



Residência em Software

Residência em Tecnologia da Informação e Comunicação SpringBoot: Persistência e JPA

Professor:

Alvaro Degas Coelho



INSTITUIÇÃO EXECUTORA



COORDENADORA



APOIO



Para Começar

- Estamos acessando dados diretamente da aplicação
- Ou de arquivos
- Agora queremos usufruir das vantagens de um SGBD
- Como nosso BD será relacional, usaremos uma ORM

JPA

• Java Persistence API

- Especificação de acesso ao BD
- Padrão comum para acesso a diferentes BDs
 - Mapeamento Objeto-Relacional (beans de entidade)
 - Hibernate
 - Eclipselink
 - ...
 - Modelo relacional: Java EE

JPA no pom.xml



```
14 <name>redesocial</name>
15 <description>Demo project for Spring Boot</description>
16 <properties>
17   <java.version>17</java.version>
18 </properties>
19 <dependencies>
20   <dependency>
21     <groupId>org.springframework.boot</groupId>
22     <artifactId>spring-boot-starter-web</artifactId>
23   </dependency>
24
25
26
27   <dependency>
28     <groupId>org.springframework.boot</groupId>
29     <artifactId>spring-boot-starter-data-jpa</artifactId>
30   </dependency>
31
32
33
34
35
36   <dependency>
37     <groupId>org.springframework.boot</groupId>
38     <artifactId>spring-boot-starter-test</artifactId>
39     <scope>test</scope>
40   </dependency>
41
```

Pode copiar do arquivo de
configurações de nosso Drive

Pela Internet pode dar trabalho:
compatibilidades

SGBD

- É necessário adicionar as dependências
- Serão “encaixadas” no JPA
- Múltiplas opções
 - Postgres, MySQL, Oracle, SQLServer, DB2, etc.
 - H2
 - Disponível, roda na memória

H2 no pom.xml

```
*redesocial/... X application.... RedesocialA... HelloControl... User.java
14 <name>redesocial</name>
15 <description>Demo project for Spring Boot</description>
16 <properties>
17   <java.version>17</java.version>
18 </properties>
19 <dependencies>
20   <dependency>
21     <groupId>org.springframework.boot</groupId>
22     <artifactId>spring-boot-starter-web</artifactId>
23   </dependency>
24
25
26
27   <dependency>
28     <groupId>org.springframework.boot</groupId>
29     <artifactId>spring-boot-starter-data-jpa</artifactId>
30   </dependency>
31
32   <dependency>
33     <groupId>com.h2database</groupId>
34     <artifactId>h2</artifactId>
35     <scope>runtime</scope>
36   </dependency>
37
38
39
40
41   <dependency>
42     <groupId>org.springframework.boot</groupId>
```

