



UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO  
DEPARTAMENTO DE ESTATÍSTICA E INFORMÁTICA  
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO  
BANCO DE DADOS (BC3) – 2015.2

# PROJETO

# AVA

Alunos: Guilherme Melo  
João Nascimento  
Paulo Menezes

Recife - 2015

## Sumário

1	Introdução.....	3
2	Descrição do projeto .....	3
3	Requisitos funcionais.....	3
4	Requisitos não funcionais.....	4
5	Atores .....	5
	Aluno .....	5
	Professor .....	5
	Administrador .....	5
6	Relatórios .....	6
7	Principais Casos de Uso .....	6
8	DIAGRAMAS .....	8
9	Triggers, procedimento e funções .....	9
	9.1 Triggers.....	9
	9.2 Procedimentos .....	10
	9.3 Funções .....	11
10	Instruções para rodar o programa .....	12
11	Principais Telas do Sistema .....	13
	TELA LOGIN.....	13
	TELA INICIO.....	13
	TELA VISUALIZAÇÃO .....	14
	TELA CADASTRO .....	14
	TELA MATRICULA .....	15
	TELA APLICAR NOTAS .....	15
12	Conclusão .....	16

## 1 Introdução

O projeto AVA surgiu após a implementação de sistema homônimo na Universidade Federal Rural de Pernambuco (UFRPE). AVA é um acrônimo para Ambiente Virtual de Aprendizagem e é o ambiente da UFRPE usado por docentes e discentes, sendo útil ao primeiro para gerenciamento de conteúdo aos alunos, administração do curso e do acompanhamento do desempenho de estudantes, enquanto que para os estudantes, facilita a comunicação com o professor e demais alunos e da permanência de todo o conteúdo a ser utilizado por ele em sua vida acadêmica num único local. Sendo assim, o AVA funciona de forma similar a Edmodo ou Edulify. Aproveitando o lançamento oficial do AVA no segundo semestre de 2015, foi-se confirmado a escolha pelos estudantes para desenvolvê-lo.

No entanto, o projeto AVA não tem por objetivo reproduzir o AVA, pois aquele possui objetivos distintos desse, pois (o projeto AVA) não é focado única e exclusivamente em funcionalidade do AVA, mas também em funcionalidades do Sig@ (Sistema de Informações e Gestão Acadêmica). Seria até, talvez, mais adequado o uso de Sig@VA ou qualquer aglutinação desses dois siglemas para representar o projeto. Ademais, o projeto AVA tem como objetivos fornecer uma aplicação capaz de modular um ambiente de gerenciamento de informações pelos atores do sistema (professor, aluno e secretária) e permitir a troca dessas informações, construindo um ambiente integrado e uma ferramenta que busca simplificar processos comuns da vida acadêmica.

## 2 Descrição do projeto

O Ambiente Virtual de Aprendizagem (AVA) é um sistema usado para facilitar a gerência e a comunicação do corpo discente e docente de uma universidade. Os usuários do sistema têm acesso a serviços do ambiente virtual, tais como: matrícula em uma disciplina, inserção de projetos de pesquisas, solicitação para participar de projetos etc.

## 3 Requisitos funcionais

1. O sistema é composto por usuários que possuem CPF, nome, e-mail, senha e tipo (professor, aluno ou administrador[adm]). Esses usuários podem ser alunos ou professores. Professores podem desempenhar a função de coordenadores dos cursos oferecidos pela universidade.
2. O aluno de uma universidade ou é da graduação (graduando) ou da pós-graduação (mestrando ou doutorando). Para tal condição (a de aluno), deve estar matriculado em um determinado curso da universidade, na qual possui vários alunos. Na universidade existem vários cursos disponíveis.
3. Um curso tem como atributos: nome, código de identificação, departamento a qual está associado, quantidade de alunos matriculado no curso que ainda estejam vinculados a universidade e seu tipo (graduação ou pós-graduação). Cada curso deve possuir várias disciplinas e uma disciplina pode estar vinculada a mais de um curso.
4. Uma disciplina possui: código, nome, carga horária e nº créditos. Uma disciplina pode exigir nenhum ou alguma(s) disciplina(s) como pré-requisito. Só é permitido "pagar" uma disciplina se, e somente se, todos os pré-requisitos forem satisfeitos. Os pré-requisitos da disciplina que o aluno deseja ser matriculado serão comparados com as disciplinas do seu histórico e verificado se os pré-

requisitos foram “pagos” e aluno obteve êxito (conseguiu aprovação) neles para serem “satisfeitos”. Cada disciplina deve oferecer uma ou mais ofertas de disciplina para a matrícula de alunos, na qual cada oferta está vinculada a uma disciplina.

5. A oferta de uma disciplina disponibilizará: semestre, ano, dias e horário de realização das aulas da disciplina, bem como quantidade de alunos matriculados. Cada oferta de disciplina pode ter vários alunos matriculados.
6. Uma oferta de disciplina deve ser ministrada por um professor. Um professor pode ministrar mais de uma oferta de disciplina. O professor ficará responsável de organizar todo o calendário dessa oferta da disciplina.
7. É necessário que o aluno esteja matriculado ao menos em uma disciplina e no máximo em seis. Na realização da matrícula do aluno, deve ser guardado a data de realização de matrícula e o número de protocolo.
8. O aluno que está matriculado numa oferta de uma disciplina deve possuir notas que são usadas para verificar o aproveitamento e a aprovação do aluno. A média geral deve estar vinculada a um aluno.
9. A nota de cada aluno, tem como atributos: notas da 1ª VA, 2ª VA, 3ª VA e prova final. Para conseguir aprovação na disciplina, o aluno deve ter uma média igual ou superior a 7 (sete) na oferta da disciplina quando até a realização da terceira VA, e 5 (cinco) ou mais quando realizada a prova final. Conseguindo até o término da oferta da disciplina igual ou acima da média, a situação do aluno naquela disciplina é tida como aprovada. Senão, ele é reprovado. A média é calculada da seguinte forma:  $(VA1^a \text{ Maior Nota} + VA 2^a \text{ Maior Nota}) / 2$ . Se não obtiver média suficiente para sua aprovação até a 3ª VA, a sua média final é calculada a seguir:  $(\text{Média aritmética das duas maiores notas (1ª VA, 2ª VA e 3ª VA)} + \text{Nota da prova final}) / 2$ .
10. Professores podem criar projetos de pesquisas. Um projeto de pesquisa tem um título, um código de identificação, modalidade (PIBIC, PIBIT, PICME etc.), organização (CNPq, FACEPE, Capes etc.), valor da bolsa dada aos interessados e nº de vagas restantes. Um professor pode coordenar mais de um projeto, e um projeto deve ser coordenado por um ou mais professores.
11. Alunos podem participar de projetos de pesquisa. Um projeto deve ter participação de alunos. Vários alunos podem participar de vários projetos de pesquisa.
12. Com o desenvolvimento de um projeto de pesquisa, um ou mais artigos podem ser gerados. Um artigo está associado ao código de identificação de um projeto. O artigo tem como atributos: nome, tema, objetivo e área científica.

