
  `senha` VARCHAR(64) NOT NULL COMMENT '',
  `turno` VARCHAR(10) NULL COMMENT '',
  `cr` FLOAT NULL COMMENT '',
  UNIQUE INDEX `email_UNIQUE` (`email` ASC) COMMENT '',
  PRIMARY KEY (`ra`) COMMENT '',
  UNIQUE INDEX `ra_UNIQUE` (`ra` ASC) COMMENT '')
ENGINE = InnoDB;
```

Tabela “Curso”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Curso` (
  `idCurso` INT NOT NULL COMMENT '',
  `nome` VARCHAR(45) NOT NULL COMMENT '',
  PRIMARY KEY (`idCurso`) COMMENT '')
```

```
ENGINE = InnoDB;
```

Tabela “Coordenador”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Coordenador` (  
  `idCoordenador` INT NOT NULL COMMENT '',  
  `email` VARCHAR(60) NULL COMMENT '',  
  `senha` VARCHAR(64) NOT NULL COMMENT '',  
  PRIMARY KEY (`idCoordenador`) COMMENT '',  
  UNIQUE INDEX `email_UNIQUE` (`email` ASC) COMMENT '')  
ENGINE = InnoDB;
```

Tabela “Disciplina”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Disciplina` (  
  `idDisciplina` VARCHAR(7) NOT NULL COMMENT '',  
  `nome` VARCHAR(45) NULL COMMENT '',  
  `t` TINYINT(2) NULL COMMENT '',  
  `p` TINYINT(2) NULL COMMENT '',  
  `i` TINYINT(2) NULL COMMENT '',  
  `Coordenador_idCoordenador` INT NOT NULL COMMENT '',  
  PRIMARY KEY (`idDisciplina`) COMMENT '',  
  INDEX `fk_Disciplina_Coordenador1_idx` (`Coordenador_idCoordenador` ASC)  
  COMMENT '',  
  CONSTRAINT `fk_Disciplina_Coordenador1`  
    FOREIGN KEY (`Coordenador_idCoordenador`)  
    REFERENCES `matricula`.`Coordenador` (`idCoordenador`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)
```



```
ENGINE = InnoDB;
```

Tabela “Turma”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Turma` (  
  `idTurma` INT NOT NULL AUTO_INCREMENT COMMENT '',  
  `codigo` VARCHAR(16) NOT NULL COMMENT '',  
  `quadrimestre` CHAR(6) NOT NULL COMMENT '',  
  `vagas` INT NOT NULL COMMENT '',  
  `Coordenador_idCoordenador` INT NOT NULL COMMENT '',  
  `Disciplina_idDisciplina` VARCHAR(7) NOT NULL COMMENT '',  
  PRIMARY KEY (`idTurma`) COMMENT '',  
  UNIQUE INDEX `codigo_UNIQUE` (`codigo` ASC) COMMENT '',  
  INDEX `fk_Turma_Coordenador1_idx` (`Coordenador_idCoordenador` ASC)  
  COMMENT '',  
  INDEX `fk_Turma_Disciplina1_idx` (`Disciplina_idDisciplina` ASC)  
  COMMENT '',  
  CONSTRAINT `fk_Turma_Coordenador1`  
    FOREIGN KEY (`Coordenador_idCoordenador`)  
    REFERENCES `matricula`.`Coordenador` (`idCoordenador`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  CONSTRAINT `fk_Turma_Disciplina1`  
    FOREIGN KEY (`Disciplina_idDisciplina`)  
    REFERENCES `matricula`.`Disciplina` (`idDisciplina`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

Tabela “Docente”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Docente` (  
  `siape` INT NOT NULL COMMENT '',  
  `nome` VARCHAR(45) NOT NULL COMMENT '',  
  PRIMARY KEY (`siape`) COMMENT '')  
ENGINE = InnoDB;
```

Tabela “Aluno_curso”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Aluno_curso` (  
  `Aluno_ra` INT NOT NULL COMMENT '',  
  `Curso_idCurso` INT NOT NULL COMMENT '',  
  PRIMARY KEY (`Aluno_ra`, `Curso_idCurso`) COMMENT '',  
  INDEX `fk_Curso_has_Aluno_Aluno1_idx` (`Aluno_ra` ASC) COMMENT '',  
  INDEX `fk_Curso_has_Aluno_Curso1_idx` (`Curso_idCurso` ASC) COMMENT '',  
  CONSTRAINT `fk_Curso_has_Aluno_Curso1`  
    FOREIGN KEY (`Curso_idCurso`)  
    REFERENCES `matricula`.`Curso` (`idCurso`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  CONSTRAINT `fk_Curso_has_Aluno_Aluno1`  
    FOREIGN KEY (`Aluno_ra`)  
    REFERENCES `matricula`.`Aluno` (`ra`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

Tabela “Aula”

