

**Framework Spring**

Maior framework para desenvolvimento de aplicações Java. Composto de diversas bibliotecas e módulos voltados para criação de aplicações Java, principalmente para Web.

Dentro do **Spring Framework**, temos 2 principais tecnologias para desenvolvimento de aplicações web:

* **Spring MVC**

Conjunto de bibliotecas do Spring para criação de projetos web através do padrão **Model**, **View** e **Controller**.

* **Spring Boot**

Conjunto de bibliotecas do Spring para criação de projetos web baseados em arquitetura de microserviços, ou seja, projetos que funcionam como **APIs REST**.

Além destas tecnologias para desenvolvimento web, o **Spring Framework** também possui módulos para:

* Envio de emails
* Acesso a banco de dados, através da JPA e Hibernate
* Etc.

Escopo do projeto:

Sistema de controle de finanças

* É preciso desenvolver uma aplicação que permita a um usuário gerenciar suas contas e movimentações financeiras. O sistema deverá permitir:
  + O usuário deverá criar uma conta de acesso no sistema
  + O usuário, através da sua conta, poderá fazer um login na aplicação
  + O sistema deverá permitir ao usuário cadastrar suas contas, consultá-las, editá-las e exclui-las
  + As contas poderão ser a pagar ou a receber e deverão ter:
    - Nome, Data, Valor, Tipo (Pagar ou Receber), Observações e Categoria (que poderá ser cadastrado pelo usuário)
  + O sistema deverá gerar relatórios e dashboards para o usuário consolidando dados de suas contas
  + O sistema também deverá permitir ao usuário alterar sua senha de acesso.

**Parte 01: Preparando o ambiente**

* Criando o banco de dados do projeto
* Criar um repositório no GITHUB para publicar o código fonte do projeto

**PgAdmin**

Nome do banco de dados: **bd\_sistemacontas**

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Criando um repositório no GITHUB para publicar**

**o código fonte do projeto:**

<https://github.com/>

Tela de computador com página de internet informando algo

Descrição gerada automaticamente com confiança média

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Repositórios no GitHub**

Recurso através do qual podemos publicar o código fonte de um projeto. Para cada projeto que iremos salvar no GITHUB precisamos criar um repositório.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

<https://github.com/smendescoti/sistemacontas>

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Parte 02: Criando o projeto**

Eclipse IDE

Desenho de um animal

Descrição gerada automaticamente com confiança média

**Apache Maven**

Framework Java utilizado para gerenciar os pacotes e bibliotecas utilizados em um projeto. Utiliza um arquivo de configuração no projeto para que possamos mapear quais bibliotecas serão instaladas na aplicação.

Este arquivo chama-se **/pom.xml**

Inclusive, o Maven já possui um recurso para criar projetos Java com o mínimo de configuração necessária (Hello World)

Sendo assim, vamos utilizar o Maven para criar um projeto Hello World baseado em Spring MVC.

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Selecione um Archetype**

Aqui, iremos escolher o tipo de projeto que o Maven irá criar.

Selecione o tipo: **spring-mvc-archetype**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Preenchendo as informações do projeto:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Word

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Para que possamos executarmos o projeto, precisamos de um servidor local que possa “rodar” a aplicação. Este servidor será o **Apache Tomcat**.

**Apache Tomcat**

Servidor utilizado para executar projetos Java Web.

<https://tomcat.apache.org/>



Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

<https://tomcat.apache.org/download-90.cgi>

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Exibindo a janela de configuração de servidores do eclipse:

**Windows / Show View / Other**

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Configurando o servidor:

Interface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Selecione a pasta do servidor:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Adicionando o projeto no servidor:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Word

Descrição gerada automaticamente

**Configurando a porta para execução do projeto:**

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Antes de executar o projeto, vamos configurar em qual navegador ele será executado:

Interface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Executando o projeto:

Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Log de execução do tomcat:**

Texto

Descrição gerada automaticamente

<http://localhost:8081/sistemacontas/>

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

/WEB-INF/views/**home.jsp**

Página inicial do projeto (Está configurada como página de inicialização do projeto). As páginas web do projeto Java possuem extensão   
**JSP – Java Server Pages**.