#### 4 Requisitos não funcionais

1. O professor atribui as notas (podem ter valores distintos) aos alunos individualmente.
2. O número de protocolo da matrícula em um curso será gerado automaticamente e aleatoriamente pelo sistema.
3. Os usuários podem visualizar avisos num quadro (tela) direcionados a ele (por pagar uma oferta de disciplina ou enviado diretamente a ele) ou públicos (todos podem visualizar). Os avisos são organizados de acordo com sua prioridade (prioridade mais alta está acima dos de prioridade mais abaixo). O aviso só pode

- ser visualizado a determinado conjunto, podendo ser direcionado a: uma pessoa, a uma turma (oferta de uma disciplina) ou a todos os usuários.
4. A nota de um aluno será de 0 (zero) a 10 (dez). O professor poderá publicar e modificar a nota dentro do prazo definido pela universidade.

## 5 Atores

### Aluno

O Aluno poderá realizar essas operações no sistema:

- O aluno poderá consultar uma tela na qual se pode visualizar todos os avisos a ele direcionado. Ele também pode consultar suas notas das ofertas do período que ele está cursando no momento.
- O aluno poderá realizar sua matricula no sistema, visualizando e selecionando as ofertas disponíveis para ele no momento.
- O aluno poderá visualizar e enviar solicitações para participar de projetos disponíveis no sistema.

### Professor

O professor como ator do sistema, poderá realizar essas operações:

- Assim como aluno, o professor pode consultar seus avisos, como solicitações de projetos e outras mensagens.
- O professor poderá cadastrar, atualizar e remover projetos de pesquisa no sistema.
- O professor será capaz de visualizar todo o projeto de pesquisa que ele está envolvido.
- O professor poderá visualizar as ofertas das disciplinas que ele está ministrando, também poderá visualizar os alunos dessas ofertas.
- O professor será capaz de aceitar as solicitações de participação no projeto que ele está participando, vinculando assim o aluno ao projeto, ele também será capaz de aplicar notas aos alunos que estão vinculados as ofertas que ele ministra.

### Administrador

O administrador do sistema poderá realizar as seguintes funcionalidades:

- O administrador poderá Cadastrar, Remover ou Atualizar:
  - Usuários do Sistema.
  - Cursos do Sistema.
  - Disciplinas do Sistema.
  - Ofertas do Sistema.
  - Projetos do Sistema.
- O administrador poderá também vincular um professor a uma oferta de uma disciplina.
- OBS.: O administrador não precisa se matricular no sistema ele realizará o Login preenchendo no CPF: admin e SENHA: admin.

## 6 Relatórios

Os Relatórios se encontram em anexo na pasta do projeto AVA:

Anexo 1 - ALUNOS

- Listar condição de todos alunos do banco de dados em cada oferta já paga

Anexo 2 - CURSOS

- Listar todos os alunos que fazem parte de um projeto

Anexo 3 - PROJETOS

- Listar todas as ofertas de disciplinas já realizadas, mostrando o nome da disciplina, tipo, carga horária, quantidade de créditos, curso na qual foi realizada, departamento.

Obs.: Esses relatórios estão presentes na pasta:

C:\Users\[nome do usuário]\...\AVA-javafx-aspect-2\ProjetoBanco, no arquivo *Relatório.sql*.

## 7 Principais Casos de Uso

DESCRIÇÃO CASOS DE USO AVA:

<b>CASO DE USO: Matricular Aluno</b>
<b>IDENTIFICADOR:</b> RF01
<b>DESCRIÇÃO:</b> Este caso de uso é responsável pela matrícula do aluno
<b>ATOR(ES):</b> Aluno
<b>PRIORIDADE:</b> Importante
<b>REQUISITOS NÃO-FUNCIONAIS:</b> O sistema disponibilizará uma tela onde o aluno poderá visualizar as ofertas da disciplina para matrícula disponíveis
<b>PRÉ-CONDIÇÃO:</b> O aluno deverá estar ativo no sistema e matriculado em algum curso
<b>PÓS-CONDIÇÃO:</b> O aluno estará matriculado nas ofertas (de 01 a 06 ofertas) que o próprio escolheu para o período
<b>FLUXO DE EVENTOS PRINCIPAL:</b> <ol style="list-style-type: none"><li>1. O aluno escolhe a opção de matrícula no sistema</li><li>2. O sistema disponibiliza todas as ofertas que o aluno está apto a pagar</li><li>3. O aluno faz escolhe das ofertas desejadas (de 01 a 06 ofertas)</li><li>4. O aluno confirma a matrícula nas ofertas</li></ol>

5. O sistema cadastrará as ofertas escolhidas
<b>FLUXO SECUNDÁRIO:</b> <ol style="list-style-type: none"> <li>1. No passo 2, se o aluno escolher menos de 3 ofertas ou mais de 10 ofertas, o sistema o notifica que chegou ao limite de escolhas.</li> <li>2. No passo 5, caso o aluno já tenha efetuado o cadastro no sistema, o sistema notificará e retornará ao passo 2.</li> </ol>