```
CREATE TABLE IF NOT EXISTS `matricula`.`Aula` (  
  `idAula` INT NOT NULL AUTO_INCREMENT COMMENT '',  
  `dia_semana` VARCHAR(20) NOT NULL COMMENT '',  
  `inicio` INT NOT NULL COMMENT '',  
  `fim` INT NOT NULL COMMENT '',  
  `sala` VARCHAR(10) NOT NULL COMMENT '',  
  `tipo` VARCHAR(7) NOT NULL COMMENT '',  
  `campus` VARCHAR(24) NOT NULL COMMENT '',  
  `Turma_idTurma` INT NOT NULL COMMENT '',  
  `Docente_siape` INT NOT NULL COMMENT '',  
  PRIMARY KEY (`idAula`, `Turma_idTurma`, `Docente_siape`) COMMENT '',  
  INDEX `fk_Turma_has_Docente_Docente1_idx` (`Docente_siape` ASC) COMMENT  
  '',  
  INDEX `fk_Turma_has_Docente_Turma1_idx` (`Turma_idTurma` ASC) COMMENT  
  '',  
  CONSTRAINT `fk_Turma_has_Docente_Turma1`  
    FOREIGN KEY (`Turma_idTurma`)  
    REFERENCES `matricula`.`Turma` (`idTurma`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  CONSTRAINT `fk_Turma_has_Docente_Docente1`  
    FOREIGN KEY (`Docente_siape`)  
    REFERENCES `matricula`.`Docente` (`siape`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

Tabela “Aluno_Turma”

```

CREATE TABLE IF NOT EXISTS `matricula`.`Aluno_Turma` (
  `Aluno_ra` INT NOT NULL COMMENT '',
  `Turma_idTurma` INT NOT NULL COMMENT '',
  PRIMARY KEY (`Aluno_ra`, `Turma_idTurma`) COMMENT '',
  INDEX `fk_Turma_has_Aluno_Aluno1_idx` (`Aluno_ra` ASC) COMMENT '',
  INDEX `fk_Turma_has_Aluno_Turma1_idx` (`Turma_idTurma` ASC) COMMENT '',
  CONSTRAINT `fk_Turma_has_Aluno_Turma1`
    FOREIGN KEY (`Turma_idTurma`)
      REFERENCES `matricula`.`Turma` (`idTurma`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  CONSTRAINT `fk_Turma_has_Aluno_Aluno1`
    FOREIGN KEY (`Aluno_ra`)
      REFERENCES `matricula`.`Aluno` (`ra`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

FUNÇÕES PRINCIPAIS DO SISTEMA

Login no sistema

(importante para manter a privacidade e autenticidade)

```
$sqlCoordenador = mysqli_query($mysqli, "SELECT idCoordenador, email, senha FROM coordenador WHERE email = '$email' AND senha = '$senha'");  
  
$sqlAluno = mysqli_query($mysqli, "SELECT ra FROM aluno WHERE email = '$email' AND senha = '$senha'");
```

Cadastro de disciplina

(essencial para criação de turmas)

```
$sqlAdicionaDisciplina = mysqli_query($mysqli, "INSERT INTO disciplina  
VALUES ('$codDisciplina', '$nome', '$t', '$p', '$i', '$numCoord')");
```

Cadastro de turma

(essencial para criação de aulas)

```
$sqlAdicionaDisciplina = mysqli_query($mysqli, "INSERT INTO disciplina  
VALUES ('$codDisciplina', '$nome', '$t', '$p', '$i', '$numCoord')");
```

Cadastro de aula

(essencial para finalizar a criação de turmas)