<%@page contentType=*"text/html"* pageEncoding=*"UTF-8"*%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"*>

<title>Home</title>

</head>

<body>

<h1>Hello World!</h1>

<p>This is the homepage!</p>

</body>

</html>

**Podemos agora já publicar o código**

**fonte do projeto no GITHUB:**

\*\* Para que possamos publicar o projeto no GITHUB, precisamos criar alguns arquivos de configuração no projeto:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**.gitignore**

Arquivo de configuração onde vamos definir quais pastas   
e arquivos do projeto não serão enviados para o GITHUB.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

# Created by https://www.gitignore.io/api/eclipse

### Eclipse ###

.metadata

bin/

tmp/

\*.tmp

\*.bak

\*.swp

\*~.nib

local.properties

.settings/

.loadpath

.recommenders

# External tool builders

.externalToolBuilders/

# Locally stored "Eclipse launch configurations"

\*.launch

# PyDev specific (Python IDE for Eclipse)

\*.pydevproject

# CDT-specific (C/C++ Development Tooling)

.cproject

# Java annotation processor (APT)

.factorypath

# PDT-specific (PHP Development Tools)

.buildpath

# sbteclipse plugin

.target

# Tern plugin

.tern-project

# TeXlipse plugin

.texlipse

# STS (Spring Tool Suite)

.springBeans

# Code Recommenders

.recommenders/

# Scala IDE specific (Scala & Java development for Eclipse)

.cache-main

.scala\_dependencies

.worksheet

### Eclipse Patch ###

# Eclipse Core

.project

# JDT-specific (Eclipse Java Development Tools)

.classpath

# End of https://www.gitignore.io/api/eclipse

**README.md**

(Arquivo de instruções do projeto)

Arquivo utilizado pelo GITHUB para criarmos uma página de apresentação do projeto, falando sobre o sistema e explicando o que é necessário para executar a aplicação.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Registrando o usuário do GITHUB:**

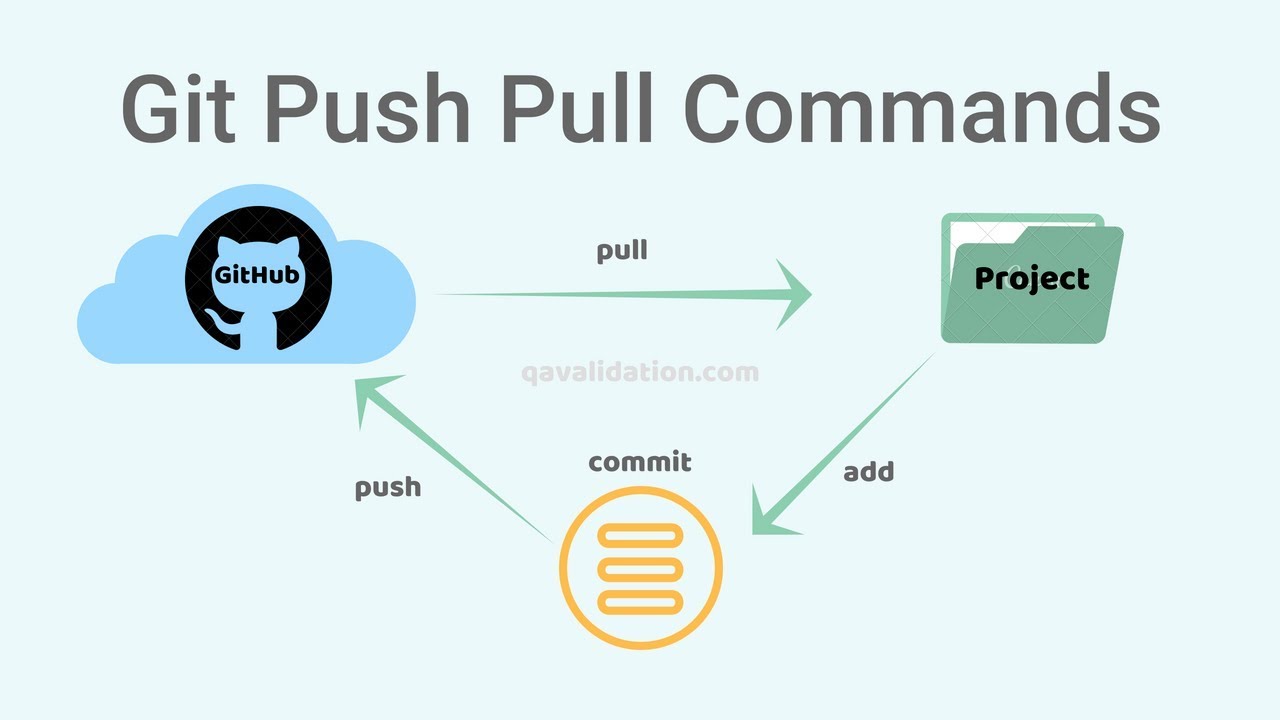
Samsung@DESKTOP-P9F6D9F MINGW64 ~/Desktop/COTI - Aulas/2023 - Java WebDeveloper SQS 18h as 22h (Início em 09.01)/workspace/sistemacontas