<b>CASO DE USO :</b> Participar de um projeto de Pesquisa
<b>IDENTIFICADOR:</b> RF02
<b>DESCRIÇÃO:</b> Este caso de uso é responsável pela integração de alunos a um projeto de pesquisa
<b>ATOR(ES):</b> Aluno
<b>PRIORIDADE:</b> Importante
<b>REQUISITOS NÃO-FUNCIONAIS:</b> O sistema disponibilizará um campo na tela para pesquisa do projeto de pesquisa
<b>PRÉ-CONDIÇÃO:</b> O aluno deverá estar ativo no sistema
<b>PÓS-CONDIÇÃO:</b> O aluno participará de um projeto de pesquisa
<b>FLUXO DE EVENTOS PRINCIPAL:</b> <ol style="list-style-type: none"> <li>1. O aluno escolhe a opção de projeto de pesquisa no menu do sistema</li> <li>2. O sistema mostra todos os projetos de pesquisa cadastrados com vagas disponíveis</li> <li>3. O aluno escolhe os projetos que deseja participar e manda uma requisição para participar do(s) projeto(s) escolhido(s)</li> <li>4. O sistema envia as requisições aos professores</li> <li>5. O professor aceita o a solicitação</li> <li>6. O sistema verifica que foi aceita a requisição</li> <li>7. O aluno é vinculado ao projeto de pesquisa</li> </ol>
<b>FLUXO SECUNDÁRIO:</b> <ol style="list-style-type: none"> <li>1. No passo 5, caso o professor não aceite a solicitação o sistema não vinculará o aluno ao projeto e o sistema mostrará que a solicitação foi recusada.</li> </ol>

<b>CASO DE USO:</b> Cadastrar usuário
<b>IDENTIFICADOR:</b> RF03
<b>DESCRIÇÃO:</b> Este caso de uso é responsável pelo cadastro de um usuário no sistema
<b>ATOR(ES):</b> Administrador
<b>PRIORIDADE:</b> Importante
<b>REQUISITOS NÃO-FUNCIONAIS:</b>
<b>PRÉ-CONDIÇÃO:</b> O Administrador deve estar ativo no sistema
<b>PÓS-CONDIÇÃO:</b> Um usuário novo é cadastrado no sistema
<b>FLUXO DE EVENTOS PRINCIPAL:</b> <ol style="list-style-type: none"> <li>1. O administrador selecionar a opção cadastrar usuário</li> <li>2. O administrador passará os dados necessários para o cadastro</li> <li>3. O sistema criará e salvará um novo usuário no sistema</li> <li>4. O sistema mostrará que o usuário foi cadastrado com sucesso</li> </ol>
<b>FLUXO SECUNDÁRIO:</b> <ol style="list-style-type: none"> <li>1. No passo 3, caso o usuário já estiver cadastrado, o sistema mostrará que o usuário já possui uma conta e voltará ao passo 2.</li> </ol>

## 8 DIAGRAMAS

Os Diagramas Estão em Anexo na Pasta Diagramas do Projeto AVA:

Anexo 1 - Activity Diagram Cadastrar Usuario.

Anexo 2 -Activity Diagram Maricula.

Anexo 3 - Activity Diagram Solicitação Projeto.

Anexo 4 -Sequence Diagram Cadastrar Usuario.

Anexo 5 -Sequence Diagram Matricula.

Anexo 6 -Sequence Diagram Solicitacao Projeto.



## 9 Triggers, procedimento e funções

Uma breve descrição das triggers, procedimentos e funções usadas no projeto AVA.

### 9.1 Triggers

- **salvarAlunoProjeto after update on solicitacaoprojeto** - após qualquer mudança na tabela de *solicitacaoprojeto* é tomado uma decisão que depende da seguinte condição. Se a variável booleana *estado = 1*, que indica que foi aprovada sua solicitação para a entrada no projeto, insira uma linha na tabela de *participarprojeto* e delete sua linha em *solicitacaoprojeto*. Caso contrário, *estado = 0*, apenas remova a linha da tabela *solicitacaoprojeto*.
- **inserirAvisoAposMatriculaEInserirNotaNaTabela after insert on matricular** - após inserida uma linha na tabela matricular, a trigger apenas insere uma linha na tabela nota para que as notas do aluno naquela oferta possam ser inseridas. Além disso, um aviso é criado após a inserção de linha na tabela *matricular*, direcionado ao aluno matriculado para que saiba que houve a confirmação de sua matrícula.
- **inserirAlunoNaOfertaEmHistorico after insert on nota** - sua função é apenas inserir uma linha na tabela, porém na tabela *historico*, toda vez que for inserida uma linha em nota.
- **avisarAlunoNotaECalcularMedia after update on nota** - após uma modificação de uma nota na tabela nota. O aluno associado àquela nota modificada, receberá um aviso informando que sua nota já se encontra no sistema, e, portanto, pode ser consultada. Além disso sua nota é calculada toda vez que uma nota é atualizada, chamando o procedimento *calcularMediaAluno()* [veja mais detalhes sobre esse procedimento na parte de procedimento do documento].
- **atualizarNAlunosOferta after update on matricular** - essa trigger é engatilhada toda a vez que houver uma modificação na tabela matricular. A tabela matricular em sua essência não permite atualização de uma linha, devido ao seu significado. Ou se está matriculado numa oferta de disciplina ou não está. Por isso, quando adicionada uma linha nessa tabela, é feita uma atualização na variável *qtdAlunos* da tabela *ofertadisciplina*, adicionando uma unidade a ela. O contrário, a remoção de uma linha, também é válido, porém, o número de alunos (*qtdAlunos*) é diminuído uma unidade.
- **atualizarNAlunosCurso after update on aluno** - assim como a trigger anterior, ela faz a atualização do número de alunos, só que dessa vez, de um curso. Toda vez que um novo aluno for inserido no banco, mais precisamente na tabela aluno, ela atualizará a variável *qtdAlunos* da tabela *curso*, somando uma unidade. Se um aluno for deletado da tabela, será subtraído um da quantidade de alunos (*qtdAlunos*) da tabela *curso*.
- **criarAvisoProjeto after insert on projetopesquisa** - após inserido um novo projeto pelo professor no banco de dados, um aviso é criado automaticamente com informações básica sobre o novo projeto a todos os usuários cadastrados do banco.
- **avisarProfessorDeSolicitacao after insert on solicitacaoprojeto** - quando o aluno se interessa por uma projeto de pesquisa é possível que solicite ao professor sua entrada nele. Após a solicitação do projeto pelo aluno, um aviso é criado direcionado ao professor que coordena o projeto.