Pode copiar do arquivo de
configurações de nosso Drive

Pela Internet pode dar trabalho:
compatibilidades

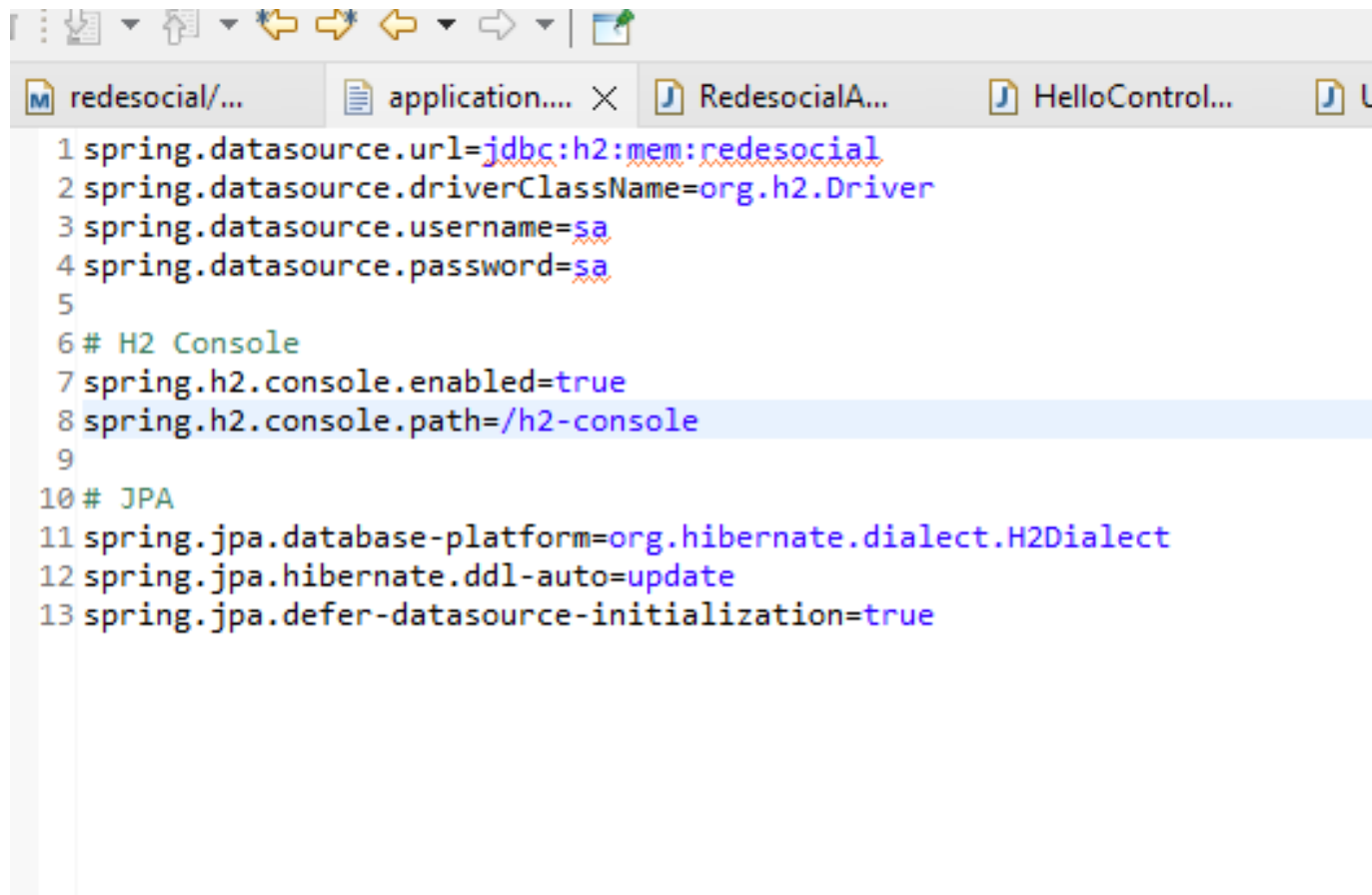
Para usar o SGBD

- Depois que as dependências forem baixadas
- Usar o SGBD
 - Parâmetros de conexão
 - Configuração do JPA para o SGBD
- Utilitários
 - Console para gerenciamento do SGBD

Configuração H2

- Arquivo application.properties
 - Pasta resources
- O que precisamos configurar
 - SGBD
 - Driver, URL, Username, Password
 - JPA
 - Classe p/ comunicar Hibernate e BD (Dialect)
 - Atualizar automaticamente tabelas do BD a partir das entidades Springboot
 - Console BD
 - Habilitar e definir o endereço

H2 no application.properties



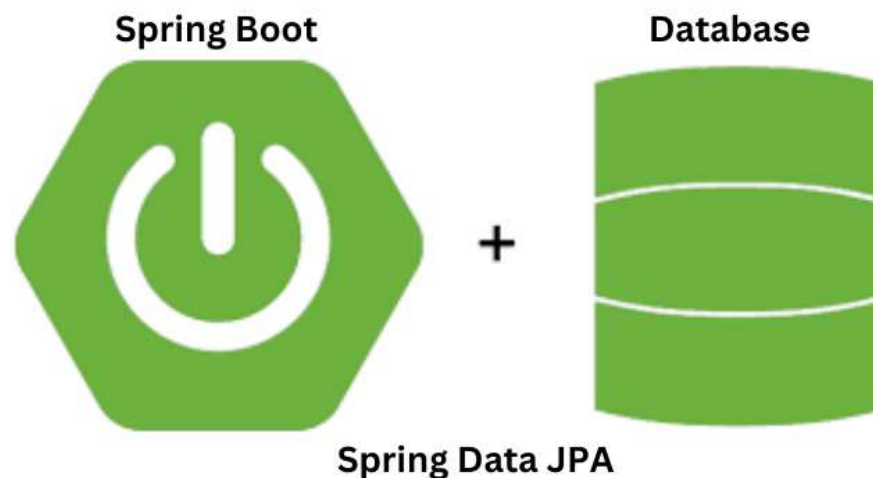
```
1 spring.datasource.url=jdbc:h2:mem:redesocial
2 spring.datasource.driverClassName=org.h2.Driver
3 spring.datasource.username=sa
4 spring.datasource.password=sa
5
6 # H2 Console
7 spring.h2.console.enabled=true
8 spring.h2.console.path=/h2-console
9
10 # JPA
11 spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
12 spring.jpa.hibernate.ddl-auto=update
13 spring.jpa.defer-datasource-initialization=true
```