**$ git config --global user.name 'smendescoti'**

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

**$ git config --global user.email 'sergio.coti@gmail.com'**

**Em seguida, vamos iniciar um repositório do GIT na pasta do projeto:**



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

**$ git init**

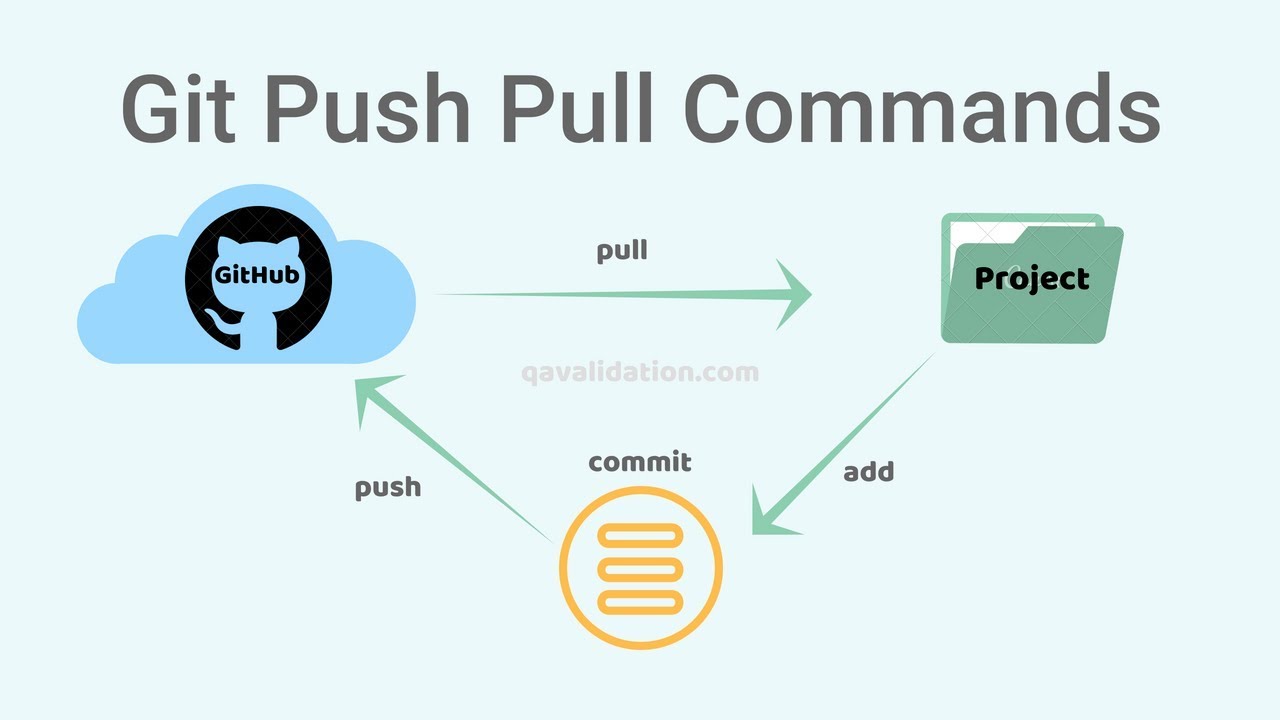
Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**COMMIT**

Nome dado ao processo de enviar arquivos do projeto local para o repositório do GITHUB. Em um commit podemos enviar desde muitos arquivos e pastas até somente alterações pequenas feitas no projeto.