- ***atualizarNVagasProjeto after update on participarprojeto*** - toda vez que ocorrer alguma alteração na tabela *participarprojeto*, uma inserção ou deleção (pela sua natureza não existe atualização, pois ou você está participando de um projeto X ou não está), ele diminui ou aumenta uma unidade a variável *nVagas* na tabela *projetoPesquisa*, respectivamente. Além disso, é criado um aviso para informar aos interessados, a depender do que tenha acontecido com a tabela. Se houve uma inserção na tabela *participarprojeto*, um aviso é criado somente para o aluno que conseguiu a confirmação de sua solicitação de participar projeto de pesquisa. Se houve uma deleção, um aluno deixou de participar de um projeto, um aviso é criado direcionados a todos os alunos sobre a oportunidade de uma vaga no projeto.

## 9.2 Procedimentos

- ***atualizarNAlunosCursoQuandoNull(in idC int)*** - procedimento usado apenas para atualizar o número de alunos nos cursos já criados no banco de dados com os dados que estão já lá inseridos. É passado como argumento o *idC*, que é o identificador do curso. Usada para evitar a inserção manual dos dados e de um possível erro.
- ***atualizarNAlunosOfertaQuandoNull(in idO int)*** - função semelhante ao procedimento acima, porém atualiza o número de alunos de cada oferta já inclusa no banco de dados. É passado como argumento o *idO*, que é o identificador da oferta. Usada para evitar a inserção manual dos dados e de um possível erro.
- ***atualizarAlunosCurso()*** - procedimento que itera a variável inteira *nCurso* de um a duzentos chamando o procedimento *atualizarNAlunosCursoQuandoNull(in idC int)*, passando *nCurso* como parâmetro.
- ***atualizarAlunosOferta()*** - procedimento que itera a variável inteira *nOferta* de um a duzentos chamando o procedimento *atualizarNAlunosOfertaQuandoNull(in idO int)*, passando *nOferta* como parâmetro.
- ***calcularMediaDeCadaAlunoNaOferta(in idO int)*** - procedimento com intuito de atualizar o banco de dados com a média e condição na tabela *historico* de cada aluno nas ofertas já cadastradas.
- ***calcularMediaAluno(idOf int, cpfAl varchar(14))*** - procedimento usado para o cálculo da média e condição do aluno (aprovado, aprovado por média ou reprovado). É chamada toda a vez que há inserção de nota (atualização de uma das variáveis de nota, que são: *nota1*, *nota2*, *nota3* e *notaFinal*) de um aluno numa oferta de disciplina. São os seus parâmetros: *in idOf int*, identificador da oferta e *in cpfAl varchar(14)*, CPF do aluno.
- ***atualizarHistorico()*** - procedimento parecido com *atualizarAluno[...]()*, porém itera a variável inteira *nOferta* de um a duzentos passando-a como argumento do procedimento *calcularMediaDeCadaAlunoNaOferta(in idO int)*, atualizando todo o histórico já existente no banco de dados.
- ***adicionarAlunoAO oferta(in cpfAl varchar(14), in idO int, in dataMatr date, in numProt varchar(20))*** - adiciona um aluno a uma oferta cadastrada no sistema, ou seja, um aluno é matriculado numa oferta na qual solicitou matrícula. Para isso, é inserida uma linha na tabela matricular passando os parâmetros: *in cpfAl*

*varchar(14)*, o CPF do aluno; *in idO int*, o identificador de oferta; *in dataMatr date*, a data de matrícula na oferta e *in numProt varchar(20)*, número de protocolo da matrícula efetuada.

- ***adicionarProfessorAO oferta(in cpfProf varchar(14), in idO int)*** - adiciona um professor a uma oferta cadastrada, ou seja, o professor passa a ministrar aquela oferta da disciplina. Para tal, é inserida uma linha na tabela *ministraroferta* passando os parâmetros: *in cpfProfessor varchar(14)*, o CPF do professor e *in idO int*, a oferta na qual o professor ministrará suas aulas.
- ***adicionarSolicitacaoDeProjetoDeUmAluno(in cpfAl varchar(14), in idP int)*** - adiciona uma linha na tabela *solicitacaoprojeto* quando o aluno manifesta interesse em participar de um projeto de pesquisa de um professor. São passados os parâmetros: *in cpfAl varchar(14)*, CPF do aluno, e *in idP int*, identificador do projeto.
- ***adicionarAlunoAoProjeto(in cpfAl varchar(14), in idP int)*** - adiciona uma linha na tabela *participarprojeto*. Isso ocorre quando o professor aceita a solicitação do aluno de participar no projeto por ele desenvolvido. São passados os parâmetros: *in cpfAl varchar(14)*, CPF do aluno, e *in idP int*, identificador do projeto.
- ***adicionarNotaAA aluno(in cpfAl varchar(14), in idOf int, in nota double)*** - é atualizada uma linha na tabela *nota*. Esse procedimento é usado para inserir uma determinada nota (valor da nota) a um aluno numa oferta de disciplina. São passados os parâmetros: *in cpfAl varchar(14)*, CPF do aluno; *in idOf int*, identificador da oferta de disciplina (turma) e *in nota double*, nota a ser inserida.

