

**Técnico de Informática – Sistemas 2019/2022**

2021/2022

**Prova de Aptidão Profissional**

GESRES 1.0

**Vítor Pereira Albernaz**

Ponta Delgada, março de dois mil e vinte e dois



**Técnico de Informática – Sistemas 2019/2022**

2021/2022

**Prova de Aptidão Profissional**

GESRES 1.0

**Formando:** Vítor Albernaz

**Orientador:** Duarte Cota

Ponta Delgada, março de dois mil e vinte e dois

**AGRADECIMENTOS**

|  |
| --- |
| Primeiramente queria agradecer à instituição da Escola de Novas Tecnologias dos Açores pela oportunidade de tirar o meu curso de nível IV nesta instituição.  Começando por agradecer à minha turma por toda a convivência durante estes incríveis três anos do curso.  De seguida agradecer a todos os formadores e professores que pelo meu progresso formativo passaram. Da área técnica os formadores Jorge Soares, Jorge Costa, Luís Carreira, Daniel Oliveira, Nuno Melo, Duarte Cota e Óscar Neto. Na área sociocultural e científica os professores/as Pedro Fonseca, Paulo Martinho, Mariana Silva, Henrique Medeiros, Renato Lameirão, Silvia Gouveia, Ana Pereira e Anabela Ferreira. Também a todo o trabalho do staff da escola, a dona Gabriela Viveiros, doutora Patrícia Ribeiro e ao doutor João Lima, que sempre fizeram o melhor possível pela escola.  Um especial obrigado ao formador Duarte Cota, que me ajudou sempre que precisava enquanto orientador da minha prova de aptidão profissional. Também à diretora de turma, Ana Pereira, por nos ajudar no processo de arranjar estágio e de preparar-nos para da melhor maneira para as nossas provas finais de curso.  Obrigado pela confiança! |

Assinatura

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Resumo:**

GESRES é um software principalmente de vendas num estabelecimento bar ou restaurante. Este, por sua vez, exerce alguns extras, tais como: ligação com um programa para a cozinha, gestão de stock e de faturas em tempo real, e controlo de funcionários.

Para o desenvolvimento do projeto foi utilizado softwares com fins de criar ambientes gráficos (SceneBuilder), ajudar e agilizar no processo de desenvolver a parte lógica da aplicação através do IDE Intellij (Ambiente de desenvolvimento integrado) utilizando a linguagem de programação JAVA com componentes do JavaFX SDK 17 e outras bibliotecas como mysql-connector para efetuar a conexão com a base de dados e *software*, criação e gestão da base de dados (MySQl Workbench 8.0 CE).

//Resultados finais

//conclusão

Índice geral:

[Índice geral: 6](#_Toc95508640)

[Índice de Imagens: 7](#_Toc95508641)

[Introdução: 8](#_Toc95508642)

[Desenvolvimento: 9](#_Toc95508643)

[Vista inicial- login 9](#_Toc95508644)

[Dashboards 10](#_Toc95508645)

[Dashboard- Admin 10](#_Toc95508646)

[Dashboard- Funcionário 13](#_Toc95508647)

[Dashboard- Cozinha 17](#_Toc95508648)

[Base de dados 18](#_Toc95508649)

[Estrutura base de dados 18](#_Toc95508650)

Índice de Imagens:

[Imagem 1 Modelo MVC 8](#_Toc93769155)

Introdução:

GESRES é programa de vendas e gestão de um estabelecimento de Bar e ou Restauração.

Este projeto tem como objetivos: a aplicação de conhecimentos obtidos nas aulas e/ou pesquisas na área da programação com interfaces gráficas, e gestão de bases de dados. Construir uma ferramenta de fácil utilização e aprendizagem da mesma. Também permitir que a mesma consiga gerir pedidos entre o setor de Bar e Cozinha, integrar um sistema de gestão de stock através dos pedidos dos clientes, e gerir a origem dos pedidos (estabelecimento, takeaway ou entregas ao domicílio, se existirem).

Um programa desses tem muita importância no mundo dos negócios, pois pode facilitar muito o trabalho dos empregados e até mesmo do administrador do estabelecimento.

Escolhi um projeto deste calibre pois sempre foi do meu interesse como era gerido um estabelecimento deste tipo.

Para o desenvolvimento do projeto foi utilizado um método de arquitetura MVC (“Model-View-Controller”) utilizando as linguagens Java e FXML (JavaFx).

MVC é um padrão de aquitetura de sofware responsavel por contribuir para a otimização da velocicade de resposta aos pedidos do utilizador.

