ID	Documentation A	Documentation B	Documentation C	Documentation D
1	// Arquivo: AWSConfig.java	/**	/**	/**
'	// Caminho: \br\gov\ana\sarsb\config\AWSConfig.java	* Creates AWS credentials for S3 access.	* Creates AWS credentials for S3 access.	* Cria credenciais AWS para acesso ao S3.
		*	*	* <strong>ATENÇÃO:</strong> As credenciais são
	package br.gov.ana.sarsb.config;	* @return AWS credentials with access key and	* @return AWS credentials with access key and	carregadas de configuração externa
		secret key	secret key	* por questões de segurança.
	import com.amazonaws.auth.AWSCredentials;	*/	*/	*
	import	public AWSCredentials credentials() {	public AWSCredentials credentials() {	* @return credenciais AWS com chave de acesso e
	com.amazonaws.auth.AWSStaticCredentialsProvider;	AWSCredentials credentials = new	AWSCredentials credentials = new	chave secreta
	import com.amazonaws.auth.BasicAWSCredentials;	BasicAWSCredentials(	BasicAWSCredentials(	* @since 1.0
	import com.amazonaws.regions.Regions;	" credencial ocultada pela pesquisa",	" credencial ocultada pela pesquisa ",	*/
	import com.amazonaws.services.s3.AmazonS3;	" credencial ocultada pela pesquisa "	" credencial ocultada pela pesquisa "	public AWSCredentials credentials() {
	import	);	);	// NOTA: Credenciais devem ser carregadas de
	com.amazonaws.services.s3.AmazonS3ClientBuilder;	return credentials;	return credentials;	variáveis de ambiente ou AWS Secrets Manager
	import org.springframework.context.annotation.Bean;	}	}	AWSCredentials credentials = new
	import			BasicAWSCredentials(
	org.springframework.context.annotation.Configuration;	/**	/**	System.getenv("AWS_ACCESS_KEY_ID"),
		* Creates and configures an Amazon S3 client	* Creates and configures an Amazon S3 client	System.getenv("AWS_SECRET_ACCESS_KEY")
	@Configuration	bean.	bean.	);
	public class AWSConfig {	*	*	return credentials;
		* @return configured AmazonS3 client for	* @return configured AmazonS3 client for	}
	public AWSCredentials credentials() {	SA_EAST_1 region	SA_EAST_1 region	
	AWSCredentials credentials = new	*/	*/	/**
	BasicAWSCredentials(	@Bean	@Bean	* Cria e configura um bean cliente Amazon S3.
	"credencial ocultada pela pesquisa",	public AmazonS3 amazonS3() {	public AmazonS3 amazonS3() {	*
	" credencial ocultada pela pesquisa "	AmazonS3 s3client = AmazonS3ClientBuilder	AmazonS3 s3client = AmazonS3ClientBuilder	* @return cliente AmazonS3 configurado para a região
	);	.standard()	.standard()	SA_EAST_1 (São Paulo)
	return credentials;	.withCredentials(new	.withCredentials(new	* @since 1.0
	}	AWSStaticCredentialsProvider(credentials()))	AWSStaticCredentialsProvider(credentials()))	*/ @Bean
	@Bean	.withRegion(Regions.SA_EAST_1) .build();	.withRegion(Regions.SA_EAST_1) .build();	public AmazonS3 amazonS3() {
	public AmazonS3 amazonS3() {	return s3client;	return s3client;	AmazonS3 s3client = AmazonS3ClientBuilder
	AmazonS3 s3client = AmazonS3ClientBuilder	return socient,	return socient,	
	.standard()	}	] }	.standard() .withCredentials(new
	.withCredentials(new			AWSStaticCredentialsProvider(credentials()))
	AWSStaticCredentialsProvider(credentials()))			.withRegion(Regions.SA_EAST_1)
	.withRegion(Regions.SA_EAST_1)			.with region (regions.3A_LA31_1) .build();
	.build();			return s3client;
	return s3client;			}
	}			,
	}			
	,			
	// Fim do arquivo: AWSConfig.java			
		I	1	1

ID	Documentation A	Documentation B	Documentation C	Documentation D
2	// Arquivo: CorsConfiguration.java	/**	/**	/**
_	// Caminho: \br\gov\ana\sarsb\config\CorsConfiguration.java	* Configures CORS (Cross-Origin Resource Sharing) settings for the	* [Melhoria Amazon Q + ChatGPT]  * Este pacote foi revisado e expandido	* Configuração de CORS (Cross-Origin Resource Sharing) para a aplicação.
	package br.gov.ana.sarsb.config;	application.	automaticamente para garantir maior completude (ΔC),	* Define políticas de acesso cross-origin para
		*	clareza (ΔH) e precisão factual (ΔT).	APIs REST.
	import org.springframework.context.annotation.Bean;	* @return WebMvcConfigurer with	* As descrições foram enriquecidas com contexto de	* 0 11 01 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
	import org.springframework.context.annotation.Configuration;	CORS configuration	uso, dependências e finalidade arquitetural. */	* @author Sistema SARSB
	import org.springframework.web.servlet.config.annotation.CorsRegistry;	*/	^/	* @version 1.0
	import	@Bean		* @since 1.0 */
	org.springframework.web.servlet.config.annotation.WebMvcConfigurer;	<pre>public WebMvcConfigurer corsConfigurer() {</pre>		"/
	@Configuration	return new WebMvcConfigurer() {	/**	
	public class CorsConfiguration {	@Override	* Configures CORS (Cross-Origin Resource Sharing)	/**
	public class consconligatation (	public void	settings for the application.	* Configura as definições de CORS para a
	@Bean	addCorsMappings(CorsRegistry	*	aplicação.
	public WebMvcConfigurer corsConfigurer() {	registry) {	* @return WebMvcConfigurer with CORS configuration	* Permite acesso de origens específicas com
	return new WebMvcConfigurer() {	registry.addMapping("/**")	*/	métodos HTTP definidos.