### 9.3 Funções

- ***buscarCpfProfessor(idP int) returns varchar(14) deterministic*** - função que busca o CPF de um determinado professor e o retorna, passando *idP int* (identificador da tabela *projetopesquisa*).
- ***buscarNomeProjeto(idP int) returns varchar(35) deterministic*** - função que retorna o nome de um projeto de pesquisa, sendo passado como parâmetro *idP int*.
- ***buscarNomeUsuario(cpfU varchar(14)) returns varchar(35) deterministic*** - função que retorna o nome de usuário dado como parâmetro o seu CPF.
- ***buscarIdDisciplina(idOf int) returns int deterministic*** - retorna o identificador de uma disciplina (na tabela, *idDisciplina*), sendo passado o parâmetro *idOf int*, que é o identificador de uma oferta de disciplina. Usada basicamente como função secundária para encontrar o nome da disciplina, que é feito pela função logo abaixo.
- ***buscarNomeDisciplina(idDisc int) returns varchar(35) deterministic*** - retorna o nome da disciplina dado *idDisc int* (identificador de disciplina) como seu parâmetro.

## 10 Instruções para rodar o programa

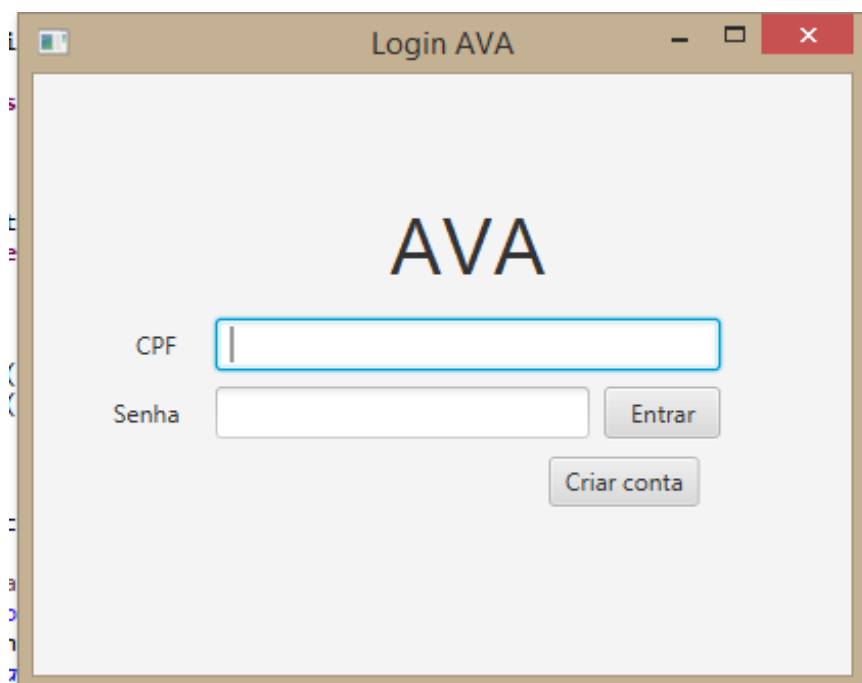
1. Abra o projeto numa IDE de sua escolha que rode Java (recomendamos [Eclipse](#))
2. Crie uma nova conexão em [MySQL Workbench 6.3 CE](#) e insira seu usuário e sua senha ou use uma conexão já estabelecida
3. Abra os arquivos `.sql`, encontrados dentro do projeto na pasta *ProjetoBanco* na conexão iniciada
4. Rode todos os scripts. Siga a seguinte ordem na hora de compilá-los: GerarTabelas.sql, PopularTabelas.sql, ConsultarTabelas.sql, Views.sql, calcularMédia.sql, calcularMédiaPorAluno.sql e, por fim, Triggers e Procedures.sql.
5. Após isso vá até a classe `com.ufrpe.ava.aspecto.ConexaoMySQL.java` e na linha: `"connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/ava", "root", " ");"`, substitua o `root` pelo nome do usuário posto na conexão e `" "` pela sua senha nessa mesma conexão.
6. Para ter acesso aos cadastros de usuários, cursos, disciplinas entre outros é preciso que se entre como administrador, passando na tela inicial da aplicação, o CPF *admin* e senha *admin*. Para outros usuários, que são professores e alunos, deve-se cadastrar primeiro, por meio do administrador, o usuário, para ter acesso às funcionalidades de matrículas, inserção e solicitação de projetos e avisos.
7. A aplicação necessita de três bibliotecas para seu funcionamento correto. Segue os links logo abaixo para sua instalação:

**OBS.:** Verificar se já se encontram no projeto. Se sim, não há necessidade de prosseguir o passo.

- [JavaFX SDK](#)
- [MySQL Connector Java](#)
- [AspectJ](#)