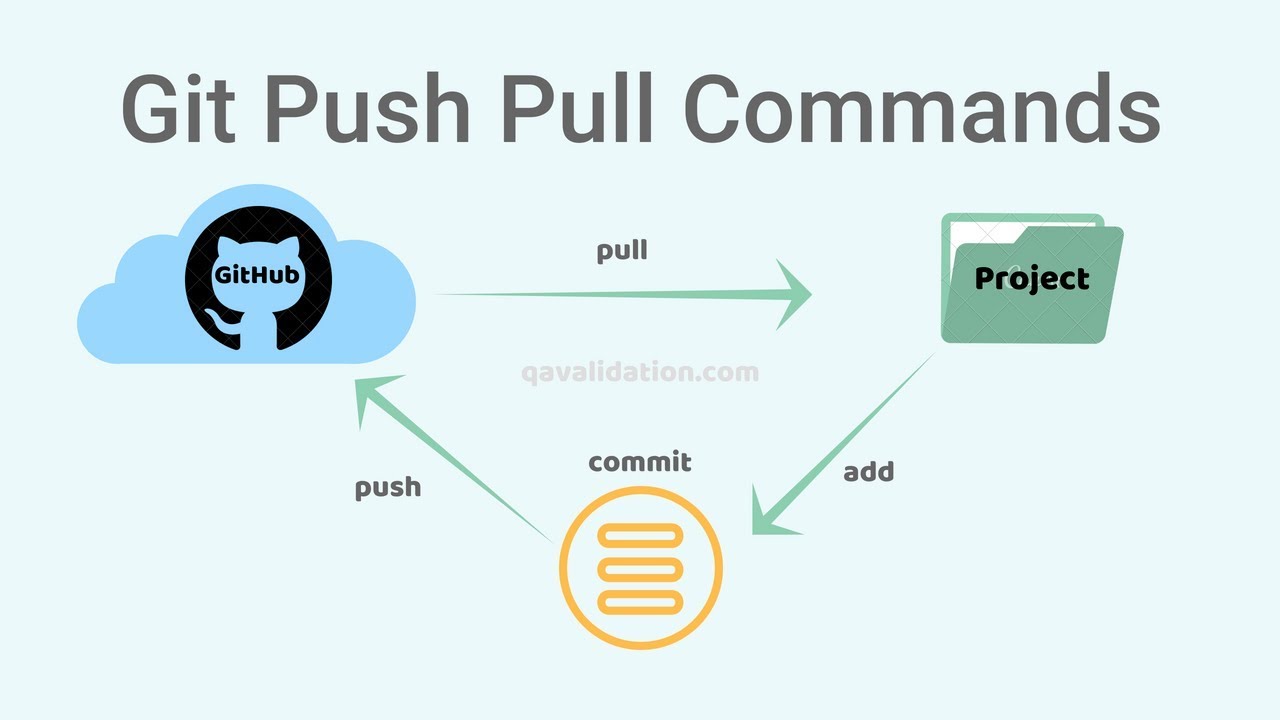
* Primeiro, vamos adicionar todos os arquivos do projeto no COMMIT.



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

**$ git add .**

* Em seguida, vamos dar um nome para o COMMIT.