	@Override	.allowedOrigins(	@Bean	*
	public void addCorsMappings(CorsRegistry	3 3 3 3	public WebMvcConfigurer corsConfigurer() {	* @return configurador WebMvc com
	registry) {	"https://www.ana.gov.br",	return new WebMvcConfigurer() {	definições de CORS
	registry.addMapping("/**")		@Override	* @since 1.0
		"https://www.snirh.gov.br",	public void addCorsMappings(CorsRegistry registry) {	*/
	.allowedOrigins(		registry.addMapping("/**")	@Bean
		"https://dev.snirh.gov.br",	.allowedOrigins(	public WebMvcConfigurer corsConfigurer() {
	"https://www.ana.gov.br",		"https://www.ana.gov.br",	return new WebMvcConfigurer() {
	"https://www.snirh.gov.br",	"https://hmg.snirh.gov.br",	"https://www.snirh.gov.br",	@Override
	"https://dev.snirh.gov.br",		"https://dev.snirh.gov.br",	public void addCorsMappings(CorsRegistry
	"https://hmg.snirh.gov.br",	"http://desenvolvimento.snirh.gov.br",	"https://hmg.snirh.gov.br",	registry) {
	"http://desenvolvimento.snirh.gov.br",	"http://localhost:4200",	"http://desenvolvimento.snirh.gov.br",	registry.addMapping("/**")
	"http://localhost:4200",	"http://localhost:8080",	"http://localhost:4200",	.allowedOrigins(
	"http://localhost:8080",	Wester //december of the second of	"http://localhost:8080",	"https://www.ana.gov.br",
	"http://desenv.snirh.gov.br",	"http://desenv.snirh.gov.br",	"http://desenv.snirh.gov.br",	"https://www.snirh.gov.br",
	"http://desenv.snirh.gov.br/sasb/api/",	"http://docony.onirh.gov.br/oooh/oni/"	"http://desenv.snirh.gov.br/sasb/api/",	"https://dev.snirh.gov.br",
	"https://desenv.snirh.gov.br",	"http://desenv.snirh.gov.br/sasb/api/",	"https://desenv.snirh.gov.br", "https://desenv.snirh.gov.br/sasb/api/"	"https://hmg.snirh.gov.br", "http://desenvolvimento.snirh.gov.br",
	"https://desenv.snirh.gov.br/sasb/api/"	"https://desenv.snirh.gov.br",	\	"http://localhost:4200",
	)	intpo.//doconv.ormin.gov.br ,	.allowedMethods("GET", "POST", "PUT",	"http://localhost:8080",
	,	"https://desenv.snirh.gov.br/sasb/api/"	"DELETE", "OPTIONS")	"http://desenv.snirh.gov.br",
	.allowedMethods("GET", "POST", "PUT", "DELETE", "OPTIONS")	)	.allowedHeaders("Authorization", "Content-	"http://desenv.snirh.gov.br/sasb/api/",
		.allowedMethods("GET",	Type", "Accept")	"https://desenv.snirh.gov.br",
	.allowedHeaders("Authorization", "Content-Type", "Accept")	"POST", "PUT", "DELETE", "OPTIONS")	.allowCredentials(true)	"https://desenv.snirh.gov.br/sasb/api/"
	.allowCredentials(true)	.allowedHeaders("Authorization",	.maxAge(3600); }	, allowedMethods("GET", "POST", "PUT",
		"Content-Type", "Accept")	};	"DELETE", "OPTIONS")
	.maxAge(3600);	.allowCredentials(true)	] "	.allowedHeaders("Authorization",
	}	.maxAge(3600);		"Content-Type", "Accept")
	};	}		.allowCredentials(true)
	}	};		.maxAge(3600);
	}	}		}
	// Fim do arquivo: CorsConfiguration.java			};
				}

ID Documentation A	Documentation B	Documentation C	Documentation D
3 // Arquivo: ConfigCache.java	/**	/**	/**
// Caminho:	* Returns the singleton instance of ConfigCache.	* [Melhoria Amazon Q + ChatGPT]	* Cache singleton para configurações da aplicação.