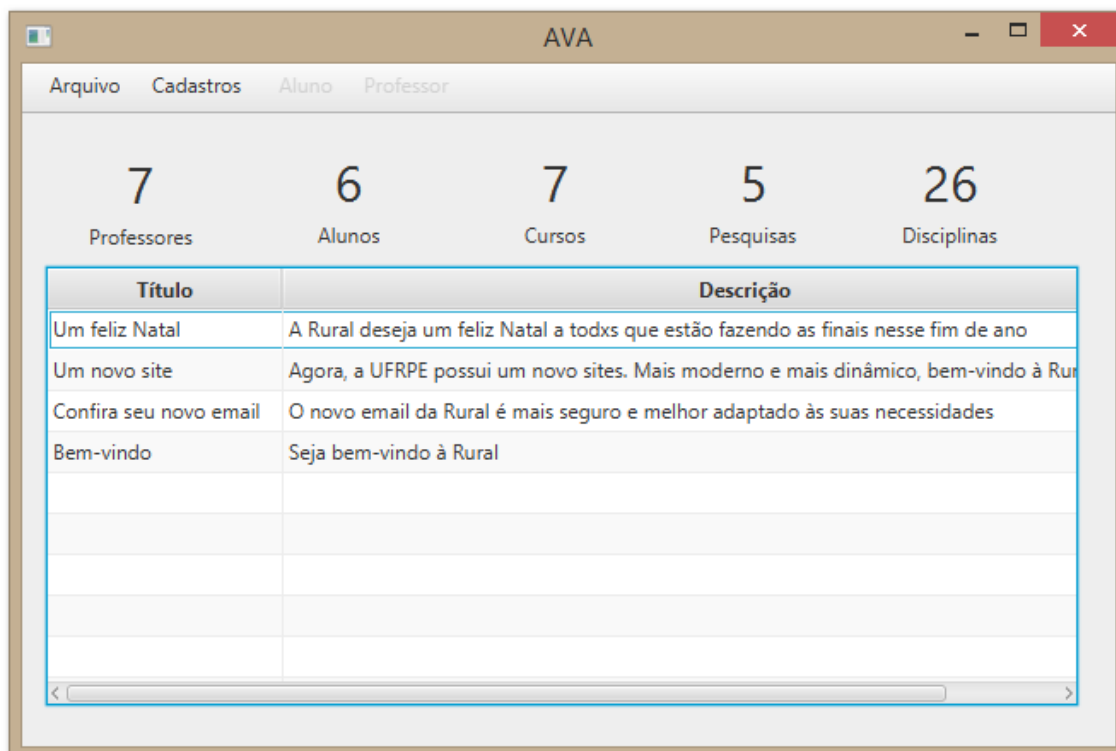
## 11 Principais Telas do Sistema

### TELA LOGIN



The image shows a login window titled "Login AVA". It features the "AVA" logo at the top center. Below the logo, there are two input fields: "CPF" and "Senha". To the right of the "Senha" field is an "Entrar" button. Below the "Entrar" button is a "Criar conta" button.

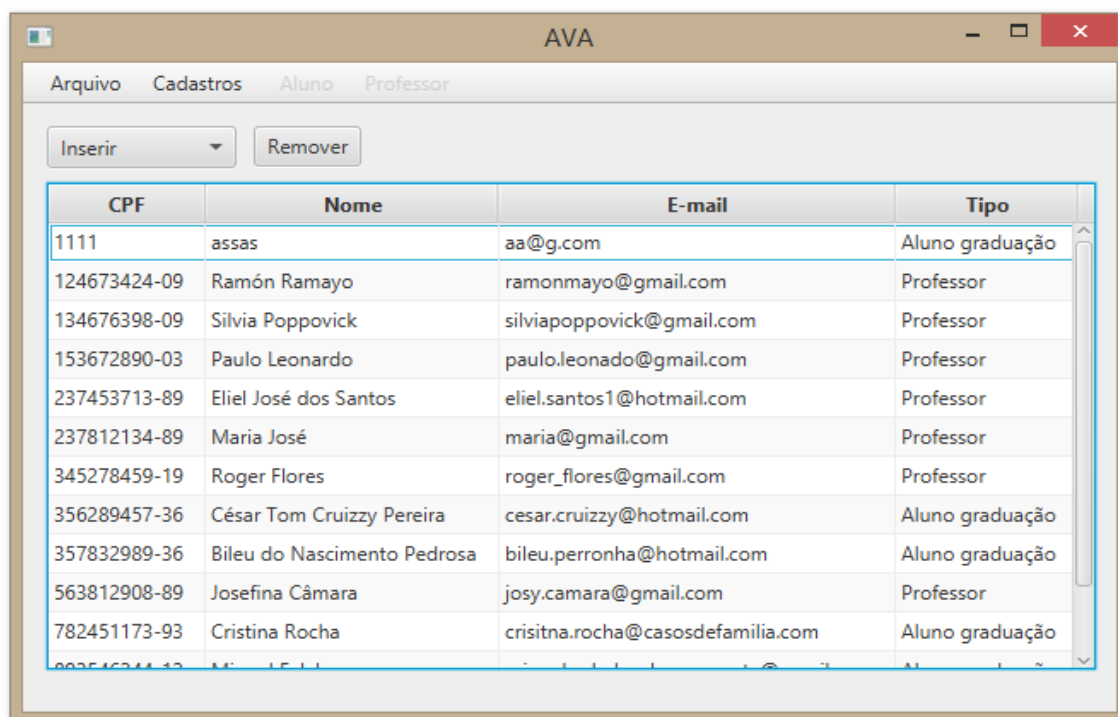
### TELA INICIO



The image shows the home screen of the AVA system. It has a menu bar with "Arquivo", "Cadastros", "Aluno", and "Professor". Below the menu bar, there are five statistics: 7 Professores, 6 Alunos, 7 Cursos, 5 Pesquisas, and 26 Disciplinas. Below these statistics is a table with two columns: "Título" and "Descrição".

Título	Descrição
Um feliz Natal	A Rural deseja um feliz Natal a todxs que estão fazendo as finais nesse fim de ano
Um novo site	Agora, a UFRPE possui um novo sites. Mais moderno e mais dinâmico, bem-vindo à Rural
Confira seu novo email	O novo email da Rural é mais seguro e melhor adaptado às suas necessidades
Bem-vindo	Seja bem-vindo à Rural

