**Tarefa:**

Fazer com que a página de consulta já possa abrir exibindo as contas cadastradas para o mês atual (do primeiro ao ultimo dia do mês atual).

Criando uma classe auxiliar para retornar as das do primeiro dia do mês atual e do último dia do mês atual:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**package** br.com.cotiinformatica.helpers;

**import** java.text.SimpleDateFormat;

**import** java.util.Calendar;

**import** java.util.Date;

**import** java.util.GregorianCalendar;

**public** **class** DateHelper {

**public** **static** String getFirstDayOfMonth() {

Calendar calendar = Calendar.*getInstance*();

Date dataIni = **new** GregorianCalendar

(calendar.get(Calendar.***YEAR***),

calendar.get(Calendar.***MONTH***), 1).getTime();

**return** **new** SimpleDateFormat("yyyy-MM-dd").format(dataIni);

}

**public** **static** String getLastDayOfMonth() {

Calendar calendar = Calendar.*getInstance*();

Date dataFim = **new** GregorianCalendar

(calendar.get(Calendar.***YEAR***),

calendar.get(Calendar.***MONTH***),

calendar.getActualMaximum

(Calendar.***DAY\_OF\_MONTH***)).getTime();

**return** **new** SimpleDateFormat("yyyy-MM-dd").format(dataFim);

}

}

/controller/ConsultaContasController.java

Programando para que a página já abra carregando as contas do mês atual.

**package** br.com.cotiinformatica.controller;

**import** java.text.SimpleDateFormat;

**import** java.util.Date;

**import** java.util.List;

**import** javax.servlet.http.HttpServletRequest;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RequestMethod;

**import** org.springframework.web.servlet.ModelAndView;

**import** br.com.cotiinformatica.dtos.UsuarioDTO;

**import** br.com.cotiinformatica.entities.Conta;

**import** br.com.cotiinformatica.helpers.DateHelper;

**import** br.com.cotiinformatica.models.ConsultaContasModel;

**import** br.com.cotiinformatica.repositories.ContaRepository;

@Controller

**public** **class** ConsultaContasController {

@Autowired //inicialização automática (injeção de dependência)

**private** ContaRepository contaRepository;

@RequestMapping(value = "/admin/consulta-contas")

**public** ModelAndView consultaContas(HttpServletRequest request) {

//WEB-INF/views/admin/consulta-contas.jsp

ModelAndView modelAndView

= **new** ModelAndView("admin/consulta-contas");

ConsultaContasModel model = **new** ConsultaContasModel();

**try** {

model.setDataIni(DateHelper.*getFirstDayOfMonth*());

model.setDataFim(DateHelper.*getLastDayOfMonth*());

List<Conta> contas = obterConsultaDeContas

(model, request);

modelAndView.addObject("contas", contas);

}

**catch**(Exception e) {

modelAndView.addObject("mensagem",

"Falha ao consultar contas: "

+ e.getMessage());

}

modelAndView.addObject("model", model);

**return** modelAndView;

}

@RequestMapping(value = "/admin/consultar-contas",

method = RequestMethod.***POST***)

**public** ModelAndView consultarContas

(ConsultaContasModel model, HttpServletRequest request) {

ModelAndView modelAndView = **new** ModelAndView

("admin/consulta-contas");

**try** {

List<Conta> contas = obterConsultaDeContas

(model, request);

modelAndView.addObject("contas", contas);

}

**catch**(Exception e) {

modelAndView.addObject("mensagem",

"Falha ao consultar contas: " + e.getMessage());

}

modelAndView.addObject("model", model);

**return** modelAndView;

}

**private** List<Conta> obterConsultaDeContas(ConsultaContasModel

model, HttpServletRequest request) **throws** Exception {

//capturar os dados do usuário gravado em sessão

UsuarioDTO usuarioDto = (UsuarioDTO) request.getSession()

.getAttribute("usuario");

//capturar as datas selecionadas no formulário

Date dataIni = **new** SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataIni());

Date dataFim = **new** SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataFim());

//consultar as contas do usuário no banco de dados

**return** contaRepository.findByUsuarioAndData

(usuarioDto.getIdUsuario(), dataIni, dataFim);

}

}

<http://localhost:8082/sistemacontas/admin/consulta-contas>

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Desenvolvendo a página de dashboard:**

Mostrar gráficos e totais na tela de apresentação do sistema.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Testando as consultas no banco de dados:**

**select \* from usuario;**

**select \* from conta where idusuario = 1;**

**select sum(valor) from conta**

**where idusuario = 1**

**and data between '2023-02-01' and '2023-02-28'**

**and tipo = 1;**

**select sum(valor) from conta**

**where idusuario = 1**

**and data between '2023-02-01' and '2023-02-28'**

**and tipo = 2;**

/repositories/ContaRepository.java

**public** **double** sumByUsuarioAndData(Integer idUsuario, Date dataIni, Date dataFim, Integer tipo) **throws** Exception {