Pode copiar do arquivo de
configurações de nosso Drive

Pela Internet pode dar trabalho:
compatibilidades

Usaremos uma abordagem ORM

- Tabelas criadas a partir das Entidades
- SpringBoot -> JPA (Hibernate) -> H2



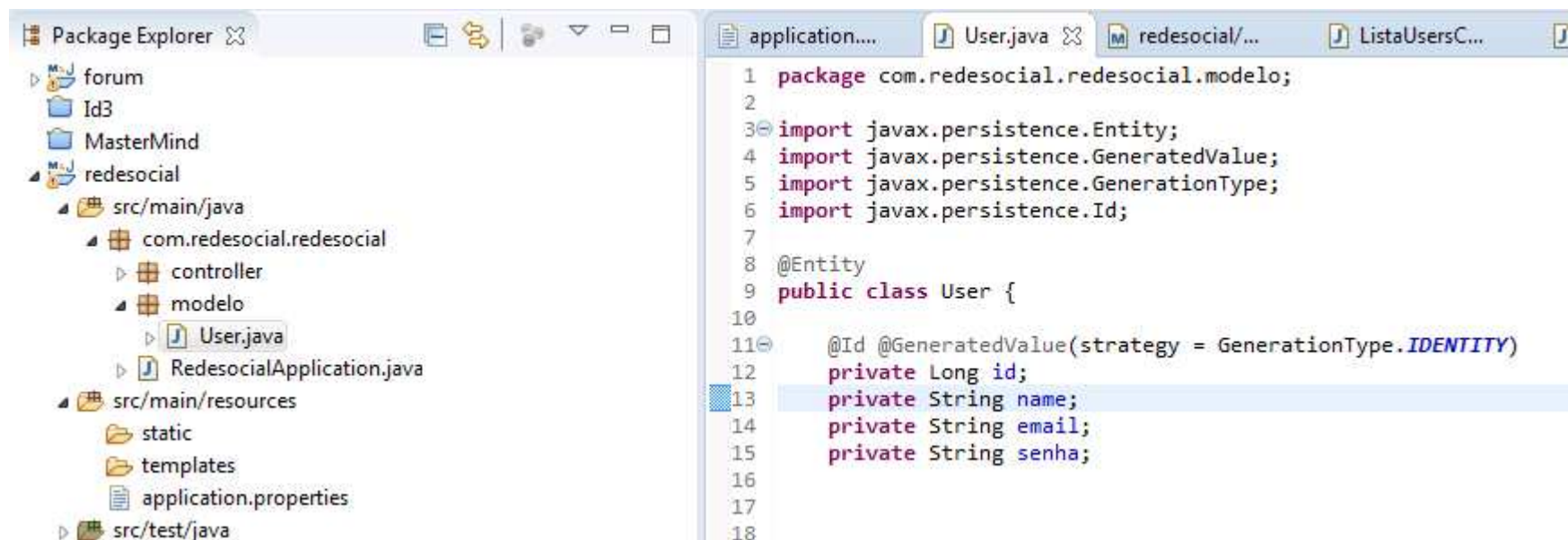
No nosso exemplo

- Vamos tornar a classe User uma entidade
- O JPA deverá construir esta tabela dentro do SGBD
- Se alteramos a classe, o JPA altera a tabela
- Se alteramos a tabela... O nosso sistema não funciona!
- ORM: gerir a partir da aplicação
 - Ou, pelo menos, inicialmente

Relembrando JPA

- A partir de entidades
 - Anotação `@Entity` na classe que deve ser entidade
- Chave primária
 - Anotação `@Id`
 - Estratégia de geração automática (se houver)
 - Auto, Identity, Table ou sequence (a depender da política usada pelo BD)
 - Anotação `@GeneratedValue(...)`

Entidades



The screenshot shows an IDE with the Package Explorer on the left and the Source Editor on the right. The Package Explorer displays the project structure, including the 'redesocial' package and its sub-packages. The Source Editor shows the code for the 'User' entity, which is a JPA entity with fields for 'id', 'name', 'email', and 'senha'.

```
1 package com.redesocial.redesocial.modelo;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7
8 @Entity
9 public class User {
10
11     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
12     private Long id;
13     private String name;
14     private String email;
15     private String senha;
16
17
18 }
```