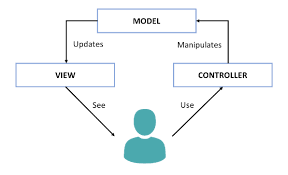


Imagem 1 Modelo MVC

Desenvolvimento:

Começando por desenhar o programa, que inicialmente ia ser um programa com três dahsboards, “Admin”, “Funcionário” e “Cozinha, porem decidi separar a dashboard da cozinha para um programa separado. Realizei as vistas do programa com a ajuda do SceneBuilder, ferramenta que gerou o código FXML para uma interface gráfica no meu programa. Ao todo foram criadas quinze vistas para o programa:

-Uma de login;

-Sete dedicadas à dashboard de “Admin”;

-Sete dedicadas à dashboard de “Funcionário”;

-Uma para o programa da “Cozinha”;

Todas as vistas foram feitas para uma medida de aproximadamente, ou não exceder, as 15 polegadas, medida vulgarmente utilizada nos monitores de estabelecimentos do tipo bar/restaurante,

Vista inicial- login

Vista composta por uma ComboBox, onde os seus valores são os users disponíveis para o programa, no caso “Admin” e “Funcionário”. E uma PasswordField para colocar a respetiva palavra-passe.

Esta tem como objetivo direcionar o utilizador, através de credenciais de acesso, para o destino desejado. Também é uma maneira de guardar informações restritas expostas na dashboard de Admin.

//imagem do login

Dashboards

Dashboard- Admin

A principal de Admin foi elaborada para: adicionar novos funcionários e remover caso necessário, controlar a atividade dos mesmos, e nesta temos acesso a opções expostas no MenuBar para: Controlo de stock, receitas e faturas. Informações que um empregado normal não terá acesso sem as credenciais de acesso.

Logo ao entrar na dashboard, deparamo-nos com três TextFields, para o primeiro nome, último nome, e número de funcionário, um DatePicker para a data de nascimento, e uma ComboBox para selecionar o sexo do funcionário. Estes servem para adicionar um funcionário após carregar no botão “Adicionar”, que irá é preparado um objeto da classe “Funcionarios” para ser introduzido na base de dados.

Também tem uma tabela, onde apresentam alguns dados do funcionário: nome completo, atividade, número de funcionário. Esta é preenchida logo ao entrar na vista através do método “initialize()”, que preenche a mesma com os respetivos valores presentes na base de dados.

A atividade é possível ser mudada através de uma a ComboBox e do botão “Atualizar” abaixo da tabela, que atualiza na base de dados a atividade através de um querry SQl de “UPDATE”, e seguidamente atualiza a tabela limpado todos os dados e voltando a colocá-los.

No MenuBar presente há dois menus: “STOCK” com dois submenus “Consultar Stock” e “Adicionar Produto” e “FATURAS” com três submenus “Diárias”, “Semanais” e “Mensais”.

//foto menu bar com opções todas abertas

No primeiro submenu “Consultar Stock”, com o objetivo do chefe ou o patrão do estabelecimento ir verificando e atualizando o stock disponível.

Ao carregar lá é aberta outra vista composta por uma tabela com o nome de cada produto e a respetiva quantidade em stock. Esta informação também é posta ao iniciar a vista através do método “initialize()”, que através de um querry SQl é apresentada a lista de produtos presente na base de dados sendo guardada num objeto da classe “Produtos”.

Para editar o stock de um produto nessa vista, seleciona-se o produto que pretende editar o stock na tabela, quando é selecionado aparece o nome do produto na primeira TextField e fica outra TextField para o utilizador poder por a nova quantidade de stock. Após carregar no botão “Editar” é atualizada a informação da tabela limpado todos os dados e voltando a colocá-los.

//print vista editar stock

No segundo submenu “Adicionar Produto”, este com o objetivo de adicionar produtos à base de dados para serem disponibilizados para venda.

Ao carregar lá é aberta outra vista composta por duas TextFields e duas ComboBoxs, estas para por o nome do produto a querer adicionar, o preço, selecionar o fornecedor e o tipo do produto, se é “Café”, “Sumos”,” Bebidas Alcoólicas”, “Cozinha”, “Snacks”, “Doces/Sobremesas” ou “Retalho”. Após carregar no botão “Concluir” os dados que são recolhidos dos quatro elementos são guardados num objeto da classe “Produtos” e de seguida adicionados na base de dados.

//print vista addProduto

A vista de faturas é usada nos três submenus do menu “Faturas”, onde o utilizador escolhe se quer ver as faturas diárias, semanais ou mensais, com o objetivo de ir controlando o número de pedidos do dia, semana ou do mês.

Consoante aquilo que o utilizador escolher ira aparecer na tabela do lado esquerdo todas as faturas correspondentes à escolha, mostrando apenas o número do pedido, a data e hora, e o valor total do pedido. Este ao selecionar uma linha e carregar no botão “Consultar”, mostrará todos os detalhes no lado direito todos os detalhes do pedido: produtos, respetivas quantidades e valor numa tabela, número do funcionário que registou o pedido e contribuinte se foi colocado.