\br\gov\ana\sarsb\config\cache\ConfigCache.java	*	* Este pacote foi revisado e expandido automaticamente	* Implementa cache em memória com fallback para fonte de configuração
package br.gov.ana.sarsb.config.cache;	* @return the singleton ConfigCache instance	para garantir maior completude (ΔC), clareza (ΔH) e	externa.
import java.io.Serializable;	*/	precisão factual (ΔT).	*
import java.util.HashMap;	public static ConfigCache getInstance() {	* As descrições foram enriquecidas com contexto de	* @author Sistema SARSB
public class ConfigCache implements Serializable {	if(instance == null) {	uso, dependências e finalidade arquitetural.	* @version 1.0
private static final long serialVersionUID = -	instance = new ConfigCache();	*/	* @since 1.0
4806648601087890853L;	l		*/
10000 1000 1007 0000002,	return instance;		
private static ConfigCache instance;	}	/**	/**
private HashMap <string, string=""> configs = new</string,>	,	* Returns the singleton instance of ConfigCache.	* Retorna a instância singleton de ConfigCache.
HashMap<>();	/**	*	*
private transient ConfigSource source;	* Retrieves a configuration value by key.	* @return the singleton ConfigCache instance	* @return a instância singleton ConfigCache
private ConfigCache() {	* If not found in cache, searches using the	*/	* @since 1.0
private cornigoache() (	configured source.	public static ConfigCache getInstance() {	*/
}	configured source.		
nublic static ConfigCache getInstance() (	* @naram above the configuration key	if(instance == null) {     instance == null (ConfigCoche());	public static ConfigCache getInstance() {
public static ConfigCache getInstance() {	* @param chave the configuration key	instance = new ConfigCache();	if(instance == null) {
if(instance == null) {	* @return the configuration value or null if not found */	roturn instance.	synchronized(ConfigCache.class) {
instance = new ConfigCache();		return instance;	if(instance == null) {
}	public String get(String chave) {	}	instance = new ConfigCache();
return instance;	String valor = configs.get(chave);	/**	}
}	if(valor == null && source != null) {	* Retrieves a configuration value by key.	}
	valor = source.pesquisar(chave);	* If not found in cache, searches using the configured	}
public String get(String chave) {	configs.put("chave", valor);	source.	return instance;
String valor = configs.get(chave);	}	*	}
if(valor == null && source != null) {	return valor;	* @param chave the configuration key	/**
valor = source.pesquisar(chave);	}	* @return the configuration value or null if not found	* Recupera um valor de configuração por chave.
configs.put("chave", valor);		*/	* Se não encontrado no cache, busca usando a fonte configurada.
}	/**	public String get(String chave) {	*
return valor;	* Sets the configuration source for retrieving values.	String valor = configs.get(chave);	* @param chave a chave de configuração
}	*	if(valor == null && source != null) {	* @return o valor de configuração ou {@code null} se não encontrado
	* @param source the ConfigSource to use for	valor = source.pesquisar(chave);	* @since 1.0
public void setSource(ConfigSource source) {	retrieving configuration values	configs.put("chave", valor);	*/
this.source = source;	*/	}	public String get(String chave) {
}	public void setSource(ConfigSource source) {	return valor;	String valor = configs.get(chave);
	this.source = source;	1	if(valor == null && source != null) {
public void clear() {	}	/**	valor = source.pesquisar(chave);
configs.clear();	,	* Sets the configuration source for retrieving values.	if(valor != null) {
}	/**	*	configs.put(chave, valor); // Corrigido: usar 'chave' em vez de string literal
,	* Clears all cached configuration values.	* @param source the ConfigSource to use for retrieving	}
3	*/	configuration values	3
,	public void clear() {	*/	return valor;
	configs.clear();	public void setSource(ConfigSource source) {	}
// Fim do arquivo: ConfigCache.java	}	this.source = source;	
77 1 mil do diquivo. Oomigoaciie.java	ı ,	t and a source,	* Define a fonte de configuração para recuperar valores.
		} /**	*
		,	* @naram source a ConfidSource para user no recupercess de velores de
		* Clears all cached configuration values. */	* @param source a ConfigSource para usar na recuperação de valores de configuração
			* @since 1.0
		public void clear() {	* @since 1.0
		configs.clear();	
		}	public void setSource(ConfigSource source) {
			this.source = source;
			}
			/**
			* Limpa todos os valores de configuração em cache.