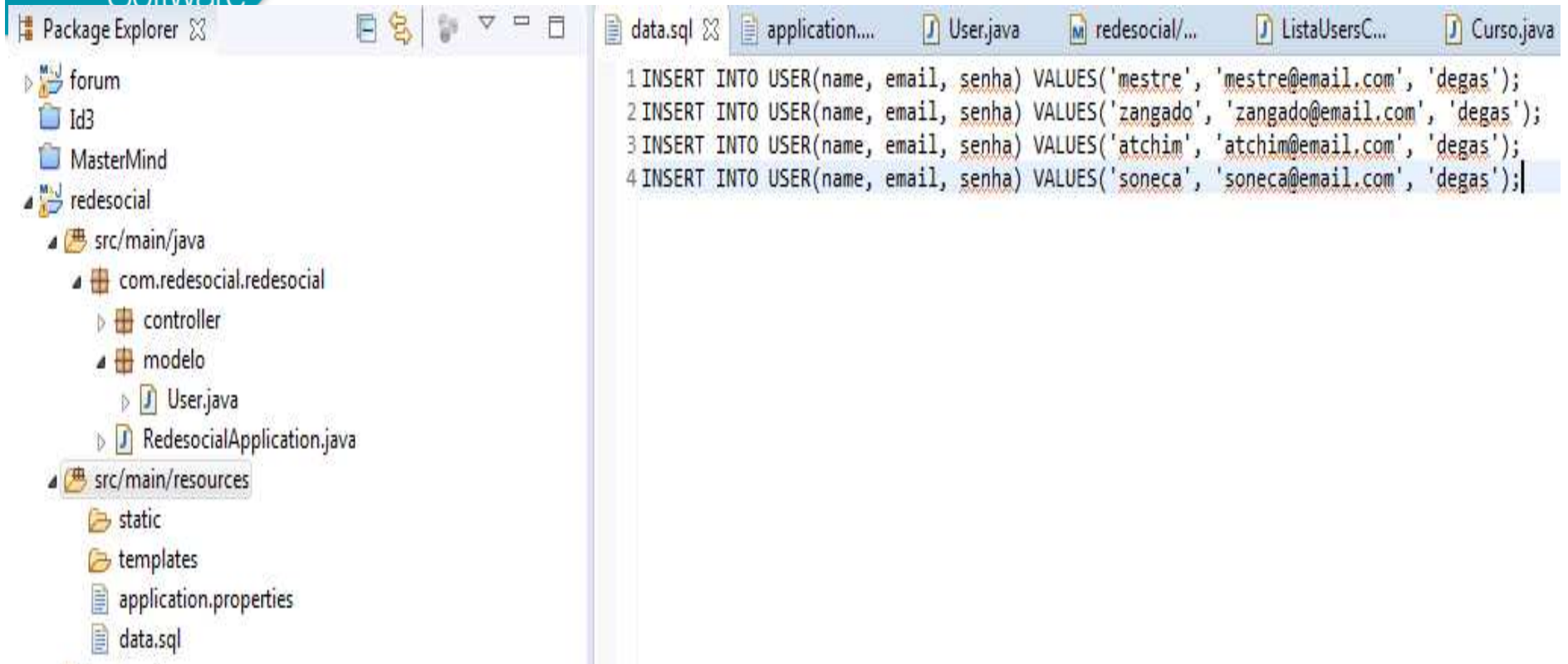

Residência
em Software

Tem um easter egg aqui... Quem
achou?

Tornando o H2 Útil

- H2 é em memória
 - Zera quando reinicia a aplicação
- Forçar inicialização com dados
- Arquivo data.sql (src/main/resources)
 - Comandos SQL para popular o BD
 - Viabilizar testes

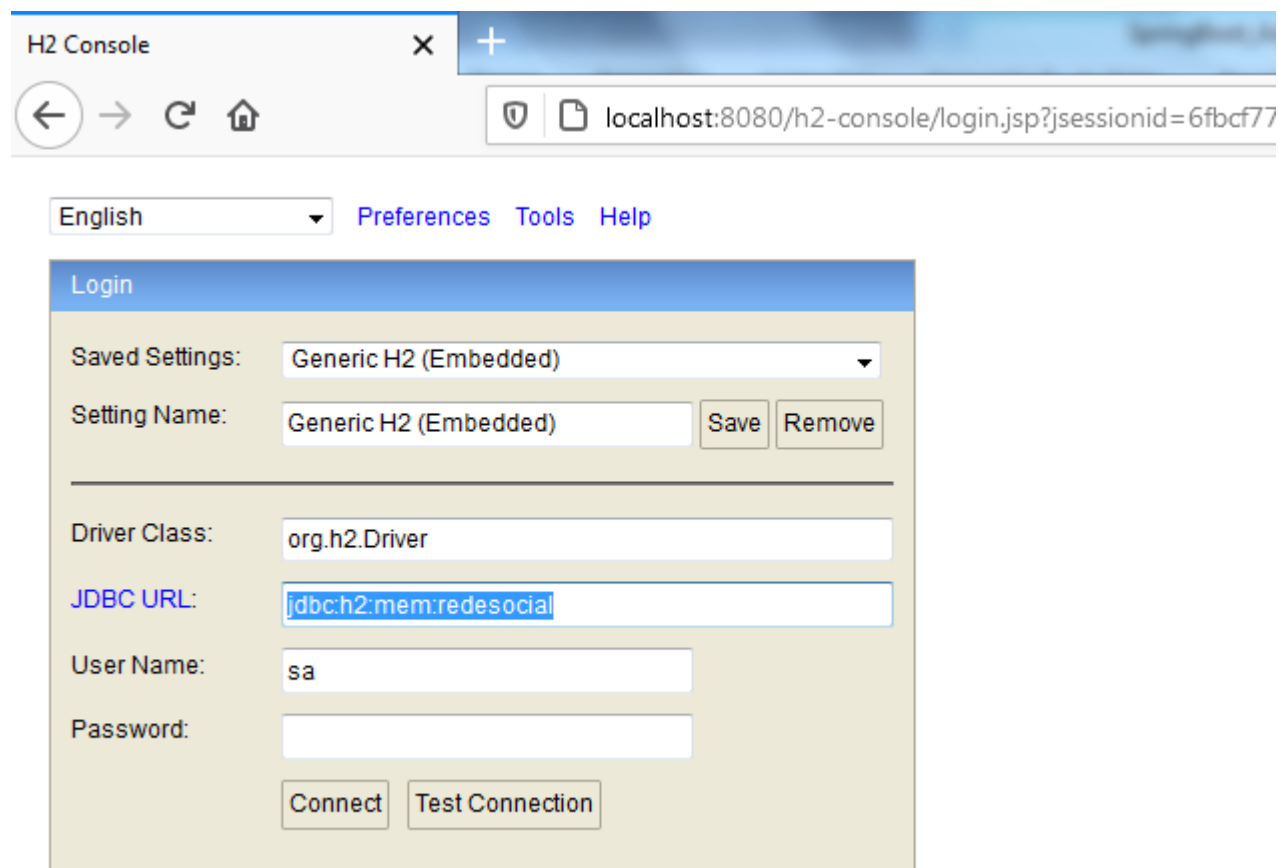
Data.sql



Checando o H2

- Inicie sua aplicação
 - TerCertificar que data.sql foi executado
- Endereço do console do h2
 - Configurado dentro do arquivo application.properties
 - No nosso caso: /h2-console
 - Localhost:8080/h2-console
- JDBS URL (certificar)
 - Configurado dentro do arquivo application.properties
 - jdbc:h2:mem:redesocial