```
$sqlPegaIdTurma = mysqli_query($mysqli, "SELECT idTurma FROM turma WHERE  
codigo = '$codTurma'");
```

```
$sqlAdicionaDisciplina = mysqli_query($mysqli, "INSERT INTO aula  
(dia_semana, inicio, fim, sala, tipo, campus, Turma_idTurma,  
Docente_siape) VALUES ('$diaSemana', '$horaInicio', '$horaFim', '$sala',  
'$tipo', '$campus', '$idTurma', '$siapeDocente')");
```

Matrícula em turma

(imprescindível para o aluno)

```
$sqlPegaIdTurma = mysqli_query($mysqli, "SELECT idTurma FROM turma WHERE  
codigo = '$codTurma'");
```

```
$sqlMatricula = mysqli_query($mysqli, "INSERT INTO aluno_turma VALUES  
('$ra', '$idTurma')");
```

CONCLUSÃO


O banco de dados apresenta modelagem satisfatória, mas precisou de diversos ajustes (trocas de tipo de valor de chave primária, uso de chaves primárias auto-incrementais e aplicação de regras de unicidade mesmo em atributos não primários) para chegar na situação atual. A aplicação cumpre com as tarefas mais básicas, mas ainda necessita de ajustes para garantir a segurança e a facilidade de uso (alterações para facilitar a navegação entre funções). Considerando a segurança de Banco de Dados, seria fortemente recomendado o uso de Stored Procedures (principalmente para evitar SQL Injection). O método de conexão a banco usado no PHP é o `mysqli_*`, que suporta Stored Procedures, o que facilita a implantação. O projeto é uma boa contribuição para compreender a importância de bancos de dados, e principalmente para aumentar a familiaridade entre programador e SGBD's. Uma grande dificuldade encontrada é a mudança de tática de programação que é vista comparando Java com PHP. O sistema foi produzido em PHP para atingir melhor similaridade com sistemas reais de matrícula.

O projeto pode ser melhorado, começando pelo acréscimo de funcionalidades requeridas no documento de orientação, e logo após passar por design visual.

Nota: todos os arquivos do projeto acompanham a submissão desse documento.

EXEMPLOS DE TELA

Tela de login:



UFABC

Login no Matriculero

E-mail

Senha

[Limpar informações](#) [Entrar](#)

Tela de coordenador:

Página do coordenador

[Cadastro de disciplina](#) [Remoção de disciplina](#) [Cadastro de turma](#) [Remoção de turma](#) [Cadastro de aula](#) [Remoção de aula](#) [Sair](#)

Disciplinas atualmente cadastradas

Código	Nome	T	P	I
MC3310	Banco de Dados	4	2	6
MC3311	Inteligência Artificial	2	2	4

Turmas atualmente cadastradas

Disciplina	Código	Quadri.	Vagas
Banco de Dados	02_MC33102015Q3	2015Q3	60
Banco de Dados	01_MC33102015Q3	2015Q3	46

Dia da semana	Início	Fim	Sala	Tipo	Campus
Segunda	8h	10h	S102	Prática	Santo André

Disciplina	Código	Quadri.	Vagas
Inteligência Artificial	01_MC33112015Q3	2015Q3	60

Docentes atualmente cadastrados

Nome

Tela de cadastro de disciplina:

Cadastro de disciplina

Código da disciplina

Nome

T

P

I

Limpar informações

Salvar

Voltar

Tela de remoção de disciplina

Remoção de disciplina

Código da disciplina

Limpar informações

Salvar

Voltar

Tela de aluno

Página do aluno

Matricula em turmas

Remover matrícula em turmas

Sair

Turmas em que está matriculado

Disciplina	Código	Vagas			
Banco de Dados	01_MC33102015Q3	46			
Dia da semana	Início	Fim	Sala	Tipo	Campus
Segunda	8h	10h	S102	Prática	Santo André

Turmas atualmente cadastradas

Disciplina	Código	Quadri.	Vagas		
Banco de Dados	02_MC33102015Q3	2015Q3	60		
Banco de Dados	01_MC33102015Q3	2015Q3	46		
Dia da semana	Início	Fim	Sala	Tipo	Campus
Segunda	8h	10h	S102	Prática	Santo André
Inteligência Artificial	01_MC33112015Q3	2015Q3	60		