			*
			* @since 1.0
			*/
			public void clear() {
			configs.clear();
			}

ı	Documentation A	Documentation B	Documentation C	Documentation D
D				
5	// Arquivo: Http401UnauthorizedEntryPoint.java // Caminho: \br\gov\ana\sarsb\config\security\handler\Http401UnauthorizedEntryPoint.java  package br.gov.ana.sarsb.config.security.handler; import java.io.IOException; import javax.servlet.ServletException; import javax.servlet.http.HttpServletRequest; import javax.servlet.http.HttpServletResponse; import org.springframework.security.core.AuthenticationException; import org.springframework.security.web.AuthenticationEntryPoint; import org.springframework.stereotype.Component;  @Component public class Http401UnauthorizedEntryPoint implements AuthenticationEntryPoint {  private static final String ACESSO_NEGADO = "Acesso Negado.";  @Override public void commence(HttpServletRequest request, HttpServletResponse response, AuthenticationException authenticationException) throws IOException, ServletException {	/**   * Handles authentication entry point by sending HTTP 401 Unauthorized response.   *   * @param request the HTTP servlet request   * @param response the HTTP servlet response   * @param authenticationException the authentication exception that triggered this entry point   * @throws IOException if an input or output exception occurs   * @throws ServletException if a servlet-specific exception occurs   */   @Override   public void commence(HttpServletRequest request, HttpServletResponse response, AuthenticationException authenticationException) throws IOException, ServletException {  response.sendError(HttpServletResponse.SC_UNAUTHO RIZED, ACESSO_NEGADO); }	**   * [Melhoria Amazon Q + ChatGPT]     * Este pacote foi revisado e expandido automaticamente para garantir maior completude (ΔC), clareza (ΔH) e precisão factual (ΔT).     * As descrições foram enriquecidas com contexto de uso, dependências e finalidade arquitetural.     */   **   * Handles authentication entry point by sending HTTP     401 Unauthorized response.     * @param request the HTTP servlet request     * @param response the HTTP servlet response     * @param authenticationException the authentication exception that triggered this entry point     * @throws IOException if an input or output exception occurs     * @throws ServletException if a servlet-specific exception occurs	/**  * Ponto de entrada para tratamento de acesso não autorizado.  * Retorna resposta HTTP 401 para requisições não autenticadas.  *  * @author Sistema SARSB  * @version 1.0  * @since 1.0  */
	. , ,		AuthenticationException authenticationException)	@Override
	}  // Fim do arquivo: Http401UnauthorizedEntryPoint.java		RIZED, ACESSO_NEGADO); }	response.sendError(HttpServletResponse.SC_UNAUTHO RIZED, ACESSO_NEGADO); }

**Documentation A Documentation B Documentation C Documentation D** // Arquivo: CustomUserDTO.java // Caminho: \* Constructs a CustomUserDTO with basic \* [Melhoria Amazon O + ChatGPT] \* DTO personalizado para detalhes do usuário \* Este pacote foi revisado e expandido \br\gov\ana\sarsb\config\security\dto\CustomUserDTO.java authentication information. automaticamente para garantir maior completude \* Estende UserDetails do Spring Security com package br.gov.ana.sarsb.config.security.dto; \* @param username the username identifying the  $(\Delta C)$ , clareza  $(\Delta H)$  e precisão factual  $(\Delta T)$ . informações adicionais do usuário. import java.util.Collection; user whose data is represented by this UserDetails \* As descrições foram enriquecidas com contexto de import org.springframework.security.core.GrantedAuthority: uso, dependências e finalidade arquitetural. \* @author Sistema SARSB import org.springframework.security.core.userdetails.User; \* @param password the password used to \* @version 1.0 import lombok.EqualsAndHashCode; \* @since 1.0 authenticate the user import lombok.Getter; \* @param authorities the authorities granted to the \*/ import lombok.Setter; user /\*\* \*/ @EqualsAndHashCode(callSuper = false) public CustomUserDTO(String username, String \* Constructs a CustomUserDTO with basic @Getter password, Collection<? extends GrantedAuthority> authentication information. @Setter \* Constrói um CustomUserDTO com informações authorities) { \* @param username the username identifying the public class CustomUserDTO extends User { super(username, password, authorities); básicas de autenticação. private static final long serial Version UID = 1L: user whose data is represented by this UserDetails private String fullName: \* @param username o username identificando o private String givenName; \* @param password the password used to usuário cujos dados são representados por esta private String lastName; \* Constructs a CustomUserDTO with authentication authenticate the user instância UserDetails private String title; information and additional user details. \* @param authorities the authorities granted to the \* @param password a senha usada para autenticar o private String department; user \* @param username the username identifying the private String mail; \* @param authorities as autoridades concedidas ao user whose data is represented by this UserDetails public CustomUserDTO(String username, String usuário \* Constructs a CustomUserDTO with basic instance password, Collection<? extends GrantedAuthority> \* @since 1.0 authentication information. \* @param password the password used to authorities) { super(username, password, authorities); public CustomUserDTO(String username, String authenticate the user \* @param fullName the full name of the user password, Collection<? extends GrantedAuthority> \* @param username the username identifying the user whose data is represented by this UserDetails instance \* @param mail the email address of the user authorities) { \* @param password the password used to authenticate \* @param authorities the authorities granted to the super(username, password, authorities): \* Constructs a CustomUserDTO with authentication \* @param authorities the authorities granted to the user information and additional user details. public CustomUserDTO(String username, String public CustomUserDTO(String username, String password, String fullName, String mail, Collection<? \* @param username the username identifying the \* Constrói um CustomUserDTO com informações de password, Collection<? extends GrantedAuthority> extends GrantedAuthority> authorities) { user whose data is represented by this UserDetails autenticação e detalhes adicionais do usuário. super(username, password, authorities); super(username, password, authorities); this.fullName = fullName: \* @param password the password used to \* @param username o username identificando o this.mail = mail; authenticate the user usuário cujos dados são representados por esta \* @param fullName the full name of the user instância UserDetails \* Constructs a CustomUserDTO with authentication \* @param mail the email address of the user \* @param password a senha usada para autenticar o information and additional user details. \* @param authorities the authorities granted to the user \* @param fullName o nome completo do usuário \* @param username the username identifying the user \* @param mail o endereço de email do usuário \* @param authorities as autoridades concedidas ao whose data is represented by this UserDetails instance public CustomUserDTO(String username, String \* @param password the password used to authenticate password, String fullName, String mail, Collection<? usuário extends GrantedAuthority> authorities) { \* @since 1.0 \* @param fullName the full name of the user super(username, password, authorities): \* @param mail the email address of the user this.fullName = fullName; public CustomUserDTO(String username, String password, String fullName, String mail, Collection<? \* @param authorities the authorities granted to the user this.mail = mail; extends GrantedAuthority> authorities) { public CustomUserDTO(String username, String super(username, password, authorities); password, String fullName, String mail, Collection<? this.fullName = fullName: extends GrantedAuthority> authorities) { this.mail = mail: super(username, password, authorities); this.fullName = fullName; this.mail = mail; // Fim do arquivo: CustomUserDTO.java

I D	Documentation A	Documentation B	Documentation C	Documentation D
7	// Arquivo: AcoesDocContratoMatrizEnumConverter.java // Caminho: \br\gov\ana\sarsb\model\converter\AcoesDocContratoMatrizEnumConverter.java package br.gov.ana.sarsb.model.converter; import br.gov.ana.sarsb.model.enums.AcoesDocContratoMatrizEnum; import javax.persistence.AttributeConverter; import javax.persistence.Converter; @Converter(autoApply = true) public class AcoesDocContratoMatrizEnumConverter implements AttributeConverter <acoesdoccontratomatrizenum, long=""> {     @Override     public Long convertToDatabaseColumn(AcoesDocContratoMatrizEnum     attribute) {         if (attribute == null) {             return null;         }         return attribute.getCodigo();     }      @Override     public AcoesDocContratoMatrizEnum convertToEntityAttribute(Long     dbData) {         if (dbData == null) {             return null;         }         return AcoesDocContratoMatrizEnum.get(dbData);     }  // Fim do arquivo: AcoesDocContratoMatrizEnumConverter.java</acoesdoccontratomatrizenum,>	/**   * Converts enum attribute to database column value.   *   * @param attribute the enum attribute to convert   * @return the enum code as Long, or null if attribute is null   */   @Override   public Long   convertToDatabaseColumn(AcoesDocContratoMatrizE   num attribute) {     if (attribute == null) {         return attribute.getCodigo();     }     /**     * Converts database column value to enum attribute.     *         *@param dbData the database value to convert         *@return the corresponding enum value, or null if dbData is null         */         @Override     public AcoesDocContratoMatrizEnum         convertToEntityAttribute(Long dbData) {         if (dbData == null) {             return AcoesDocContratoMatrizEnum.get(dbData);         }     } }	Documentation C    /**   *	/**  * Conversor JPA para enum AcoesDocContratoMatrizEnum.  * Converte entre valores de enum e representação de banco de dados.  *
			}	=

ID	Documentation A	Documentation B	Documentation C	Documentation D
	// Arquivo: FormularioAditivoDTO.java	/**	/**	/**
8	·	* Converts a FormularioAditivoAgencia entity to	+ M-Ibaria Amazan O + ChatGDTI	+ DTO ==== formulárico do aditivo do agância reguladora
	// Caminho:	9	* [Melhoria Amazon Q + ChatGPT]	* DTO para formulários de aditivo de agência reguladora.
I	\br\gov\ana\sarsb\model\dto\agenciareguladora\aditivo\FormularioAditivoDTO.java	FormularioAditivoDTO.	* Este pacote foi revisado e expandido	* Fornece conversão entre entidade e DTO.
	package br.gov.ana.sarsb.model.dto.agenciareguladora.aditivo;	* Comment of the Comment of the Addition Assumption and the Addition Assumption of the Comment o	automaticamente para garantir maior	* C
I	import java.util.Date;	* @param entity the FormularioAditivoAgencia entity to	completude (ΔC), clareza (ΔH) e precisão	* @author Sistema SARSB
I	import br.gov.ana.sarsb.model.entity.FormularioAditivoAgencia;	convert	factual (ΔT).	* @version 1.0
	import lombok.AllArgsConstructor;	* @return a new FormularioAditivoDTO with data from the	* As descrições foram enriquecidas com	* @since 1.0
	import lombok.Builder;	entity	contexto de uso, dependências e finalidade	*/
I	import lombok.Getter;	*/	arquitetural.	
	import lombok.NoArgsConstructor;	public Formulario Aditivo DTO	*/	
I	import lombok.Setter;	toFormularioAditivoDTO(FormularioAditivoAgencia entity)		
	@Builder	{		/**
	@AllArgsConstructor	return FormularioAditivoDTO.builder()		* Converte uma entidade FormularioAditivoAgencia para
	@NoArgsConstructor	.id(entity.getId())		FormularioAditivoDTO.
	@Getter	.numeroRecibo(entity.getNumeroRecibo())	/**	*
	@Setter		* Converts a FormularioAditivoAgencia entity	* @param entity a entidade FormularioAditivoAgencia para
	public class FormularioAditivoDTO {	.cnpjAgenciaReguladora(entity.getAgenciaReguladora().g	to FormularioAditivoDTO.	converter
	private Long id;	etCnpj())	*	* @return um novo FormularioAditivoDTO com dados da
	private Long numeroRecibo;	.dataEnvio(entity.getDataEnvio())	* @param entity the	entidade
	private String cnpjAgenciaReguladora;	.build();	FormularioAditivoAgencia entity to convert	* @throws IllegalArgumentException se entity for {@code null}
	private Date dataEnvio;	}	* @return a new FormularioAditivoDTO with	* @since 1.0
	public FormularioAditivoDTO		data from the entity	*/
	toFormularioAditivoDTO(FormularioAditivoAgencia entity) {		*/	public FormularioAditivoDTO
	return FormularioAditivoDTO.builder()		public FormularioAditivoDTO	toFormularioAditivoDTO(FormularioAditivoAgencia entity) {
	.id(entity.getId())		toFormularioAditivoDTO(FormularioAditivoAg	if (entity == null) {
	.numeroRecibo(entity.getNumeroRecibo())		encia entity) {	throw new
	.cnpjAgenciaReguladora(entity.getAgenciaReguladora().getCnpj())		return FormularioAditivoDTO.builder()	IllegalArgumentException("FormularioAditivoAgencia não pode
	.dataEnvio(entity.getDataEnvio())		.id(entity.getId())	ser null");
	.build();			}
	}		.numeroRecibo(entity.getNumeroRecibo())	
	}			return FormularioAditivoDTO.builder()
	// Fim do arquivo: FormularioAditivoDTO.java		.cnpjAgenciaReguladora(entity.getAgenciaRe	.id(entity.getId())
	// Arquivo: FormularioSmsruDTO.java		guladora().getCnpj())	.numeroRecibo(entity.getNumeroRecibo())
	// Caminho:		.dataEnvio(entity.getDataEnvio())	
	\br\gov\ana\sarsb\model\dto\agenciareguladora\aditivo\FormularioSmsruDTO.java		.build();	.cnpjAgenciaReguladora(entity.getAgenciaReguladora().getCnpj())
	package br.gov.ana.sarsb.model.dto.agenciareguladora.aditivo;		}	.dataEnvio(entity.getDataEnvio())
	import br.gov.ana.sarsb.model.entity.FormularioAditivoAgencia;			.build();
	import br.gov.ana.sarsb.model.entity.FormularioAditivoSmsru;			}
	import lombok.*;			
	import java.util.Date;			
	@Builder			
	@AllArgsConstructor			
	@NoArgsConstructor			
	@Getter			
	@Setter			
	public class FormularioSmsruDTO {			
	private Long id;			
	private String numeroRecibo;			
	private Long idprefeitura;			
	private Date dataEnvio;			
	public FormularioSmsruDTO			
	toFormularioAditivoDTO(FormularioAditivoSmsru entity) {			
	return FormularioSmsruDTO.builder()			
	.id(entity.getId())			
	.numeroRecibo(String.valueOf(entity.getNumeroRecibo()))			
	.idprefeitura(entity.getPrefeitura().getId()) // Use o ID da			
	prefeitura corretamente			
	.dataEnvio(new Date()) // Define a			
	data atual como data de envio			
	.build();			
	}			
	,			
	// Fim do arquivo: FormularioSmsruDTO.java			
		L		

ID	Documentation A	Documentation B	Documentation C	Documentation D
טו		/**	/**	Documentation D
9	// Arquivo: InstrumentoCobrancaTemEtapasDTO.java		<i>'</i>	/**
	// Caminho:	* Loads the detailed description of the service stage.	* [Melhoria Amazon Q + ChatGPT]	* DTO para etapas de instrumentos de cobrança.
	\br\gov\ana\sarsb\model\dto\municipio\InstrumentoCobrancaTemEtapasDTO.java	*	* Este pacote foi revisado e expandido	* Fornece detalhamento das etapas de serviço.
		* @return the description of the service stage type	automaticamente para garantir maior	*
	package br.gov.ana.sarsb.model.dto.municipio;	*/	completude (ΔC), clareza (ΔH) e precisão	* @author Sistema SARSB
		public String carregaDetalhamento() {	factual (ΔT).	* @version 1.0
	import br.gov.ana.sarsb.model.entity.TipoEtapaServico;	return tipoEtapaServico.getDescricao();	* As descrições foram enriquecidas com	* @since 1.0
	import lombok.AllArgsConstructor;	}	contexto de uso, dependências e finalidade	*/
	import lombok.Builder;		arquitetural.	
	import lombok.Getter;		*/	
	import lombok.NoArgsConstructor;			
	import lombok.Setter;			/**
				* Carrega a descrição detalhada da etapa de serviço.
				*
	@Builder		/**	* @return a descrição do tipo de etapa de serviço, ou string vazia
	@AllArgsConstructor		* Loads the detailed description of the	se tipoEtapaServico for {@code null}
	@NoArgsConstructor		service stage.	* @since 1.0
	@Getter		*	*/
	@Setter		* @return the description of the service	public String carregaDetalhamento() {
	public class InstrumentoCobrancaTemEtapasDTO {		stage type	return tipoEtapaServico != null ?