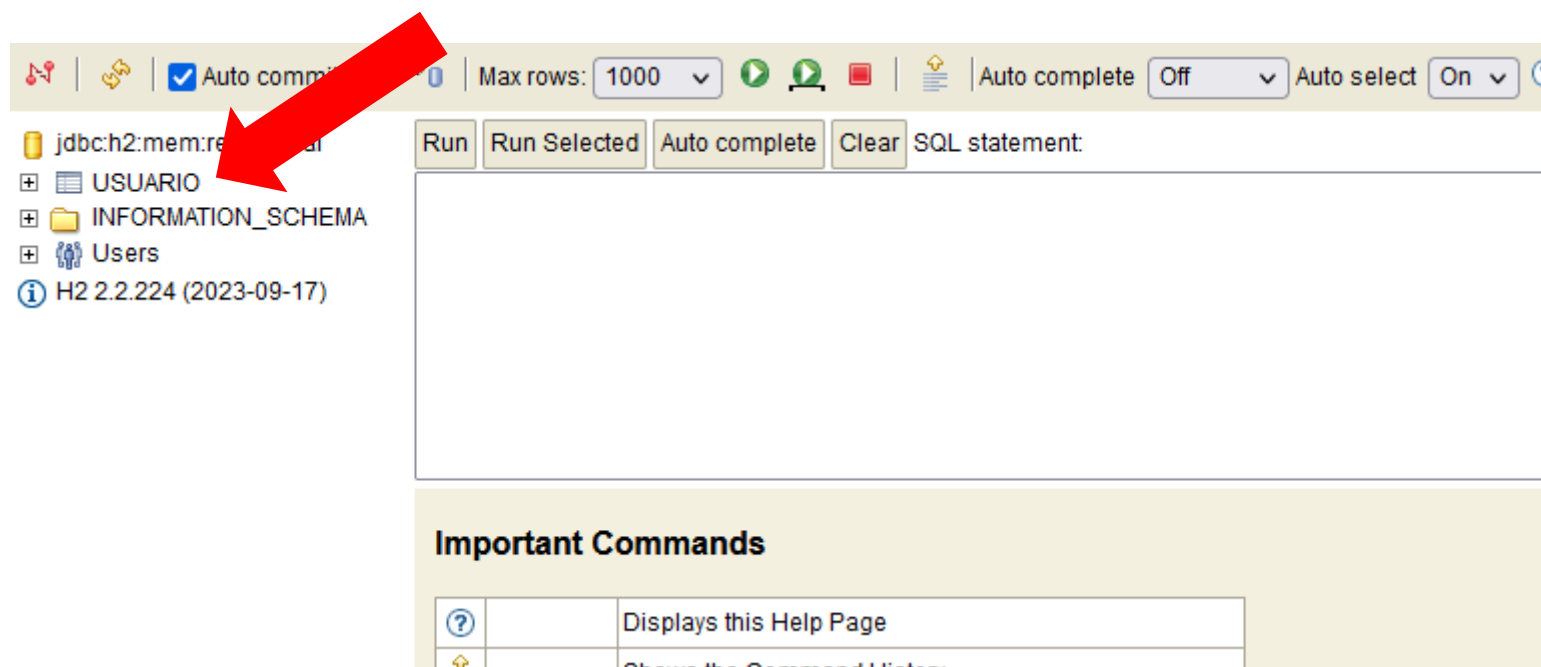
H2 Console



The screenshot shows a web browser window titled "H2 Console" with the address bar displaying "localhost:8080/h2-console/login.jsp?jsessionid=6fbcf77". The page has a navigation bar with "English" (a dropdown menu), "Preferences", "Tools", and "Help". The main content area is titled "Login" and contains the following fields and buttons:

- Saved Settings:** A dropdown menu showing "Generic H2 (Embedded)".
- Setting Name:** A text input field containing "Generic H2 (Embedded)", with "Save" and "Remove" buttons to its right.
- Driver Class:** A text input field containing "org.h2.Driver".
- JDBC URL:** A text input field containing "jdbc:h2:mem:redesocial".
- User Name:** A text input field containing "sa".
- Password:** An empty text input field.
- Buttons:** "Connect" and "Test Connection" buttons at the bottom.

H2 Console



H2 Console

🔗 | 📌 | ☒ Auto commit | 📄 | Max rows: 1000 | ▶️ | 🔄 | 🛑 | 📄 | Auto complete Off

📁 jdbc:h2:mem:redesocial

- 📄 USUARIO
- 📁 INFORMATION_SCHEMA
- 👤 Users
- 📄 H2 2.2.224 (2023-09-17)

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM USUARIO
```

SELECT * FROM USUARIO;

ID	EMAIL	NOME	SENHA
1	mestre@tutu	mestre	mestre
2	zangado@tutu	zangado	zangado
3	atchim@tutu	atchim	atchim
4	soneca@tutu	soneca	soneca

(4 rows, 0 ms)

Edit

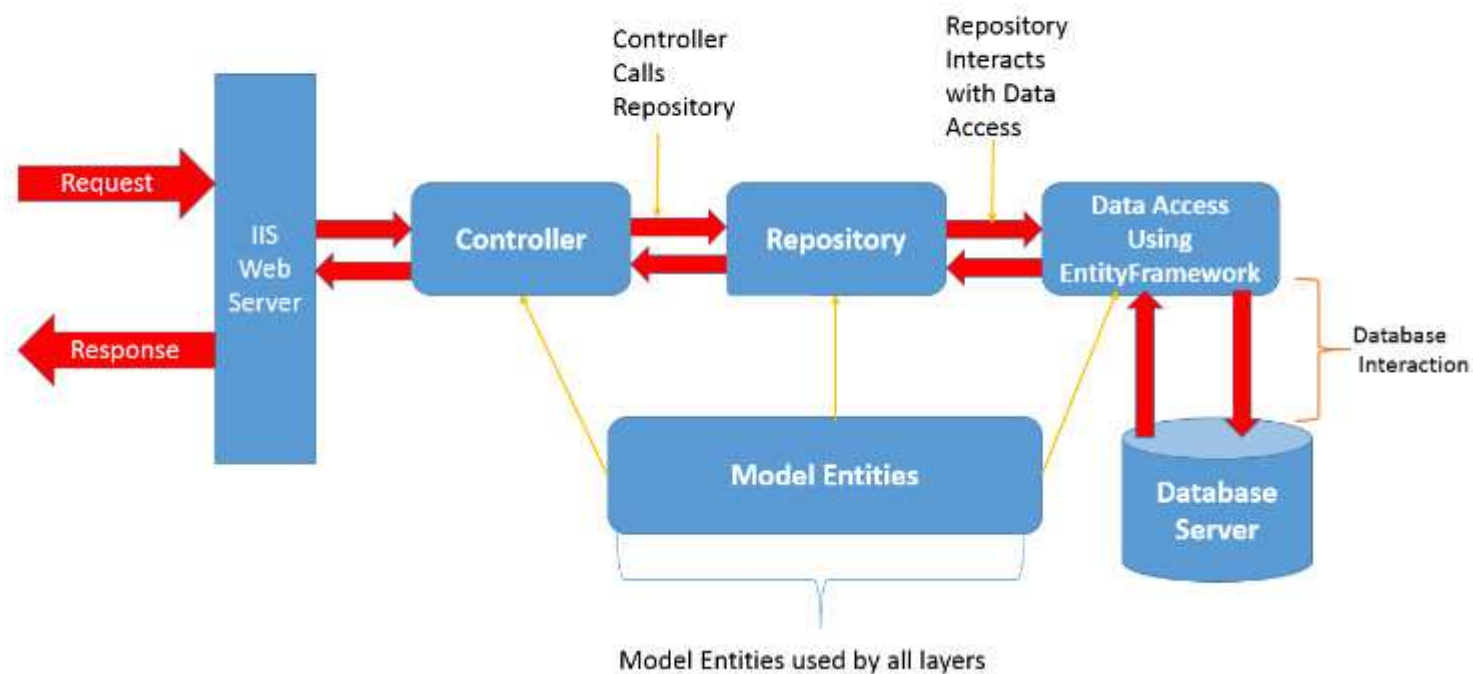
Buscando dados do BD

- ListaUsersController
 - Vamos alterar: não vai mais buscar uma lista estática; buscará dados do BD
- Observações
 - Não é boa prática “espalhar” @Entity pelo código
 - Manter na package Modelo.
 - Padrão DAO – Data Access Object: classes específicas que acessam o BD
 - CRUD: muito parecido (muda só a entidade)

Separando Dado da Ação

- Ação
 - CRUD (tipicamente)
- Dado
 - Qualquer entidade do BD
- Ideia
 - Criar uma interface que permita associar uma ação a uma entidade
- Padrão Repository

Padrão Repository



Padrão Repository

- Uma interface
 - Herda do SpringData múltiplos métodos
 - Criar, apagar, inserir, listar, alterar, etc.
- COMO?
- Crie a package repository
 - “irmã” de controller e modelo
- Crie a interface UsuarioRepository (Um repositório para cada classe de modelo)
 - Extends JpaRepository (Usuario, Integer – tipo da chave)
 - Informar a entidade e o tipo de dados da chave primária
- **IMPORTANTE!**
 - **Toda entidade precisa ter um construtor default**
 - **Sem atributos**

UsuarioRepository

```
redesocial/...  application....  RedesocialA...  HelloControl...  Usuario.java
1 package com.redesocial.redesocial.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5
6
7 public interface UsuarioRepository extends JpaRepository<Usuario, Integer> {
8
9 }
10
```

@Autowired

- Anotação que indica uma injeção de dependência
- A partir desse momento as funcionalidades da classe (ou interface) que foi injetada ficam disponíveis
- Nossa estratégia
 - Usar @Autowired para injetar funcionalidades automáticas no Controller

Como usar

- Injetar UsuarioRepository numa classe (controller)
 - Anotação @Autowired
 - Disponibiliza uma série de métodos
- Listar todos os usuários
 - UserRepository.findAll()
 - findAll() já está pronto
 - select * from <entity>
 - SpringData substitui <Entity> por User

Como usar

- No método `listaUsuarios()` da classe `ListaUsersController`
- Criar um atributo do tipo `UsuarioRepository`
 - `@Autowired`
- No método `listaUsuarios()`
 - Criar um `ArrayList` de `Usuario`
 - Preenchido pelo método `findAll()` do `Repository`
 - Precisa de um `Cast`
 - Converter para o `DTO`
 - `Return`
- Não esquecer do construtor default na entidade!

Entidade Usuario

```
redesocial/... application.... RedesocialA... HelloControl... Usuario.java ×
1 package com.redesocial.redesocial.modelo;
2
3 import jakarta.persistence.Entity;
4 import jakarta.persistence.GeneratedValue;
5 import jakarta.persistence.GenerationType;
6 import jakarta.persistence.Id;
7
8 @Entity
9 public class Usuario {
10
11     @Id
12     @GeneratedValue (strategy = GenerationType.IDENTITY)
13     private Integer Id;
14     private String nome;
15     private String email;
16     private String senha;
17
18     Usuario() {
19
20     }
21
22     public Usuario(int id, String nome, String email, String senha) {
23         super();
24         Id = id;
25         this.nome = nome;
26         this.email = email;
27         this.senha = senha;
28
29     }
30
31
32
33     public Integer getId() {
34         return Id;
```


ListaUsersController

```
desocial/...  application....  RedesocialA...  HelloControl...  Usuario.java  ListaUsersC...  X

package com.redesocial.redesocial.controller;

import java.util.ArrayList;
import java.util.List;

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.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

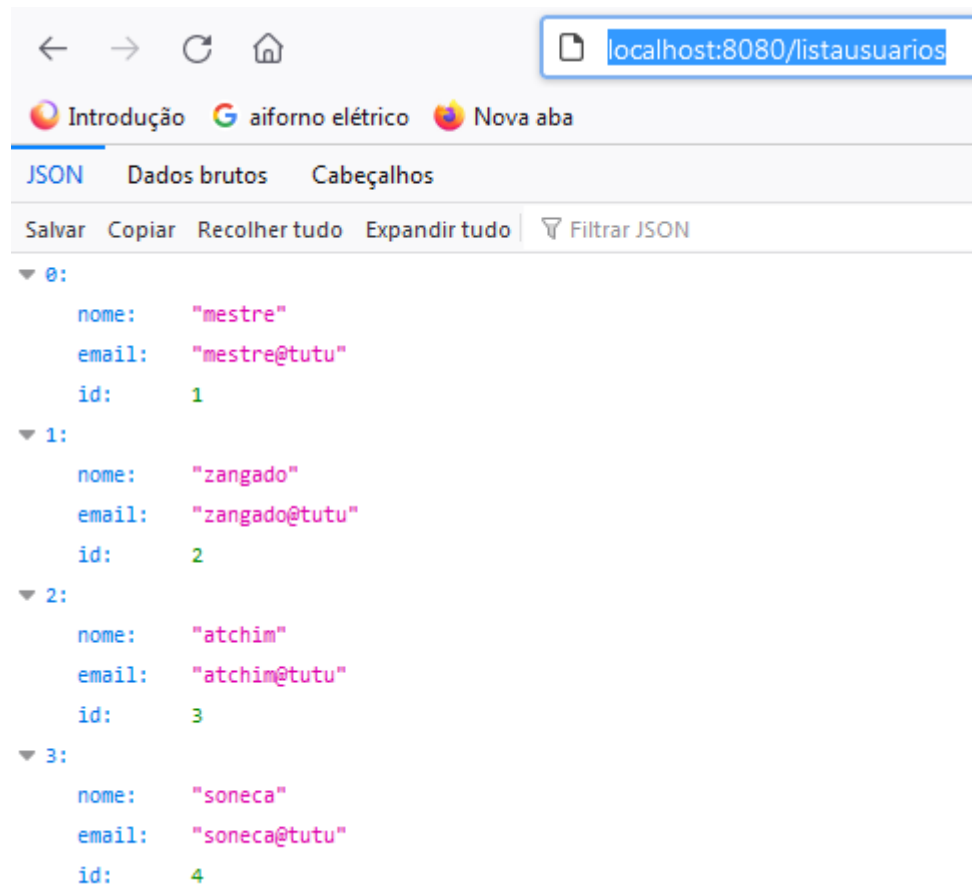
import com.redesocial.redesocial.controller.dto.UserDTO;
import com.redesocial.redesocial.modelo.Usuario;
import com.redesocial.redesocial.repository.UsuarioRepository;