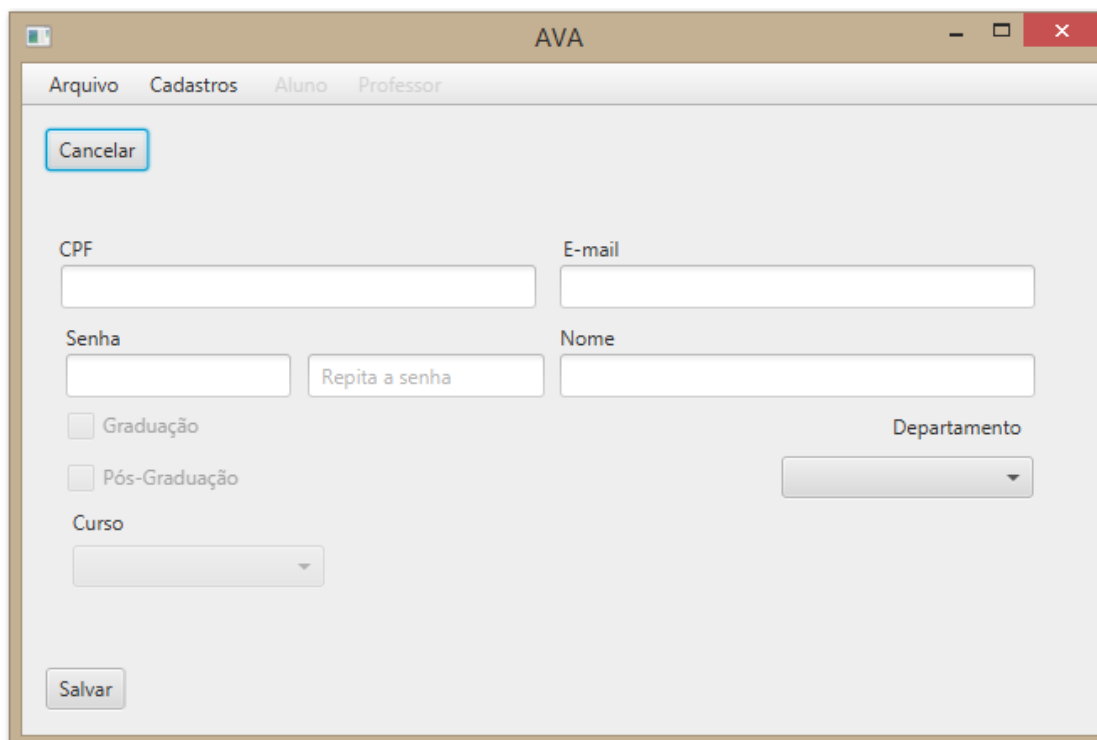
## TELA VISUALIZAÇÃO



A screenshot of the AVA application window showing the 'Arquivo' menu and the 'Cadastros' tab. The 'Aluno' sub-tab is selected, displaying a table of student records. The table has four columns: CPF, Nome, E-mail, and Tipo. The records are as follows:

CPF	Nome	E-mail	Tipo
1111	assas	aa@g.com	Aluno graduação
124673424-09	Ramón Ramayo	ramonmayo@gmail.com	Professor
134676398-09	Silvia Poppovick	silviapoppovick@gmail.com	Professor
153672890-03	Paulo Leonardo	paulo.leonado@gmail.com	Professor
237453713-89	Eliel José dos Santos	eliel.santos1@hotmail.com	Professor
237812134-89	Maria José	maria@gmail.com	Professor
345278459-19	Roger Flores	roger_flores@gmail.com	Professor
356289457-36	César Tom Cruizy Pereira	cesar.cruizy@hotmail.com	Aluno graduação
357832989-36	Bileu do Nascimento Pedrosa	bileu.perronha@hotmail.com	Aluno graduação
563812908-89	Josefina Câmara	josy.camara@gmail.com	Professor
782451173-93	Cristina Rocha	crisitna.rocha@casosdefamilia.com	Aluno graduação

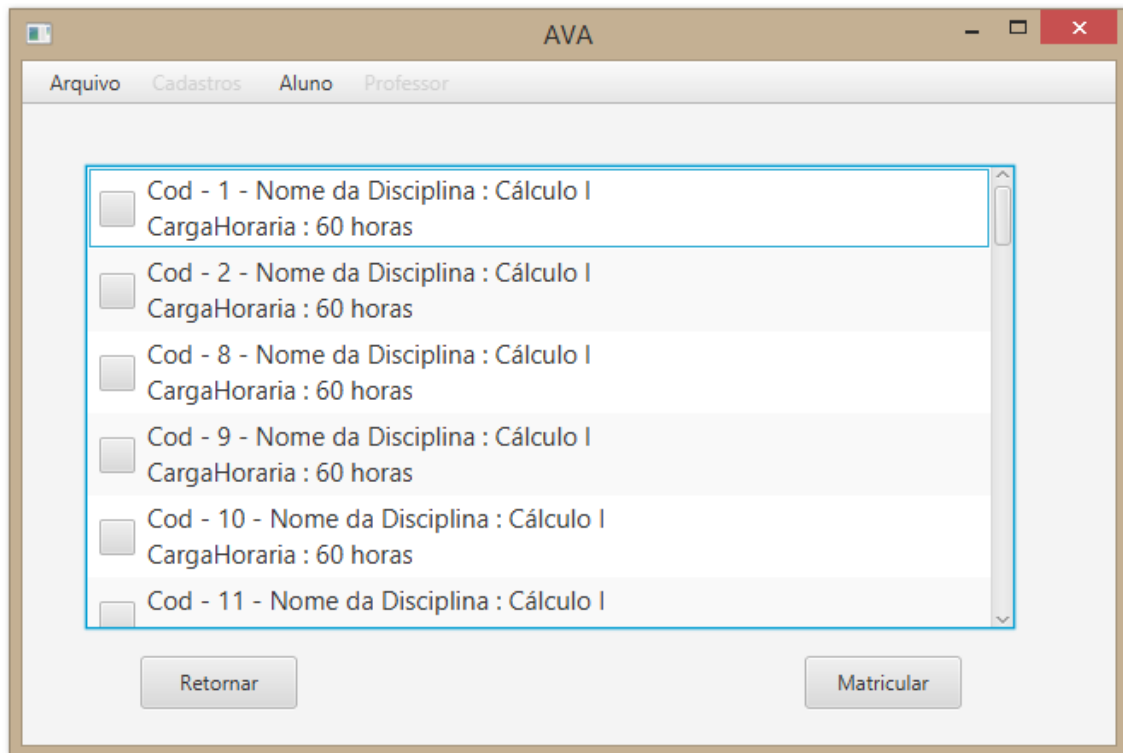
## TELA CADASTRO



A screenshot of the AVA application window showing the 'Arquivo' menu and the 'Cadastros' tab. The 'Aluno' sub-tab is selected, displaying a form for adding a new student record. The form includes the following fields and controls:

- CPF**: Text input field.
- E-mail**: Text input field.
- Senha**: Text input field.
- Repita a senha**: Text input field for password confirmation.
- Nome**: Text input field.
- Departamento**: Dropdown menu.
- Curso**: Dropdown menu.
- Gradação**: Radio button.
- Pós-Graduação**: Radio button.
- Cancelar**: Button to cancel the operation.
- Salvar**: Button to save the record.