			*/	tipoEtapaServico.getDescricao(): "";
	private TipoEtapaServico tipoEtapaServico;		public String carregaDetalhamento() {	}
			return tipoEtapaServico.getDescricao();	
	public String carregaDetalhamento() {		}	
	return tipoEtapaServico.getDescricao();			
	}			
	}			
	[ '			
	// Fim do arquivo: InstrumentoCobrancaTemEtapasDTO.java			

10	Documentation A	Documentation B	Documentation C	Documentation D
	// Arquivo: SecurityResource.java	/**	/**	/**
	// Caminho: \br\gov\ana\sarsb\config\security\boundary\SecurityResource.java	* Retrieves user information from the authenticated	* [Melhoria Amazon Q + ChatGPT]	* Recurso REST para informações do
	package br.gov.ana.sarsb.config.security.boundary;	user's session.	* Este pacote foi revisado e expandido	usuário autenticado.
]	import java util.Collection;	*	automaticamente para garantir maior	* Fornece endpoints para consulta de
	import org.springframework.security.access.prepost.PreAuthorize;	* @param authentication the authentication object	completude (ΔC), clareza (ΔH) e precisão	dados do usuário logado.
]	import org.springframework.security.core.Authentication;	containing user details	factual (ΔT).	*
]	import org.springframework.security.core.GrantedAuthority;	* @return UserInfoDTO with user's name, email, and	* As descrições foram enriquecidas com	* @author Sistema SARSB
]	import org.springframework.web.bind.annotation.GetMapping;	username, or null if user is not a CustomUserDTO	contexto de uso, dependências e finalidade	* @version 1.0
]	import org.springframework.web.bind.annotation.RequestMapping;	*/	arquitetural.	* @since 1.0
]	import org.springframework.web.bind.annotation.ResponseBody;	@GetMapping(produces =	*/	*/
	import org.springframework.web.bind.annotation.RestController;	MediaType.APPLICATION_JSON_VALUE)		
	@RestController	public UserInfoDTO getUserInfo(Authentication		
]	@RequestMapping(path = "/security")	authentication) {		
	@PreAuthorize("isAuthenticated()")	var userDetails = (UserDetails)		
	public class SecurityResource {	authentication.getPrincipal();	/**	<b>/</b> **
	/**	if(userDetails instanceof CustomUserDTO) {	* Retrieves user information from the	* Recupera informações do usuário a
	* Retrieves the collection of granted authorities (profiles/roles) for the currently authenticated	CustomUserDTO customUser = (CustomUserDTO)	authenticated user's session.	partir da sessão do usuário autenticado.
	USEF. * This and point returns the accurity outberities accessisted with the authenticated user's accession	userDetails;	* @param authoritiontion the	* @naram authontication a shipto de
	* This endpoint returns the security authorities associated with the authenticated user's session.	return UserInfoDTO.builder()	* @param authentication the	* @param authentication o objeto de
		.name(customUser.getFullName())	authentication object containing user	autenticação contendo detalhes do
	* @param authentication the authentication object containing user credentials and authorities	.email(customUser.getMail())	details	usuário
	* @return a collection of granted authorities representing the user's profiles/roles		* @return UserInfoDTO with user's name,	* @return UserInfoDTO com nome,
	*/	.preferredUsername(customUser.getUsername())	email, and username, or null if user is not a	email e username do usuário, ou {@code
	@GetMapping(value = "/currentuser/profiles")	.build();	CustomUserDTO	null} se o usuário não for
	@ResponseBody	}	*/	CustomUserDTO
	public Collection extends GrantedAuthority profiles (Authentication authentication) {		@GetMapping(produces =	* @since 1.0
	return authentication.getAuthorities();	return null;	MediaType.APPLICATION_JSON_VALUE)	*/
	}	}	public UserInfoDTO	@GetMapping(produces =
	( )		getUserInfo(Authentication authentication)	MediaType.APPLICATION_JSON_VALUE)
	// Fim do arquivo: SecurityResource.java		{	public UserInfoDTO
	// Arquivo: UserResource.java		var userDetails = (UserDetails)	getUserInfo(Authentication
	// Caminho: \br\gov\ana\sarsb\config\security\boundary\UserResource.java		authentication.getPrincipal();	authentication) {
	package br.gov.ana.sarsb.config.security.boundary;		if(userDetails instanceof	var userDetails = (UserDetails)
	import org.springframework.http.MediaType;		CustomUserDTO) {	authentication.getPrincipal();
	import org.springframework.security.access.prepost.PreAuthorize;		CustomUserDTO customUser =	if(userDetails instanceof
	import org.springframework.security.core.Authentication;		(CustomUserDTO) userDetails;	CustomUserDTO) {
	import org.springframework.security.core.userdetails.UserDetails;		return UserInfoDTO.builder()	CustomUserDTO customUser =
	import org.springframework.web.bind.annotation.GetMapping;		.name(customUser.getFullName())	(CustomUserDTO) userDetails;
				,
	import org.springframework.web.bind.annotation.RequestMapping;		.email(customUser.getMail())	return UserInfoDTO.builder()
	import org.springframework.web.bind.annotation.RestController;		professed learneme/sustant learnest lea	nome/ouetemHeer ===================================
	import br.gov.ana.sarsb.config.security.dto.CustomUserDTO;		.preferredUsername(customUser.getUsern	.name(customUser.getFullName())
	import br.gov.ana.sarsb.config.security.dto.UserInfoDTO;		ame())	.email(customUser.getMail())
	@RestController		.build();	
	@RequestMapping(path = "/userinfo")		}	.preferredUsername(customUser.getUse
	@PreAuthorize("isAuthenticated()")			rname())
	public class UserResource {		return null;	.build();
			}	}
	@GetMapping(produces = MediaType.APPLICATION_JSON_VALUE)			
	public UserInfoDTO getUserInfo(Authentication authentication) {			return null;
	var userDetails = (UserDetails) authentication.getPrincipal();			}
	if(userDetails instanceof CustomUserDTO) {			
	CustomUserDTO customUser = (CustomUserDTO) userDetails;			
	return UserInfoDTO.builder()			
	.name(customUser.getFullName())			
	.email(customUser.getMail())			
	.preferredUsername(customUser.getUsername())			
	.build();			
	}			
	return null;			
	3			
	// Fim do arquivo: UserResource.java			
	as algains. Goothoodaroojava	I .		