@RestController
public class ListaUsersController {

    @Autowired
    private UsuarioRepository usuarioRepository;

    @RequestMapping("/listausuarios")
    public List<UserDTO> listaUsuarios() {
        List<Usuario> listaUsuarios = (ArrayList<Usuario>) usuarioRepository.findAll();
        List<UserDTO> lista = new ArrayList<UserDTO>();
        for (Usuario u: listaUsuarios) {
            UserDTO ud = new UserDTO(u);
            lista.add(ud);
        }
        return lista;
    }
}
```

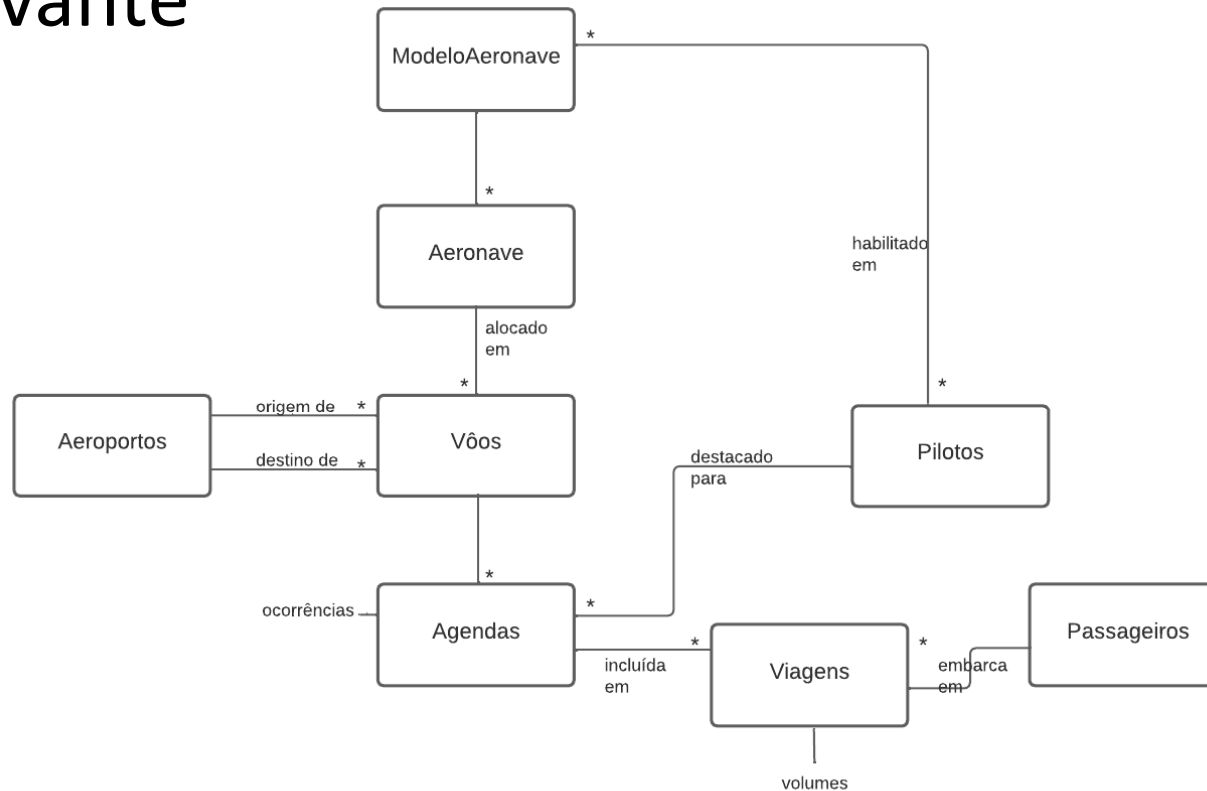
localhost:8080/listausuarios



```
{
  "0": {
    "nome": "mestre",
    "email": "mestre@tutu",
    "id": 1
  },
  "1": {
    "nome": "zangado",
    "email": "zangado@tutu",
    "id": 2
  },
  "2": {
    "nome": "atchim",
    "email": "atchim@tutu",
    "id": 3
  },
  "3": {
    "nome": "soneca",
    "email": "soneca@tutu",
    "id": 4
  }
}
```

Exercício

- O diagrama abaixo vai nos acompanhar doravante



Entidades para Começar

- Piloto(\$Id, Nome, NumBreve)
 - Ex. (01, “Juca Pires”, “12354”)
- ModeloAeronave(\$Id, Fabricante, Nome)
 - Ex. (01, “Embraer”, “EMB195”)
- Aeroporto(\$Id, ICAO, Nome)
 - Ex. (01, “SBIL”, “Ilhéus”)

Seu trabalho

- Crie um novo projeto SpringBoot
- Crie as entidades
 - Com os respectivos repositórios
- Crie um data.sql para popular o BD
- Crie os procedimentos para atender aos requests
 - /listamodeloaeronaves
 - /listapilotos
 - /listaaeroportos