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

**$ git commit -m 'first commit'**

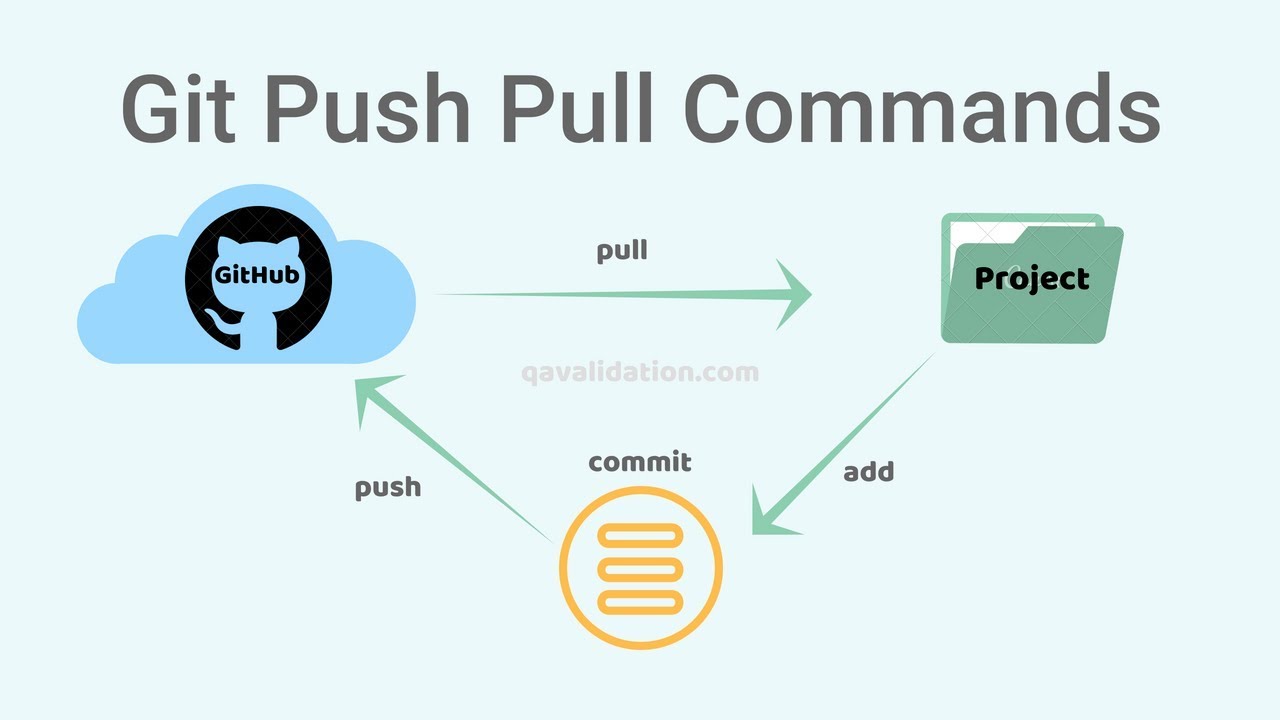
Texto

Descrição gerada automaticamente

**PUSH**

Comando para fazer o upload do conteúdo

de um COMMIT para o repositório do GITHUB:



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

**$ git branch -M main**

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

**$ git remote add origin https://github.com/smendescoti/sistemacontas.git**

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

<https://github.com/smendescoti/sistemacontas>

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

<https://chrome.google.com/webstore/detail/octotree-github-code-tree/bkhaagjahfmjljalopjnoealnfndnagc>

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Voltando ao projeto:**

Instalando bibliotecas que precisamos para desenvolver o projeto:

* Driver JDBC para conexão com o PostGreSQL
* Lombok

/**pom.xml**

Principal arquivo de configuração do framework MAVEN. É utilizado para que possamos configurar quais bibliotecas serão instaladas no nosso projeto.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Adicionando a biblioteca do POSTGRESQL:**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

<https://mvnrepository.com/artifact/org.postgresql/postgresql/42.5.1>

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Copie o XML para incluir a referência da biblioteca no /pom.xml

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Adicionando a biblioteca do LOMBOK:**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

<https://mvnrepository.com/artifact/org.projectlombok/lombok/1.18.24>

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

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

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>sistemacontas</groupId>

<artifactId>sistemacontas</artifactId>

<version>1.0</version>

<packaging>war</packaging>

<name>sistemacontas</name>

<url>http://maven.apache.org</url>

<properties>

<java.version>1.6</java.version>

<spring.version>3.1.0.RELEASE</spring.version>

<cglib.version>2.2.2</cglib.version>

</properties>

<dependencies>

<!-- Spring core & mvc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>${spring.version}</version>

<type>jar</type>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-test</artifactId>