//// print vista faturas

Dashboard- Funcionário

A dashboard funcionário é composta por um total de sete vistas diferentes. A primeira vista da mesma é composta por dez botões numerados de um a dez, correspondendo ao número das mesas do estabelecimento, esta fora elaborada com o intuito de controlar a disponibilidade das mesas do estabelecimento.

Ao carregar num dos botões é passado para a próxima vista, para mostrar os detalhes da mesa em questão, o número da mesa correspondente à mesa. Quando a mesa está livre o botão fica da cor verde, já quando está ocupada fica vermelho. E no final um botão servindo para registar os pedidos de balcão, ou seja, para quem não precisar de usufruir de mesa, este quando selecionado abre a vista introduzir o código de funcionário.

//print funcViewMesas

A vista de detalhes da mesa, criada para gerir cada mesa, é composta por uma tabela, para que caso o pedido esteja ativo mostra na mesma todo o pedido. Também é composta quatro botões: “Novo Pedido” que quando selecionado abre a vista introduzir o código de funcionário, “Editar Pedido” que é aberta novamente a vista de realizar o pedido novo, mas com a função de acrescentar ao pedido antigo, uma forma arranjada para não realizar uma vista diferente, “Despachar Mesa” quer servirá para por o botão, que corresponde ao numero da mesa, novamente verde assinalando que a mesa está livre, e por ultimo o botão “Pagamento” que é apresentado outra vista com o total do pedido escrito numa TextField, outra TextField para colocar o contribuinte do cliente caso queira, e uma ComboBox para selecionar o método de pagamento, apos carregar no botar “Finalizar” o pedido é posto noutra tabela da base de dados “faturas” e a mesa fica automaticamente livre.

//print vistaDetalhesPedidos e pagamento

A vista de introduzir um código de utilizador é sempre mostrada antes de iniciar um novo pedido, de modo a ficar registado o funcionário que fez o pedido. Esta é composta por uma TextField, que serve para introduzir um número, e após carregar no botão concluir, o número é validado verificando de existe o número na base de dados. Caso não exista é lançado um alerta referindo que o código é inexistente. Se existir o código é aberta a vista para realizar o pedido com o nome do funcionário correspondente ao código.

//print vista código func

A vista de pedidos, feita para o funcionário registar um pedido com o nível de dificuldade muito reduzido de se perceber o seu funcionamento, é composta por uma tabela onde o funcionário ira selecionar o produto que deseja continuar. Os produtos são mostrados na tabela conforme o tipo do produto desejado clicando num dos seis botões, colocados a cima da tabela (“Balcão”, “Sumos”, “Bebidas”, “Snacks”, “Doces”, “Cozinha”), estes vão buscar à base de dados todos os produtos correspondentes ao tipo. Para cada produto será obrigatório selecionar, na ComboBox, uma quantidade desejada para cada produto, de 1 a 9, e opcional uma observação ao produto. Também existe um campo, composto por uma TextField, para se for pedido produtos a retalho só indicar o valor. Após a seleção dos itens obrigatórios, o funcionário terá que pressionar no botão “Adicionar” que este guardará os dados do produto num objeto da classe Pedidos e vai mostrando noutra tabela o pedido a ser formado.

Caso o funcionário tenha, por engano, cometido um erro em algum produto, terá a opção de remover o produto do pedido, selecionando o mesmo e pressionando no botão “Eliminar”, este irá remover o produto que tinha sido gravado numa ObservableList.

Quando o pedido estiver concluído o funcionário terá de pressionar no botão “Finalizar Pedido” e este será guardado na base de dados.

//print vista pedido

Dashboard- Cozinha

A dashboard da cozinha, primeiramente era para ser realizada no programa principal, porém achei melhor criar outro projeto à parte para a realização do mesmo, com o intuito de mandar a parte do pedido que corresponda à cozinha.

Este é composto apenas por duas vistas, login que foi aproveitado o do programa principal, porém neste na ComboBox só ira aparecer a opção de fazer login no utilizador “Cozinha”, e a outra vista dedicada aos funcionários da cozinha do estabelecimento.

A segunda vista é composta por uma ComboBox, sendo atualizada automaticamente de 30 em 30 segundos, com os números dos pedidos, cujos foram validados na vista do funcionário se era necessário passar pela cozinha ou não, se fosse verificado que era então estes eram também adicionados a outra tabela na base de dados mostrando apenas os produtos correspondentes à cozinha. Após selecionar um dos pedidos da ComboBox, o funcionário terá que pressionar no botão “Ver Detalhes” e mostrará na tabela de detalhes o(s) prato(s) e as respetivas quantidades e observações, caso haja. A cima da tabela tem uma TextField onde mostrará o número do pedido em questão. Este também passará para uma tabela, tabela de confeção, onde estará a lista de pratos a ser confecionados.

Para finalizar o pedido o funcionário terá que selecionar na tabela de confeção o número de pedido que deseja finalizar, depois pressiona no botão “Finalizar”.

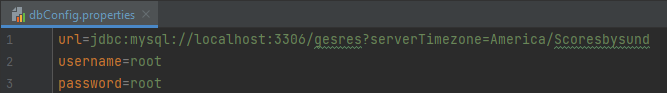
//print vista Cozinha

Base de dados

Para a base de dados utilizei duas ferramentas: MySQl Workbench, para a criação e configuração da base de dados, e para alojar a mesma utilizei um servidor virtual alojado na AWS educate.

A razão de alojar a base de dados num servidor virtual, não num servidor local, foi para poder trabalhar com mais facilidade entre escola e casa, visto que cada mudança feita seja em casa ou na escola ficava guardada no servidor e não na máquina em questão.

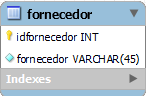
A base de dados foi ligada pelo ficheiro “dbConfig.properties”, este colocado na raiz do programa, onde neste foi colocado as credenciais de acesso à base de dados (“url”, “username”, “password”).



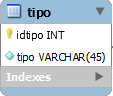
Estrutura base de dados

//print model

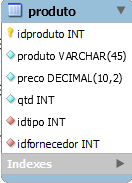
A tabela “fornedor” é composta tambem por duas colunas: “idfornecedor” e “fornecedor”, estas não podem receber valores null. A primeira coluna “idfornecedor” é uma chave primária do tipo INT. Já a segunda “fornecedor” sendo do tipo VARCHAR podendo receber no máximo quarenta e cinco caracteres.



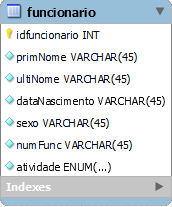
A tabela “tipo” é composta por duas colunas: “idtipo” e “tipo” , estas não podem receber valores null. Esta tabela foi criada para orientação de cada tipo dos produtos para venda. À semelhança da tabela anteriror, a coluna “idtipo” é uma chave primária do tipo INT. Já a segunda “tipo” sendo do tipo VARCHAR podendo receber no máximo quarenta e cinco caracteres.



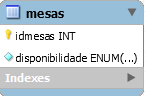
A tabela “produto” é composta por seis colunas: “idproduto”,” produto”,” preco”,” qtd”, “ idtipo”,” idfornecedor”. A coluna “idproduto” é uma chave primária do tipo INT, e é auto incremento. A segunda coluna “produto” é do tipo VARCHAR com um máximo de quarenta e cinco caracteres, para receber o nome do produto. A coluna “preco” é do tipo DECIMAL(10,2) , ou seja, pode receber números com dez números inteiros e duas casas decimais. A coluna “qtd” é do tipo INT recebendo a quantidade disponível em stock do produto em questão. As colunas “idtipo” e “idfornecedor” são chaves estrangeiras vindas das tabelas “tipo” e “fornecedor, respetivamente.



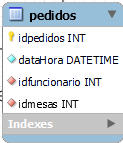
A tabela “funcionarios”. Esta é composta por sete colunas: “idfuncionario”, “primNome”, “ultiNome”, “dataNascimento”, “sexo”, “numFunc”, “atividade”, nenhuma podendo ter valores null. A primeira coluna “idfuncionario” é uma chave primária do tipo INT, e é auto incremento, ou seja, atribui um id ao funcionário automaticamente. As colunas “primNome”, “ultiNome”, “dataNascimento”, “sexo”, “numFunc”, são do tipo VARCHAR com um máximo de quarenta e cinco caracteres. A última coluna “ativadade” apenas está limitada a receber apenas dois valores,” Ausente” e “Presente.



De seguida a tabela “mesas”, esta apenas composta por duas colunas: “ifmesas” e “disponibilidade”, não podedendo ter valores null. A primeira coluna “idmesas” é uma chave primária do tipo INT. A segunda coluna “disponibilidade”, apenas está limitada a receber apenas dois valores,” True” e “False”, ou seja, se estiver “True” significa que a mesa está livre, caso contrário estará “False” assinalando que a mesma está ocupada.



A tabela “pedidos” é composta por quatro colunas: “ idpedidos”, ”dataHora”, “idfuncionario”, “idmesas”, não podedendo ter valores null. A coluna “idpedidos” é uma chave primária do tipo INT, e é auto incremento. A coluna “dataHora” é do tipo DATETIME guardando a data, hora, minuto e segundo do exato momento em que o pedido foi criado. As colunas “idfuncionario” e “idmesas” são de chaves estrangeiras vindas das tabelas “funcionário” e “mesas”, respetivamente.



A tabela “detalhespedidos” é composta por cinco