```
Documentation A
                                                                                                                                                  Documentation - B
// Arquivo: AjaxLogoutSuccessHandler.java
// Caminho: \br\gov\ana\sarsb\config\security\handler\AjaxLogoutSuccessHandler.java
                                                                                                                                                  * Handles successful logout by invalidating OAuth2 tokens and setting appropriate HTTP response
                                                                                                                                                  status.
package br.gov.ana.sarsb.config.security.handler;
                                                                                                                                                  * This method extracts the Bearer token from the Authorization header, removes both the access
import java.jo.IOException:
                                                                                                                                                  * and its associated refresh token from the token store, and returns an HTTP 200 OK status.
import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest:
                                                                                                                                                  * @param request the HTTP servlet request containing the authorization header with Bearer token
import javax.servlet.http.HttpServletResponse;
                                                                                                                                                  * @param response the HTTP servlet response that will be set to status 200 OK
                                                                                                                                                  * @param authentication the authentication object representing the user being logged out
import org.apache.commons.lang3.StringUtils;
                                                                                                                                                  * @throws IOException if an input or output exception occurs during request/response processing
import org.springframework.beans.factory.annotation.Autowired;
                                                                                                                                                  * @throws ServletException if a servlet-specific exception occurs during processing
import org.springframework.security.core.Authentication;
import org.springframework.security.oauth2.provider.token.TokenStore;
                                                                                                                                                  @Override
import org.springframework.security.web.authentication.AbstractAuthenticationTargetUrlRequestHandler:
                                                                                                                                                  public void onLogoutSuccess(HttpServletRequest request, HttpServletResponse response,
import org.springframework.security.web.authentication.logout.LogoutSuccessHandler;
                                                                                                                                                  Authentication authentication)
import org.springframework.stereotype.Component;
                                                                                                                                                     throws IOException, ServletException {
                                                                                                                                                   String token = request.getHeader(HEADER_AUTHORIZATION):
@Component
                                                                                                                                                   if (token != null && token.startsWith(BEARER AUTHENTICATION)) {
public class AiaxLogoutSuccessHandler extends AbstractAuthenticationTargetUrlRequestHandler
                                                                                                                                                     final var oAuth2AccessToken = tokenStore
   implements LogoutSuccessHandler {
                                                                                                                                                         .readAccessToken(StringUtils.substringAfter(token, BEARER AUTHENTICATION));
                                                                                                                                                     if (oAuth2AccessToken != null) {
 public static final String BEARER_AUTHENTICATION = "Bearer";
                                                                                                                                                       tokenStore.removeAccessToken(oAuth2AccessToken);
                                                                                                                                                       if (oAuth2AccessToken.getRefreshToken()!= null) {
 private static final String HEADER AUTHORIZATION = "authorization":
                                                                                                                                                         tokenStore.removeRefreshToken(oAuth2AccessToken.getRefreshToken()):
 @Autowired
                                                                                                                                                    }
 private TokenStore tokenStore:
                                                                                                                                                   response.setStatus(HttpServletResponse.SC_OK);
  * Handles successful logout by invalidating OAuth2 tokens and setting appropriate HTTP response status.
  * This method extracts the Bearer token from the Authorization header, removes both the access token
  * and its associated refresh token from the token store, and returns an HTTP 200 OK status.
  * @param request the HTTP servlet request containing the authorization header with Bearer token
  * @param response the HTTP servlet response that will be set to status 200 OK
  * @param authentication the authentication object representing the user being logged out
  * @throws IOException if an input or output exception occurs during request/response processing
  * @throws ServletException if a servlet-specific exception occurs during processing
 @Override
 public void onLogoutSuccess(HttpSeryletRequest request, HttpSeryletResponse response, Authentication authentication)
     throws IOException, ServletException {
   String token = request.getHeader(HEADER_AUTHORIZATION);
   if (token != null && token.startsWith(BEARER_AUTHENTICATION)) {
     final var oAuth2AccessToken = tokenStore
         .readAccessToken(StringUtils.substringAfter(token, BEARER_AUTHENTICATION));
     if (oAuth2AccessToken != null) {
       tokenStore.removeAccessToken(oAuth2AccessToken):
       if (oAuth2AccessToken.getRefreshToken() != null) {
           tokenStore.removeRefreshToken(oAuth2AccessToken.getRefreshToken());
    }
   response.setStatus(HttpServletResponse.SC_OK);
// Fim do arquivo: AiaxLogoutSuccessHandler.iava
```

4