String query = "select sum(valor) from conta where idusuario=?

and data between ? and ? and tipo = ?";

Object[] params = {

idUsuario,

java.sql.Date.*valueOf*(**new** SimpleDateFormat

("yyyy-MM-dd").format(dataIni)),

java.sql.Date.*valueOf*(**new** SimpleDateFormat

("yyyy-MM-dd").format(dataFim)),

tipo

};

BigDecimal valor = jdbcTemplate.queryForObject

(query, params, BigDecimal.**class**);

**if**(valor != **null**)

**return** valor.doubleValue();

**else**

**return** 0;

}

Criando a página de dashboard:

/controller/DashboardController.java

**package** br.com.cotiinformatica.controller;

**import** java.text.SimpleDateFormat;

**import** java.util.Date;

**import** javax.servlet.http.HttpServletRequest;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.servlet.ModelAndView;

**import** br.com.cotiinformatica.dtos.UsuarioDTO;

**import** br.com.cotiinformatica.helpers.DateHelper;

**import** br.com.cotiinformatica.models.ConsultaContasModel;

**import** br.com.cotiinformatica.repositories.ContaRepository;

@Controller

**public** **class** DashboardController {

@Autowired

**private** ContaRepository contaRepository;

@RequestMapping(value = "/admin/dashboard")

**public** ModelAndView dashboard(HttpServletRequest request) {

// WEB-INF/views/dashboard.jsp

ModelAndView modelAndView

= **new** ModelAndView("admin/dashboard");

**try** {

ConsultaContasModel model

= **new** ConsultaContasModel();

model.setDataIni(DateHelper.*getFirstDayOfMonth*());

model.setDataFim(DateHelper.*getLastDayOfMonth*());

UsuarioDTO usuarioDto = (UsuarioDTO)

request.getSession().getAttribute("usuario");

Date dataIni = **new** SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataIni());

Date dataFim = **new** SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataFim());

//enviar as informações para a página

modelAndView.addObject("model", model);

modelAndView.addObject("total\_contas\_receber",

contaRepository.sumByUsuarioAndData(usuarioDto.getIdUsuario(), dataIni, dataFim, 1));

modelAndView.addObject("total\_contas\_pagar",

contaRepository.sumByUsuarioAndData(usuarioDto.getIdUsuario(), dataIni, dataFim, 2));

}

**catch**(Exception e) {

modelAndView.addObject("mensagem", "Falha ao

carregar dashboard: " + e.getMessage());

}

**return** modelAndView;

}

}

Desenhando a página de Dashboard:

/WEB-INF/views/admin/dashboard.jsp

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<%@ taglib uri=*"http://www.springframework.org/tags/form"* prefix=*"form"* %>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/core"* prefix=*"c"* %>

<%@ taglib uri=*"http://java.sun.com/jsp/jstl/fmt"* prefix=*"fmt"* %>

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

<!-- CDN da folha de estilos CSS do bootstrap -->

<link href=*"https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-*

*alpha1/dist/css/bootstrap.min.css"* rel=*"stylesheet"* />

<style>

**label***.error* { color: *#df4759*; }

**input***.error***,** **textarea***.error* { border: *2px solid #df4759*; }

</style>

</head>

<body>

<!-- menu superior da aplicação -->

<jsp:include page=*"/WEB-INF/views/admin/components/navbar.jsp"*></jsp:include>

<div class=*"container mt-3"*>

<div class=*"card"*>

<div class=*"card-body"*>

<h5>Dashboard</h5>

<p>Seja bem vindo ao sistema de gestão de contas.</p>

<hr/>

<div class=*"mb-2"*>

<strong>${mensagem}</strong>

</div>

<form id=*"form\_consulta"*

action=*"filtrar-dashboard"* method=*"post"*>

<div class=*"row mb-2"*>

<div class=*"col-md-3"*>

<form:input type=*"date"*

path=*"model.dataIni"* id=*"dataIni"* name=*"dataIni"* class=*"form-control"*/>

</div>

<div class=*"col-md-3"*>

<form:input type=*"date"*

path=*"model.dataFim"* id=*"dataFim"* name=*"dataFim"* class=*"form-control"*/>

</div>

<div class=*"col-md-6"*>

<input type=*"submit"* class=*"btn*

*btn-success"*

value=*"Filtrar Resultados"*/>

</div>

</div>

</form>

<div class=*"row mt-3 mb-3"*>

<div class=*"col-md-5"*>

<div class=*"alert alert-success mb-2"*>

<div class=*"row"*>

<div class=*"col-md-8"*>

Total de contas

a receber:

</div>

<div class=*"col-md-4"*>

<strong>

<fmt:formatNumber

value=*"*${total\_contas\_receber}*"* type=*"currency"*/>

</strong>

</div>

</div>

</div>

<div class=*"alert alert-danger mb-2"*>

<div class=*"row"*>

<div class=*"col-md-8"*>

Total de contas a

pagar:

</div>

<div class=*"col-md-4"*>

<strong>

<fmt:formatNumber

value=*"*${total\_contas\_pagar}

*"* type=*"currency"*/>

</strong>

</div>

</div>

</div>

<div class=*"alert alert-light mb-2"*>

<div class=*"row"*>

<div class=*"col-md-8"*>

Saldo:

</div>

<div class=*"col-md-4"*>

<strong>

<fmt:formatNumber value=*"*${total\_contas\_receber - total\_contas\_pagar}*"* type=*"currency"*/>

</strong>

</div>

</div>

</div>

</div>

<div class=*"col-md-7"*>

<div id=*"grafico"*></div>

</div>

</div>

</div>

</div>

</div>

<!-- CDN do arquivo javascript do bootstrap -->

<script src=*"https://cdn.jsdelivr.net/npm/bootstrap*

*@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js"*></script>

<!-- CDN do arquivo javascript do JQuery -->

<script src=*"https://code.jquery.com/jquery-3.6.3.min.js"*></script>

<!-- CDN dos arquivos da biblioteca JQuery Validation -->

<script src=*"https://cdnjs.cloudflare.com/ajax/libs/jquery-*

*validate/1.19.5/jquery.validate.min.js"*></script>

<script src=*"https://cdnjs.cloudflare.com/ajax/libs/jquery-*

*validate/1.19.5/additional-methods.min.js"*></script>

<script src=*"https://cdnjs.cloudflare.com/ajax/libs/jquery-*

*validate/1.19.5/localization/messages\_pt\_BR.min.js"*></script>

<!-- CDN dos arquivos da biblioteca Highcharts -->

<script src=*"https://code.highcharts.com/highcharts.js"*></script>

<script>

$(document).ready(**function**() {

$("#form\_consulta").validate({

rules: {

'dataIni' : { required: **true** },

'dataFim' : { required: **true** }

}

});

**var** dados = [];

dados.push(['Total de Contas a Pagar', ${total\_contas\_pagar}]);

dados.push(['Total de Contas a Receber',

${total\_contas\_receber}]);

**new** Highcharts.Chart({

chart: { type : 'pie', renderTo: 'grafico' },

title: { text : 'Total de contas a receber/pagar' },

plotOptions: {

pie: { innerSize: '60%' }

},

series: [ { data: dados } ],

colors: ['#dc3545', '#198754']

});

})

</script>

</body>

</html>

<http://localhost:8082/sistemacontas/admin/dashboard>

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

No controlador, vamos criar um método para consultar os valores do dashboard por um período de datas diferente:

package br.com.cotiinformatica.controller;

import java.text.SimpleDateFormat;

import java.util.Date;

import javax.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.servlet.ModelAndView;

import br.com.cotiinformatica.dtos.UsuarioDTO;

import br.com.cotiinformatica.helpers.DateHelper;

import br.com.cotiinformatica.models.ConsultaContasModel;

import br.com.cotiinformatica.repositories.ContaRepository;

@Controller

public class DashboardController {

@Autowired

private ContaRepository contaRepository;

@RequestMapping(value = "/admin/dashboard")

public ModelAndView dashboard(HttpServletRequest request) {

// WEB-INF/views/dashboard.jsp

ModelAndView modelAndView = new ModelAndView("admin/dashboard");

try {

ConsultaContasModel model = new ConsultaContasModel();

model.setDataIni(DateHelper.getFirstDayOfMonth());

model.setDataFim(DateHelper.getLastDayOfMonth());

UsuarioDTO usuarioDto = (UsuarioDTO) request.getSession()

.getAttribute("usuario");

Date dataIni = new SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataIni());

Date dataFim = new SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataFim());

//enviar as informações para a página

modelAndView.addObject("model", model);

modelAndView.addObject("total\_contas\_receber",

contaRepository.sumByUsuarioAndData

(usuarioDto.getIdUsuario(), dataIni, dataFim, 1));

modelAndView.addObject("total\_contas\_pagar",

contaRepository.sumByUsuarioAndData

(usuarioDto.getIdUsuario(), dataIni, dataFim, 2));

}

catch(Exception e) {

modelAndView.addObject

("mensagem", "Falha ao carregar dashboard: " + e.getMessage());

}

return modelAndView;

}

@RequestMapping(value = "/admin/filtrar-dashboard",

method = RequestMethod.POST)

public ModelAndView filtrarDashboard(ConsultaContasModel model,

HttpServletRequest request) {

ModelAndView modelAndView = new ModelAndView("admin/dashboard");

try {

UsuarioDTO usuarioDto = (UsuarioDTO) request.getSession()