## TELA MATRICULA

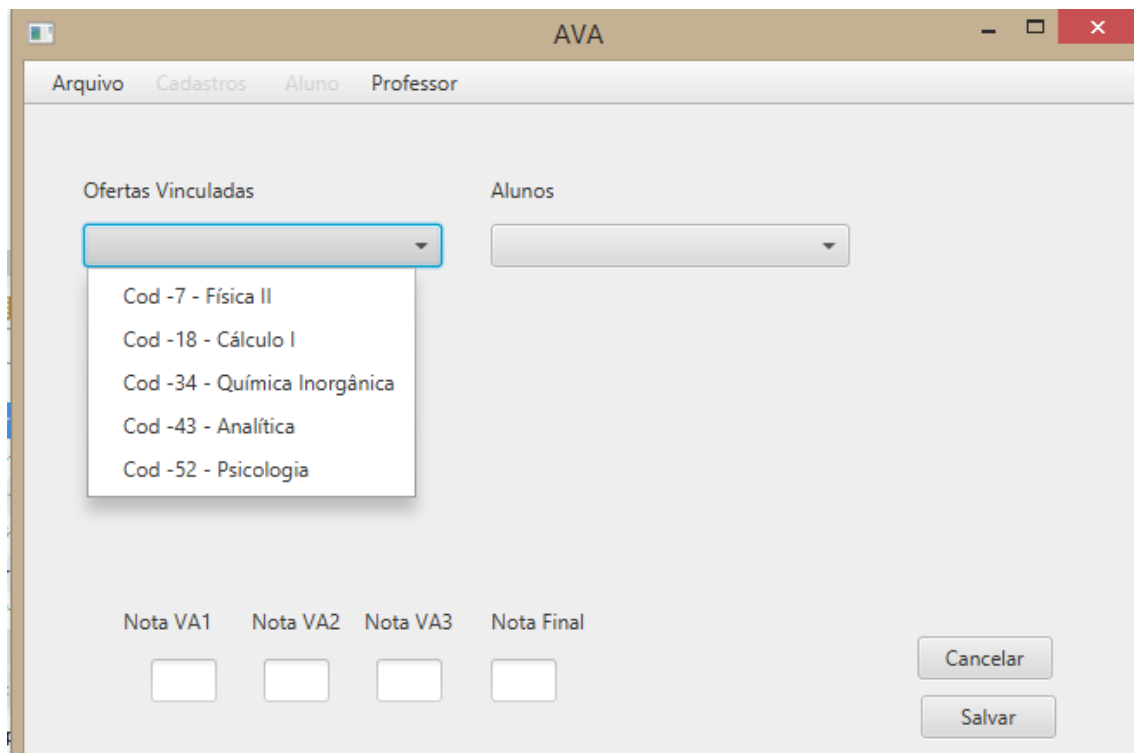


The screenshot shows the 'AVA' application window with a menu bar containing 'Arquivo', 'Cadastros', 'Aluno', and 'Professor'. The main area displays a list of six courses, each with an unchecked checkbox, its name, and a load of 60 hours. The courses are: Cod - 1 - Nome da Disciplina : Cálculo I, Cod - 2 - Nome da Disciplina : Cálculo I, Cod - 8 - Nome da Disciplina : Cálculo I, Cod - 9 - Nome da Disciplina : Cálculo I, Cod - 10 - Nome da Disciplina : Cálculo I, and Cod - 11 - Nome da Disciplina : Cálculo I. At the bottom, there are two buttons: 'Retornar' and 'Matricular'.

Cod	Nome da Disciplina	Carga Horária
Cod - 1	Nome da Disciplina : Cálculo I	60 horas
Cod - 2	Nome da Disciplina : Cálculo I	60 horas
Cod - 8	Nome da Disciplina : Cálculo I	60 horas
Cod - 9	Nome da Disciplina : Cálculo I	60 horas
Cod - 10	Nome da Disciplina : Cálculo I	60 horas
Cod - 11	Nome da Disciplina : Cálculo I	60 horas

Retornar Matricular

## TELA APLICAR NOTAS



The screenshot shows the 'AVA' application window with a menu bar containing 'Arquivo', 'Cadastros', 'Aluno', and 'Professor'. The main area has two dropdown menus: 'Ofertas Vinculadas' and 'Alunos'. The 'Ofertas Vinculadas' dropdown is open, showing a list of five courses: Cod -7 - Física II, Cod -18 - Cálculo I, Cod -34 - Química Inorgânica, Cod -43 - Analítica, and Cod -52 - Psicologia. Below the dropdowns, there are four input fields labeled 'Nota VA1', 'Nota VA2', 'Nota VA3', and 'Nota Final'. At the bottom right, there are two buttons: 'Cancelar' and 'Salvar'.

Ofertas Vinculadas Alunos

Cod -7 - Física II  
Cod -18 - Cálculo I  
Cod -34 - Química Inorgânica  
Cod -43 - Analítica  
Cod -52 - Psicologia

Nota VA1 Nota VA2 Nota VA3 Nota Final

Cancelar Salvar

## 12 Conclusão

Tantos os objetivos como também as funcionalidades na data de entrega do projeto foram cumpridas. Porém, é notável que ainda é um sistema aquém do AVA. Mas ainda, é válido lembrar que essa não foi a pretensão antes ou no decorrer do projeto para Paradigmas de Programação e Banco de Dados, pois é sabido que um sistema desse porte é necessário um maior tempo ou recurso humano. Com uma base sólida, talvez, possa se expandir o projeto e o desenvolvendo em outras disciplinas do curso de Ciência da Computação e o levando ao mesmo patamar da aplicação original, que deu base ao projeto.

Contudo, o projeto AVA conseguiu implementar todos os objetivos na qual foi focado durante seu desenvolvimento. Logo, chegada a data de sua entrega se conseguiu alcançar as expectativas estabelecidas no início do projeto.