<version>${spring.version}</version>

<type>jar</type>

<scope>test</scope>

</dependency>

<!-- CGLib for @Configuration -->

<dependency>

<groupId>cglib</groupId>

<artifactId>cglib-nodep</artifactId>

<version>${cglib.version}</version>

<scope>runtime</scope>

</dependency>

<!-- Servlet Spec -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>2.4</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>javax.servlet.jsp</groupId>

<artifactId>jsp-api</artifactId>

<version>2.1</version>

<scope>provided</scope>

</dependency>

**<!--** [**https://mvnrepository.com/artifact**](https://mvnrepository.com/artifact)

**/org.postgresql/postgresql -->**

**<dependency>**

**<groupId>org.postgresql</groupId>**

**<artifactId>postgresql</artifactId>**

**<version>42.5.1</version>**

**</dependency>**

**<!--** [**https://mvnrepository.com/artifact**](https://mvnrepository.com/artifact)

**/org.projectlombok/lombok -->**

**<dependency>**

**<groupId>org.projectlombok</groupId>**

**<artifactId>lombok</artifactId>**

**<version>1.18.24</version>**

**<scope>provided</scope>**

**</dependency>**

</dependencies>

<repositories>

<repository>

<id>springsource-milestones</id>

<name>SpringSource Milestones Proxy</name>

<url><https://oss.sonatype.org/content>

/repositories/springsource-milestones</url>

</repository>

</repositories>

<build>

<finalName>sistemacontas</finalName>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>2.0.2</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**Parte 03: Modelando as entidades do projeto**

Criando as classes JavaBean para definir a primeira versão do modelo de entidades do sistema:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**package** br.com.cotiinformatica.enums;

**public** **enum** TipoConta {

***PAGAR***,

***RECEBER***

}

**JavaBeans**

Padrão Java para construção de classes que tem como objetivo modelar entidades de um sistema (modelo de dados). Essas classes são criadas em Java seguindo o padrão:

* Atributos privados
* Construtor sem entrada de argumentos
* Construtor com entrada de argumentos (sobrecarga de métodos)
* Métodos de encapsulamento
  + Setters
  + Getters
* Sobrescrita de métodos da classe Object

**package** br.com.cotiinformatica.entities;

**import** java.util.Date;

**import** br.com.cotiinformatica.enums.TipoConta;

**import** lombok.AllArgsConstructor;

**import** lombok.Getter;

**import** lombok.NoArgsConstructor;

**import** lombok.Setter;

**import** lombok.ToString;

**@Setter**

**@Getter**

**@NoArgsConstructor**

**@AllArgsConstructor**

**@ToString**

**public** **class** Conta {

**private** Integer idConta;

**private** String nome;

**private** Double valor;

**private** Date data;

**private** TipoConta tipo;

**private** String observacoes;

**private** Usuario usuario;

}

**package** br.com.cotiinformatica.entities;

**import** java.util.List;

**import** lombok.AllArgsConstructor;

**import** lombok.Getter;

**import** lombok.NoArgsConstructor;

**import** lombok.Setter;

**import** lombok.ToString;

@Setter

@Getter

@NoArgsConstructor

@AllArgsConstructor

@ToString

**public** **class** Usuario {

**private** Integer idUsuario;

**private** String nome;

**private** String email;

**private** String senha;

**private** List<Conta> contas;

}

**Script para criação das tabelas no banco de dados:**

**create** **table** usuario(

idusuario serial **primary** **key**,

nome **varchar**(150) **not** **null**,

email **varchar**(100) **not** **null**,

senha **varchar**(50) **not** **null**

);

**create** **table** conta(

idconta serial **primary** **key**,

nome **varchar**(150) **not** **null**,

valor **decimal**(18,2) **not** **null**,

data **date** **not** **null**,

tipo **integer** **not** **null**,

observacoes **varchar**(500) **not** **null**,

idusuario **integer** **not** **null**,

**foreign** **key**(idusuario) **references** usuario(idusuario)

);

**Executando no PostGreSQL:**

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**Fazendo o commit do 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 'Entidades e script do banco de dados'**

**Fazendo o PUSH:**

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

<https://github.com/smendescoti/sistemacontas>

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente