




Desenvolvimento Java Web com Spring Boot

Set 2019

Agenda



- HTTP básico
 - Frameworks Java Web
 - Spring VS Spring Boot
 - Criando uma aplicação Spring Boot + MVC
 - Melhorando a segurança da aplicação
- 

HTTP(S)

- Protocolo utilizado na comunicação entre navegadores e servidores web
- 4 métodos básicos: GET, POST, PUT, DELETE
 - `curl -X GET http://www.google.com`

JavaEE

- Desenvolvimento Java Web em sua forma mais “pura”.
- Estabelece especificações que são utilizados por outros frameworks (banco de dados, mensageria, etc).
- Historicamente era muito complicado e necessitava de muitas configurações, algumas por XML.
- Atualizações “atrasadas” em relação as necessidades do mercado

<https://www.caelum.com.br/apostila-java-web/>

Java Web Frameworks



VERT.X



vaadin}>



e muitos outros...

Sobre o Spring

Conjunto de projetos que buscam resolver problemas comuns aos desenvolvedores (desenvolvimento web, comunicação com banco de dados, segurança de aplicações, etc...)

<https://spring.io>

Spring Framework

- Suporte a injeção de dependências
- Internacionalização
- Validação
- Testes
- Spring MVC
- Spring Data JPA



Spring Boot

- Todos os anteriores
- Aplicações stand-alone (runnable jar)
- Servidor web já embarcado
- Sempre que possível tenta configurar automaticamente o Spring e outras dependências.

Spring Framework

- Suporte a injeção de dependências
- Internacionalização
- Validação
- Testes
- Spring MVC
- Spring Data JPA



Spring Boot

- Todos os anteriores
- Aplicações stand-alone (runnable jar)
- Servidor web já embarcado
- Sempre que possível tenta configurar automaticamente o Spring e outras dependências.
- Sem precisar configurar XML



Criando projeto Spring Boot

<https://start.spring.io/>

Project: Maven Project

Language: Java

Spring Boot: 2.1.7

Project Metadata:

- Group: br.org.sidi.puccamp
- Artifact: minicurso

Dependencies:

- Spring Web Starter

Generate the project - Ctrl + ↵

Baixando o projeto

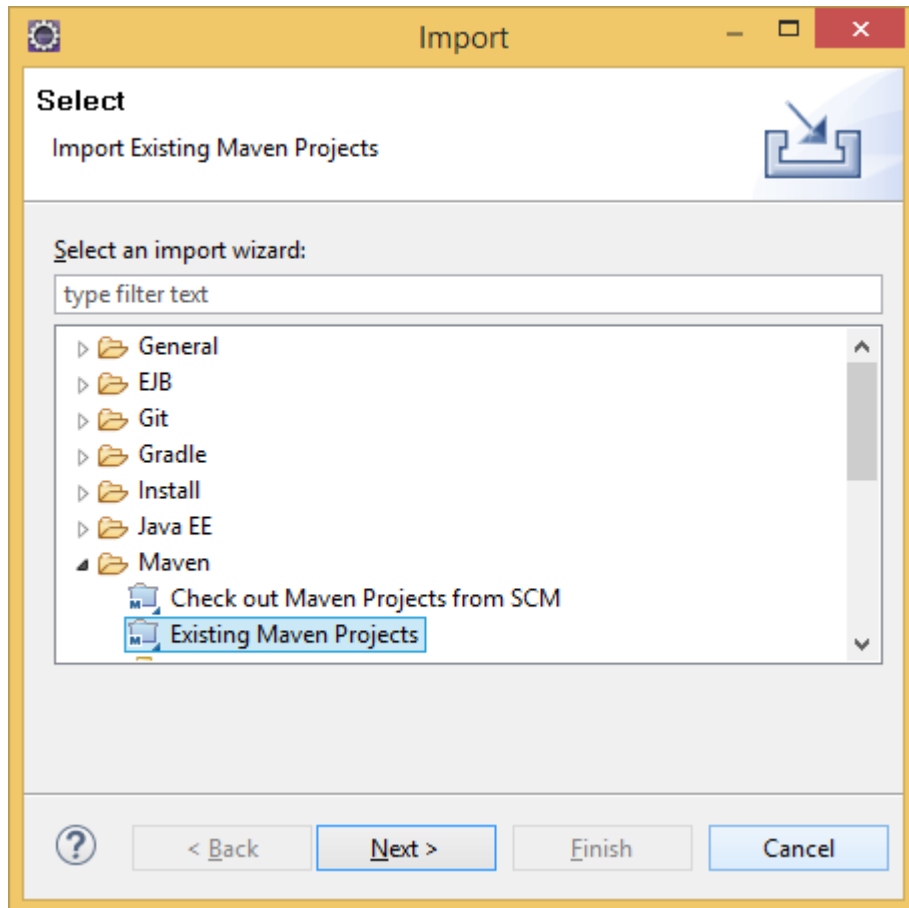
- `git clone https://github.com/rcdvl/minicurso-puccamp.git`

ou

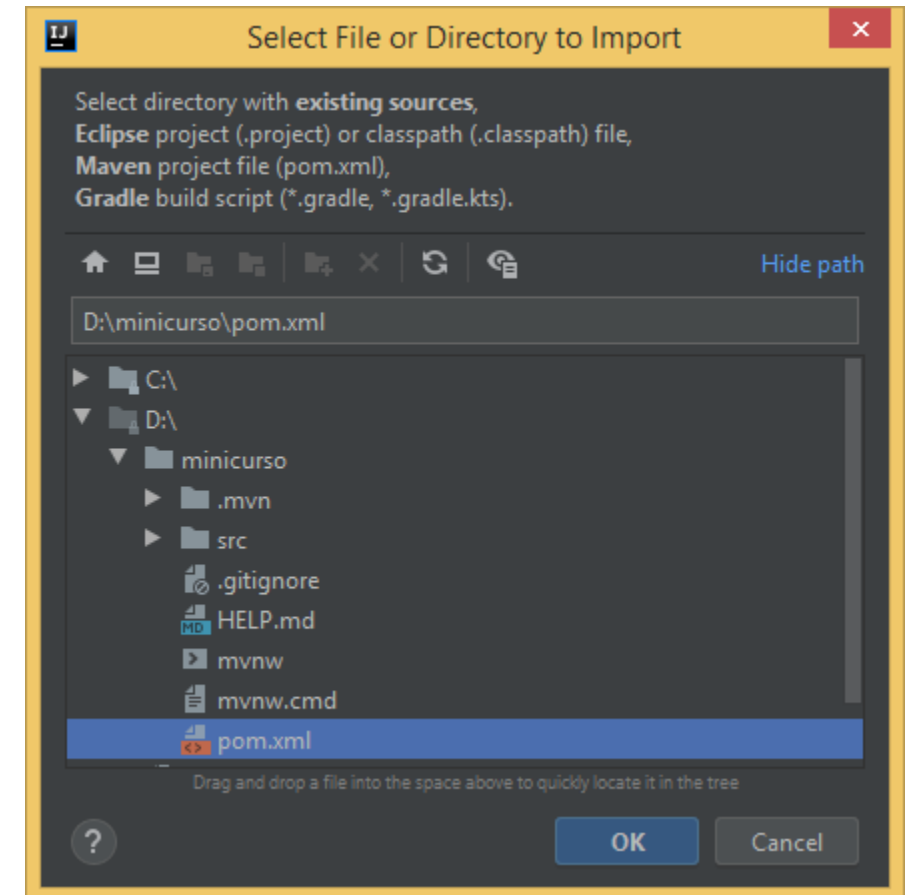
- <https://github.com/rcdvl/minicurso-puccamp/archive/master.zip>

Importando o projeto

Eclipse: File → Import



IntelliJ: File → New → Project from Existing Sources...
ou Import Project



Entendendo a aplicação

```
package br.org.sidi.puccamp.minicurso;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class MinicursoApplication {

    public static void main(String[] args) {
        SpringApplication.run(MinicursoApplication.class, args);
    }

}
```

Adicionando testes ~ TDD

+ src/test/java/br/org/sidi/puccamp/minicurso/StudentsIntegrationTests.java

```
@RunWith(SpringRunner.class)
@SpringBootTest
@AutoConfigureMockMvc
public class StudentsIntegrationTests {

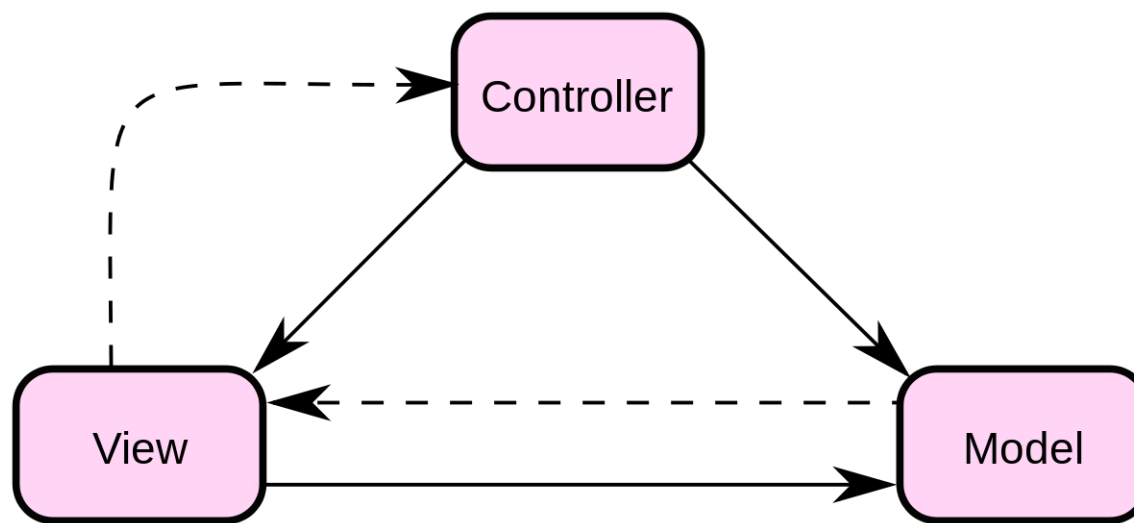
    @Autowired
    private MockMvc mockMvc;

    @Test
    public void shouldAddAndListStudents() throws Exception {
        MockHttpServletRequestBuilder addRequest = post("/students")
            .param("fullName", "Test Student");

        mockMvc.perform(addRequest)
            .andExpect(status().is3xxRedirection())
            .andExpect(redirectedUrl("/students"));

        mockMvc.perform(get("/students"))
            .andExpect(content().string(Matchers.containsString("Test Student")));
    }
}
```

Adicionando MVC a aplicação



Adicionando MVC – @Controller e @RequestMapping

+ src/main/java/br/org/sidi/puccamp/minicurso/controller/WelcomeController.java

```
package br.org.sidi.puccamp.minicurso.controller;

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

@Controller
@RequestMapping("/welcome")
public class WelcomeController {

    @RequestMapping
    public void index() {
    }
}
```


Adicionando MVC – Thymeleaf

 pom.xml

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-devtools</artifactId>
  <optional>true</optional>
</dependency>
```


Adicionando MVC – Thymeleaf

 src/main/java/br/org/sidi/puccamp/minicurso/controller/WelcomeController.java

```
...  
@RequestMapping  
public String index() {  
    return "welcome";  
}  
...
```


+ src/main/resources/templates/welcome.html

```
<!DOCTYPE html>  
<html>  
<head>  
    <title>Spring Boot PUCCAMP</title>  
</head>  
  
<body>  
<p>Welcome</p>  
</body>  
</html>
```

evoluindo a aplicação_

git checkout -f db

Adicionando MVC – suporte JPA e H2

 pom.xml

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
  <groupId>com.h2database</groupId>
  <artifactId>h2</artifactId>
</dependency>
```

 src/main/resources/application.properties

```
spring.datasource.driver-class-name=org.h2.Driver
spring.datasource.url=jdbc:h2:mem:bootapp;DB_CLOSE_DELAY=-1
spring.datasource.username=sa
spring.datasource.password=
```

Adicionando MVC - @Entity, @Id, @GeneratedValue

+ src/main/java/br/org/sidi/puccamp/minicurso/entity/Student.java

```
package br.org.sidi.puccamp.minicurso.entity;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;

@Entity
public class Student {
    @Id
    @GeneratedValue
    private long id;

    @Column(nullable = false, unique = true)
    private String fullName;

    public Student(String fullName) {
        this.fullName = fullName;
    }
}
```

Adicionando MVC - JpaRepository

+ src/main/java/br/org/sidi/puccamp/minicurso/repository/StudentRepository.java

```
package br.org.sidi.puccamp.minicurso.repository;

import br.org.sidi.puccamp.minicurso.entity.Student;
import org.springframework.data.jpa.repository.JpaRepository;

public interface StudentRepository extends JpaRepository<Student, Long> {
}
```

Adicionando MVC – Controller e View para visualizar dados

+ src/main/java/br/org/sidi/puccamp/minicurso/controller/StudentsController.java

```
@Controller
@RequestMapping("/students")
public class StudentsController {

    @Autowired
    private StudentRepository studentsRepository;

    @RequestMapping
    public ModelAndView index() {
        ModelAndView modelAndView = new ModelAndView("students/index");
        modelAndView.addObject("students", studentsRepository.findAll());
        return modelAndView;
    }
}
```

Adicionando MVC – Controller e View para visualizar dados

+ src/main/resources/templates/students/index.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
    <title>Spring Boot PUCCAMP</title>
</head>

<body>
    <p>Lista de alunos:</p>
    <ul>
        <li th:each="student : ${students}">
            <p th:text="${student.fullName}"></p>
        </li>
    </ul>
</body>
</html>
```

Adicionando MVC – Acessando console H2

<http://localhost:8080/h2-console/>

English ▼ Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded) ▼

Setting Name: Generic H2 (Embedded) Save Remove

Driver Class: org.h2.Driver

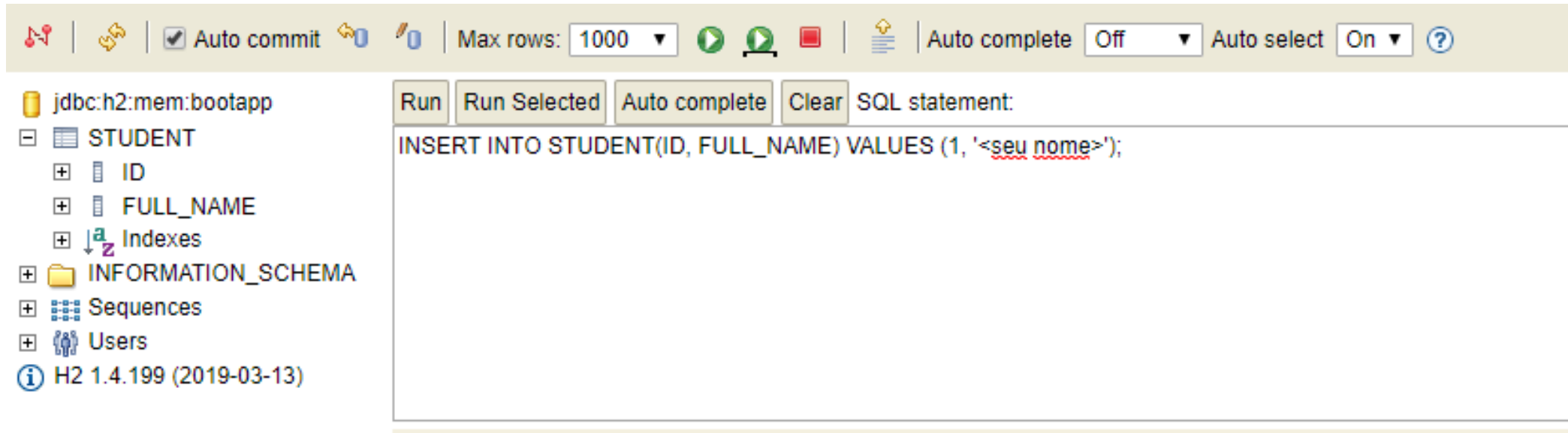
JDBC URL: jdbc:h2:mem:bootapp

User Name: sa

Password:


Connect Test Connection

Adicionando MVC – Inserindo dados via console



The screenshot displays the H2 Database Console interface. The top toolbar includes icons for connection, settings, and execution, along with controls for 'Auto commit' (checked), 'Max rows' (set to 1000), 'Auto complete' (set to Off), and 'Auto select' (set to On). The left sidebar shows the database structure for 'jdbc:h2:mem:bootapp', including a 'STUDENT' table with columns 'ID' and 'FULL_NAME', and other database objects like 'INDEXES', 'INFORMATION_SCHEMA', 'SEQUENCES', and 'USERS'. The main area, titled 'SQL statement:', contains the text: `INSERT INTO STUDENT(ID, FULL_NAME) VALUES (1, '<seu nome>');`. The text '<seu nome>' is underlined in red, indicating it is a placeholder for a user input.

Adicionando MVC - JpaRepository

 src/main/java/br/org/sidi/puccamp/minicurso/repository/StudentRepository.java


```
package br.org.sidi.puccamp.minicurso.repository;

import br.org.sidi.puccamp.minicurso.entity.Student;
import org.springframework.data.jpa.repository.JpaRepository;

import java.util.List;

public interface StudentRepository extends JpaRepository<Student, Long> {
    public List<Student> findOrderByFullNameAsc();
}
```

Adicionando MVC – Controller e View para visualizar dados

 src/main/java/br/org/sidi/puccamp/minicurso/controller/StudentsController.java

```
@Controller
@RequestMapping("/students")
public class StudentsController {


    @Autowired
    private final StudentRepository studentsRepository;

    @RequestMapping
    public ModelAndView index() {
        ModelAndView modelAndView = new ModelAndView("students/index");
        modelAndView.addObject("students",
studentsRepository.findByOrderByFullNameAsc());
        return modelAndView;
    }
}
```

formulários_

git checkout -f form

Adicionando MVC – Controller e View para visualizar dados

 src/main/java/br/org/sidi/puccamp/minicurso/controller/StudentsController.java

```
...

@RequestMapping
public ModelAndView index() {
    ModelAndView modelAndView = new ModelAndView("students/index");
    modelAndView.addObject("students", studentsRepository.findAll());
    modelAndView.addObject(new Student());
    return modelAndView;
}

@PostMapping
public String add(Student student) {
    this.studentsRepository.save(student);
    return "redirect:/students";
}
}
```

Adicionando MVC – Controller e View para visualizar dados

 src/main/resources/templates/students/index.html

```
...
<body>
  <form method="post" th:object="${student}" th:action="@{/students}">
    <input type="text" placeholder="Nome" th:field="*{fullName}"/>
    <button type="submit">Adicionar</button>
  </form>

  <p>Lista de alunos:</p>
  <ul>
    <li th:each="student : ${students}">
      <p th:text="${student.fullName}"></p>
    </li>
  </ul>
</body>
</html>
```



relacionamentos_


git checkout -f orm

Adicionando MVC - @ManyToMany

 src/main/java/br/org/sidi/puccamp/minicurso/entity/Student.java

```
public class Student {  
  
    ...  
  
    @ManyToMany  
    @JoinTable(name = "friends",  
        joinColumns = @JoinColumn(name = "studentId"),  
        inverseJoinColumns = @JoinColumn(name = "friendId"))  
    private Set<Student> friends;  
    @ManyToMany  
    @JoinTable(name = "friends",  
        joinColumns = @JoinColumn(name = "friendId"),  
        inverseJoinColumns = @JoinColumn(name = "studentId"))  
    private Set<Student> friendOf;  
    ...  
}
```


Adicionando MVC - JpaRepository

 src/main/java/br/org/sidi/puccamp/minicurso/repository/StudentRepository.java


```
package br.org.sidi.puccamp.minicurso.repository;

import br.org.sidi.puccamp.minicurso.entity.Student;
import org.springframework.data.jpa.repository.JpaRepository;

import java.util.List;

public interface StudentRepository extends JpaRepository<Student, Long> {
    List<Student> findByOrderByFullNameAsc();
    Student findByFullName(String fullName);
}
```

Adicionando MVC – Controller e View para visualizar dados


 src/main/java/br/org/sidi/puccamp/minicurso/controller/StudentsController.java

```
@Controller
@RequestMapping("/students")
public class StudentsController {

    ...

    @GetMapping("/{id}")
    public ModelAndView show(@PathVariable("id") Long id) {
        ModelAndView modelAndView = new ModelAndView("students/show");
        modelAndView.addObject("student",
studentsRepository.findById(id).get());
        return modelAndView;
    }
}
```

Adicionando MVC – Controller e View para visualizar dados

 src/main/java/br/org/sidi/puccamp/minicurso/controller/StudentsController.java

```
@Controller
@RequestMapping("/students")
public class StudentsController {

    ...

    @PostMapping("/{id}/addFriend")
    public String addFriend(@PathVariable("id") Long id,
        @RequestParam("fullName") String friendName) {
        Student user = studentsRepository.findById(id).get();
        Student friend = studentsRepository.findByFullName(friendName);
        user.getFriends().add(friend);
        studentsRepository.save(user);
        return "redirect:/students/" + id;
    }
}
```

Adicionando MVC – Controller e View para visualizar dados

+ src/main/resources/templates/students/show.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
  <title>Spring Boot PUCCAMP</title>
</head>
<body>
  <form method="post" th:object="${student}" th:action="@{/students/{id}/addFriend(id=${student.id})}">
    <input type="text" placeholder="Nome" id="fullName" name="fullName"/>
    <button type="submit">Adicionar amigo</button>
  </form>

  <p>Lista de amigos:</p>
  <ul>
    <li th:each="friend : ${student.friends}">
      <p th:text="${friend.fullName}"></p>
    </li>
  </ul>
</body>
</html>
```

segurança_
git checkout -f security

Autenticação de usuários

 pom.xml

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-security</artifactId>
</dependency>

<dependency>
  <groupId>org.thymeleaf.extras</groupId>
  <artifactId>thymeleaf-extras-springsecurity5</artifactId>
  <version>3.0.4.RELEASE</version>
</dependency>
```

Autenticação de usuários – configuração

+ src/main/java/br/org/sidi/puccamp/minicurso/WebSecurityConfig.java

```
@Configuration
@EnableWebSecurity
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http
            .authorizeRequests()
                .antMatchers("/", "/welcome").permitAll()
                .antMatchers(HttpMethod.GET, "/students").permitAll()
                .antMatchers(HttpMethod.POST, "/students").hasRole("ADMIN")
                .anyRequest().authenticated()
            .and()
                .formLogin()
                .loginPage("/login")
                .permitAll()
                .defaultSuccessUrl("/students")
            .and()
                .logout()
                .logoutSuccessUrl("/students")
                .permitAll();
    }
}
```

Autenticação de usuários – configuração

+ src/main/java/br/org/sidi/puccamp/minicurso/WebSecurityConfig.java

```
@Bean
@Override
public UserDetailsService userDetailsService() {
    UserDetails user =
        User.withDefaultPasswordEncoder()
            .username("user")
            .password("password")
            .roles("USER")
            .build();

    UserDetails admin =
        User.withDefaultPasswordEncoder()
            .username("admin")
            .password("password")
            .roles("ADMIN")
            .build();

    return new InMemoryUserDetailsManager(user, admin);
}
```


Autenticação de usuários – controller

+ src/main/java/br/org/sidi/puccamp/minicurso/controller/SecurityController.java

```
package br.org.sidi.puccamp.minicurso.controller;

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

@Controller
public class SecurityController {

    @RequestMapping("/login")
    public String login() {
        return "login";
    }
}
```

Autenticação de usuários – view

+ src/main/resources/templates/login.html

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="https://www.thymeleaf.org">
<head>
    <title>Spring Security Example </title>
</head>
<body>
    <div th:if="${param.error}">
        Invalid username and password.
    </div>
    <div th:if="${param.logout}">
        You have been logged out.
    </div>
    <form th:action="@{/login}" method="post">
        <div><label> User Name : <input type="text" name="username"/> </label></div>
        <div><label> Password: <input type="password" name="password"/> </label></div>
        <div><input type="submit" value="Sign In"/></div>
    </form>
</body>
</html>
```

Autenticação de usuários – view

 src/main/resources/templates/students/index.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
      xmlns:sec="http://www.thymeleaf.org/extras/spring-security">
...
<body>
  <div sec:authorize="isAuthenticated()">
    <p th:text="|Olá, ${#authentication.name}|"></p>

    <form th:action="@{/logout}" method="post">
      <input type="submit" value="Sign Out"/>
    </form>
  </div>
  <div sec:authorize="isAnonymous()">
    <a th:href="@{/login}">Login</a>
  </div>

  <div sec:authorize="hasRole('ROLE_ADMIN')">
    <form method="post" th:object="${student}" th:action="@{/students}">
      <input type="text" placeholder="Nome" th:field="*{fullName}"/>
      <button type="submit">Adicionar</button>
    </form>
  </div>
...
```




Correção do teste

 pom.xml

```
<dependency>
  <groupId>org.springframework.security</groupId>
  <artifactId>spring-security-test</artifactId>
  <scope>test</scope>
</dependency>
```

Desafio: correção do teste

 src/test/java/br/org/sidi/puccamp/minicurso/StudentsIntegrationTests.java

```
@Test
public void shouldAddAndListStudents() throws Exception {
    MockHttpServletRequestBuilder addRequest = post("/students")
        .param("fullName", "Test Student")
        .with(csrf());
    MockHttpServletRequestBuilder loginRequest = post("/login")
        .param("username", "admin")
        .param("password", "password")
        .with(csrf());

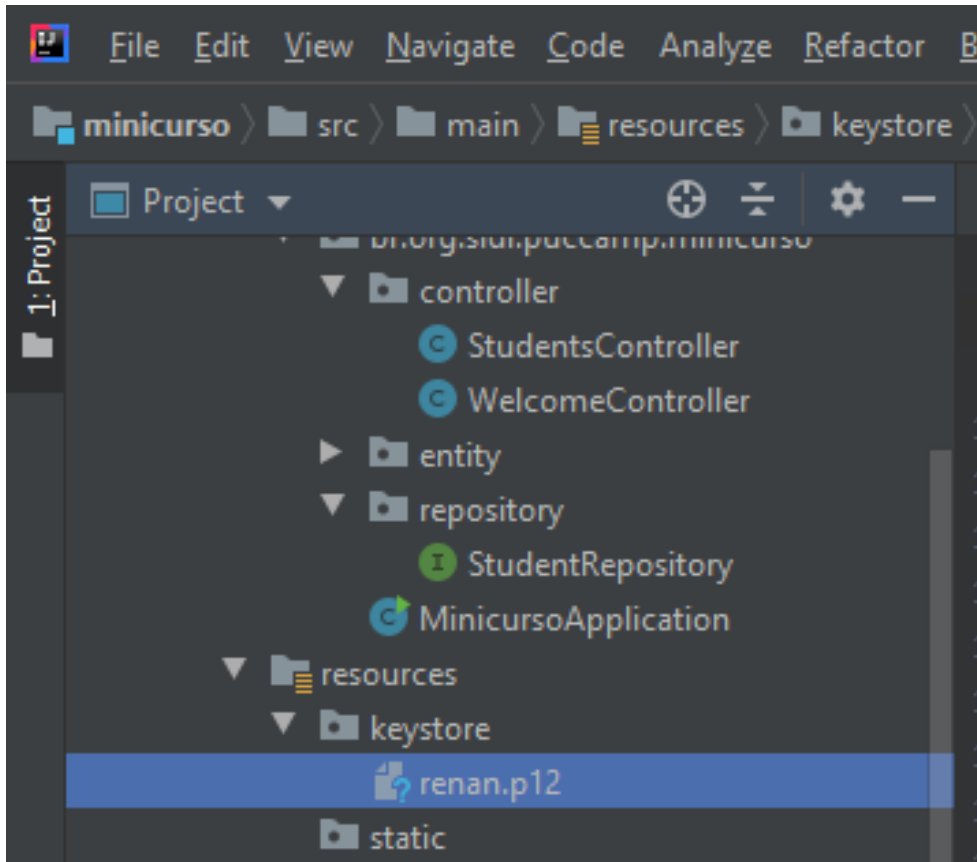
    mockMvc.perform(loginRequest)
        .andExpect(status().is3xxRedirection())
        .andExpect(redirectedUrl("/students"));
}
```

...




Segurança TLS

Mover p12 keystore para src/main/resources/keystore



Segurança TLS

 src/main/resources/application.properties

```
server.port=8443

spring.datasource.driver-class-name=org.h2.Driver
spring.datasource.url=jdbc:h2:mem:bootapp;DB_CLOSE_DELAY=-1
spring.datasource.username=sa
spring.datasource.password=

server.ssl.key-store-type=PKCS12
server.ssl.key-store=classpath:keystore/minicurso.p12
server.ssl.key-store-password=minicurso
server.ssl.key-alias=minicurso
```

UI bonus_

git checkout -f ui

Material UI – adicionando dependencias

 pom.xml

```
<dependency>
  <groupId>org.webjars</groupId>
  <artifactId>materializecss</artifactId>
  <version>1.0.0</version>
</dependency>

<dependency>
  <groupId>nz.net.ultraq.thymeleaf</groupId>
  <artifactId>thymeleaf-layout-dialect</artifactId>
  <version>2.4.1</version>
</dependency>
```

Material UI

+ src/main/java/br/org/sidi/puccamp/minicurso/AppConfig.java

```
package br.org.sidi.puccamp.minicurso;
import nz.net.ultraq.thymeleaf.LayoutDialect;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
@EnableWebMvc
public class AppConfig implements WebMvcConfigurer {
    @Bean
    public LayoutDialect layoutDialect() {
        return new LayoutDialect();
    }

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {
        registry
            .addResourceHandler("/webjars/**")
            .addResourceLocations("/webjars/");
    }
}
```

Autenticação de usuários – configuração

+ src/main/java/br/org/sidi/puccamp/minicurso/WebSecurityConfig.java

```
@Configuration
@EnableWebSecurity
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http
            .authorizeRequests()
                .antMatchers("/", "/welcome", "/webjars/**").permitAll()
                .antMatchers(HttpMethod.GET, "/students").permitAll()
                .antMatchers(HttpMethod.POST, "/students").hasRole("ADMIN")
                .anyRequest().authenticated()
            .and()
                .formLogin()
                .loginPage("/login")
                .permitAll()
                .defaultSuccessUrl("/students")
            .and()
                .logout()
                .logoutSuccessUrl("/students")
                .permitAll();
    }
}
```

Material UI – layout

+ src/main/resources/templates/layouts/layout.html

```
<!DOCTYPE html>
<html lang="en"
  xmlns:th="https://www.thymeleaf.org"
  xmlns:sec="http://www.thymeleaf.org/extras/spring-security"
  xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">
<head>
  <title layout:title-pattern="$LAYOUT_TITLE - $CONTENT_TITLE">Spring Boot PUCCAMP</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>

  <link th:href="@{/webjars/materializecss/1.0.0/css/materialize.min.css}" rel="stylesheet" media="screen"/>
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
</head>
<body>
  <nav class="purple darken-4">
    <div class="nav-wrapper container">
      <a href="#" class="brand-logo">Spring Boot</a>
      <a href="#" data-target="mobile-menu" class="sidenav-trigger"><i class="material-icons">menu</i></a>
```

Material UI – layout

+ src/main/resources/templates/layouts/layout.html


```
<ul id="nav-mobile" class="right hide-on-med-and-down">
  <li><a th:href="@{/welcome}">Welcome</a></li>
  <li><a th:href="@{/students}">Students</a></li>
  <li sec:authorize="isAnonymous()">
    <a th:href="@{/login}">Login</a>
  </li>
  <li sec:authorize="isAuthenticated()">
    <a class="dropdown-trigger" href="#" data-target="user-dropdown">
      <span th:text="${#authentication.name}"></span>
      <i class="material-icons right">arrow_drop_down</i>
    </a>
  </li>
</ul>
</div>
</nav>
<ul class="sidenav" id="mobile-menu">
  <li><a th:href="@{/welcome}">Welcome</a></li>
  <li><a th:href="@{/students}">Students</a></li>
  <li sec:authorize="isAnonymous()">
    <a th:href="@{/login}">Login</a>
  </li>
  <li><a onclick="document.getElementById('sign-out-form').submit();">Sair</a></li>
</ul>
```

Material UI – layout

+ src/main/resources/templates/layouts/layout.html


```
<form id="sign-out-form" th:action="@{/logout}" method="post" style="display: none;">
  <a onclick="document.getElementById('sign-out-form').submit();">Sair</a>
</form>
<ul class="dropdown-content" id="user-dropdown">
  <li>
    <a onclick="document.getElementById('sign-out-form').submit();">Sair</a>
  </li>
</ul>
<div class="container section">
  <div layout:fragment="content"></div>
</div>
<script th:src="@{/webjars/materializecss/1.0.0/js/materialize.min.js}"></script>
<script>
document.addEventListener('DOMContentLoaded', function() {
  var elems = document.querySelectorAll('.sidenav');
  var instances = M.Sidenav.init(elems, null);
});
document.addEventListener('DOMContentLoaded', function() {
  var elems = document.querySelectorAll('.dropdown-trigger');
  var instances = M.Dropdown.init(elems, null);
});
</script>
</body>
```

Material UI – login

 src/main/resources/templates/login.html

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:th="https://www.thymeleaf.org"
      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout" layout:decorate="~{layouts/layout}" >
<head>
  <title>Spring Security Example </title>
</head>
<body>
  <div layout:fragment="content" class="row">
    <div class="col s10 offset-s1 m6 offset-m3 l4 offset-l4 card">
      <div class="card-content">
        <span class="card-title">Login</span>
        <div th:if="${param.error}" style="color: red">
          Invalid username and password.
        </div>
        <div th:if="${param.logout}">
          You have been logged out.
        </div>
      </div>
    </div>
  </div>
```

Material UI – login

 src/main/resources/templates/login.html

```
<form th:action="@{/login}" method="post">
  <div class="input-field">
    <input type="text" id="username" name="username"/><label for="username">User Name</label>
  </div>
  <div class="input-field"><input type="password" id="password" name="password"/>
    <label for="password">Password</label>
  </div>
  <div><button type="submit" class="waves-effect waves-light btn">Sign In</button></div>
</form>
</div>
</div>
</div>
</body>
</html>
```


Material UI – alunos

 src/main/resources/templates/students/index.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
      xmlns:sec="http://www.thymeleaf.org/extras/spring-security"
      xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout" layout:decorate="~{layouts/layout}">
<head>
  <title>Students</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
</head>
<body>
  <div layout:fragment="content">
    <div sec:authorize="hasRole('ROLE_ADMIN')" class="card">
      <div class="card-content row">
        <form method="post" th:object="${student}" th:action="@{/students}">
          <div class="input-field col s10">
            <input type="text" th:field="*{fullName}"/><label for="fullName">Nome</label>
          </div>
          <div class="input-field col s2"><button type="submit" class="btn">Adicionar</button></div>
        </form>
      </div>
    </div>
  </div>
</body>
</html>
```


Material UI – alunos

 src/main/resources/templates/students/index.html

```
<h4>Lista de alunos</h4>
<table>
  <thead>
    <tr><th>Nome</th></tr>
  </thead>
  <tbody>
    <tr th:each="student : ${students}">
      <td th:text="${student.fullName}"></td>
    </tr>
  </tbody>
</table>
</div>
</body>
</html>
```

E agora?



- Tratamento de erros
 - “Remember me” no login
 - Autorização para adicionar amigos
 - Validação de campos – Hibernate Validator
- 

Obrigado,

   @segueosidi



Renan

r.cadaval@sidi.org.br

Felipe

fl.tortella@sidi.org.br