.getAttribute("usuario");

Date dataIni = new SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataIni());

Date dataFim = new SimpleDateFormat

("yyyy-MM-dd").parse(model.getDataFim());

//enviar as informações para a página

modelAndView.addObject("model", model);

modelAndView.addObject("total\_contas\_receber",

contaRepository.sumByUsuarioAndData

(usuarioDto.getIdUsuario(), dataIni, dataFim, 1));

modelAndView.addObject("total\_contas\_pagar",

contaRepository.sumByUsuarioAndData

(usuarioDto.getIdUsuario(), dataIni, dataFim, 2));

}

catch(Exception e) {

modelAndView.addObject("mensagem",

"Falha ao carregar dashboard: " + e.getMessage());

}

return modelAndView;

}

}

<http://localhost:8082/sistemacontas/admin/dashboard>

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Filters**

São componentes de aplicação Java que conseguem interceptar a navegação do usuário em uma aplicação. Vamos utilizar os filters para interceptar cada navegação feita pelo usuário e verificar se o mesmo tem permissão para acessar a página desejada.

Em resumo, todas as navegações deste sistema que acessam alguma página dentro de um diretório /**admin** só serão permitidas se o usuário tiver uma sessão criada no navegador (está autenticado).

Linha do tempo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

package br.com.cotiinformatica.filters;

import java.io.IOException;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

public class AuthenticationFilter implements Filter {

public AuthenticationFilter() {

}

public void destroy() {

}

//Método executado sempre que uma página/rota é acessada no sistema

**public void doFilter(ServletRequest request, ServletResponse response,**

FilterChain chain)

throws IOException, ServletException {

**HttpServletRequest req = (HttpServletRequest) request;**

**HttpServletResponse resp = (HttpServletResponse) response;**

//verificando se o usuário esta acessando uma página/rota

//contida em /admin

**if(req.getServletPath().contains("admin/")) {**

**//verificar se o usuário não está autenticado (sessão)**

**if(req.getSession().getAttribute("usuario") == null) {**

**//redirecionar para a página inicial**

**//do sistema (autenticar)**

**resp.sendRedirect("/sistemacontas/");**

**}**

**}**

chain.doFilter(request, response);

}

public void init(FilterConfig fConfig) throws ServletException {

}

}

**Curiosidade:**

O Filter fica mapeado no arquivo: **/web.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<web-app version=*"2.4"* xmlns=*"http://java.sun.com/xml/ns/j2ee"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://java.sun.com/xml/ns/j2ee*

*http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd"*>

<display-name>sistemacontas</display-name>

<context-param>

<param-name>contextClass</param-name>

<param-value>

org.springframework.web.context.support

.AnnotationConfigWebApplicationContext

</param-value>

</context-param>

**<filter>**

**<description>**

**</description>**

**<display-name>AuthenticationFilter</display-name>**

**<filter-name>AuthenticationFilter</filter-name>**

**<filter-class>**

**br.com.cotiinformatica.filters.AuthenticationFilter**

**</filter-class>**

**</filter>**

**<filter-mapping>**

**<filter-name>AuthenticationFilter</filter-name>**

**<url-pattern>/\*</url-pattern>**

**</filter-mapping>**

<listener>

<listener-class>

org.springframework.web.context.ContextLoaderListener

</listener-class>

</listener>

<servlet>

<servlet-name>SpringDispatcher</servlet-name>

<servlet-class>

org.springframework.web.servlet.DispatcherServlet

</servlet-class>

<init-param>

<param-name>contextClass</param-name>

<param-value>

org.springframework.web.context.support

.AnnotationConfigWebApplicationContext

</param-value>

</init-param>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>br.com.cotiinformatica</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>SpringDispatcher</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

<session-config>

<session-timeout>30</session-timeout>

</session-config>

</web-app>

**Enviando o trabalho para o GITHUB:**

Samsung@DESKTOP-P9F6D9F MINGW64 ~/Desktop/COTI - Aulas/2023 - Java WebDeveloper SQS 18h as 22h (Início em 09.01)/workspace/sistemacontas (main)

**$ git add .**

Samsung@DESKTOP-P9F6D9F MINGW64 ~/Desktop/COTI - Aulas/2023 - Java WebDeveloper SQS 18h as 22h (Início em 09.01)/workspace/sistemacontas (main)

**$ git commit -m 'Dashboard, filtros e customizações'**

Samsung@DESKTOP-P9F6D9F MINGW64 ~/Desktop/COTI - Aulas/2023 - Java WebDeveloper SQS 18h as 22h (Início em 09.01)/workspace/sistemacontas (main)

**$ git push -u origin main**

Texto

Descrição gerada automaticamente