ID	Documentation A	Documentation B	Documentation C	Documentation D

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

ID	Documentation A	Documentation B	Documentation C	Documentation D

2	// Arquivo: CorsConfiguration.java
	// Caminho: \br\gov\ana\sarsb\config\CorsConfiguration.java
	
	package br.gov.ana.sarsb.config;
	
	import org.springframework.context.annotation.Bean;
	import org.springframework.context.annotation.Configuration;
	import org.springframework.web.servlet.config.annotation.CorsRegistry;
	import
	org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
	
	@Configuration
	public class CorsConfiguration {
	
	@Bean
	public WebMvcConfigurer corsConfigurer() {
	return new WebMvcConfigurer() {
	@Override
	public void addCorsMappings(CorsRegistry
	registry) {
	registry.addMapping("/**")
	
	.allowedOrigins(
	
	"https://www.ana.gov.br",
	"https://www.snirh.gov.br",
	"https://dev.snirh.gov.br",
	"https://hmg.snirh.gov.br",
	"http://desenvolvimento.snirh.gov.br",
	"http://localhost:4200",
	"http://localhost:8080",
	"http://desenv.snirh.gov.br",
	"http://desenv.snirh.gov.br/sasb/api/",
	"https://desenv.snirh.gov.br",
	
	"https://desenv.snirh.gov.br/sasb/api/"
	)
	
	.allowedMethods("GET", "POST", "PUT", "DELETE", "OPTIONS")
	
	.allowedHeaders("Authorization", "Content-Type", "Accept")
	
	.allowCredentials(true)
	
	.maxAge(3600);
	}
	};
	}
	}
	// Fim do arquivo: CorsConfiguration.java
		/**
		* Configures CORS (Cross-Origin
		Resource Sharing) settings for the
		application.
		*
		* @return WebMvcConfigurer with
		CORS configuration
		*/
		@Bean
		public WebMvcConfigurer
		corsConfigurer() {
		return new WebMvcConfigurer() {
		@Override
		public void
		addCorsMappings(CorsRegistry
		registry) {
		registry.addMapping("/**")
		.allowedOrigins(
		
		"https://www.ana.gov.br",
		
		"https://www.snirh.gov.br",
		
		"https://dev.snirh.gov.br",
		
		"https://hmg.snirh.gov.br",
		
		"http://desenvolvimento.snirh.gov.br",
		"http://localhost:4200",
		"http://localhost:8080",
		
		"http://desenv.snirh.gov.br",
		
		"http://desenv.snirh.gov.br/sasb/api/",
		
		"https://desenv.snirh.gov.br",
		
		"https://desenv.snirh.gov.br/sasb/api/"
		)
		.allowedMethods("GET",
		"POST", "PUT", "DELETE", "OPTIONS")
		
		.allowedHeaders("Authorization",
		"Content-Type", "Accept")
		.allowCredentials(true)
		.maxAge(3600);
		}
		};
		}
			/**
			* [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.
			*/
			
			--- 	
			
			/**
			* Configures CORS (Cross-Origin Resource Sharing)
			settings for the application.
			*
			* @return WebMvcConfigurer with CORS configuration
			*/
			@Bean
			public WebMvcConfigurer corsConfigurer() {
			return new WebMvcConfigurer() {
			@Override
			public void addCorsMappings(CorsRegistry registry) {
			registry.addMapping("/**")
			.allowedOrigins(
			"https://www.ana.gov.br",
			"https://www.snirh.gov.br",
			"https://dev.snirh.gov.br",
			"https://hmg.snirh.gov.br",
			"http://desenvolvimento.snirh.gov.br",
			"http://localhost:4200",
			"http://localhost:8080",
			"http://desenv.snirh.gov.br",
			"http://desenv.snirh.gov.br/sasb/api/",
			"https://desenv.snirh.gov.br",
			"https://desenv.snirh.gov.br/sasb/api/"
			)
			.allowedMethods("GET", "POST", "PUT",
			"DELETE", "OPTIONS")
			.allowedHeaders("Authorization", "Content-
			Type", "Accept")
			.allowCredentials(true)
			.maxAge(3600);
			}
			};
			}
				/**
				* Configuração de CORS (Cross-Origin
				Resource Sharing) para a aplicação.
				* Define políticas de acesso cross-origin para
				APIs REST.
				*
				* @author Sistema SARSB
				* @version 1.0
				* @since 1.0
				*/
				--- 	
				
				/**
				* Configura as definições de CORS para a
				aplicação.
				* Permite acesso de origens específicas com
				métodos HTTP definidos.
				*
				* @return configurador WebMvc com
				definições de CORS
				* @since 1.0
				*/
				@Bean
				public WebMvcConfigurer corsConfigurer() {
				return new WebMvcConfigurer() {
				@Override
				public void addCorsMappings(CorsRegistry
				registry) {
				registry.addMapping("/**")
				.allowedOrigins(
				"https://www.ana.gov.br",
				"https://www.snirh.gov.br",
				"https://dev.snirh.gov.br",
				"https://hmg.snirh.gov.br",
				"http://desenvolvimento.snirh.gov.br",
				"http://localhost:4200",
				"http://localhost:8080",
				"http://desenv.snirh.gov.br",
				"http://desenv.snirh.gov.br/sasb/api/",
				"https://desenv.snirh.gov.br",
				"https://desenv.snirh.gov.br/sasb/api/"
				)
				.allowedMethods("GET", "POST", "PUT",
				"DELETE", "OPTIONS")
				.allowedHeaders("Authorization",
				"Content-Type", "Accept")
				.allowCredentials(true)
				.maxAge(3600);
				}
				};
				}

ID	Documentation A	Documentation B	Documentation C	Documentation D

3	// Arquivo: ConfigCache.java
	// Caminho:
	\br\gov\ana\sarsb\config\cache\ConfigCache.java
	package br.gov.ana.sarsb.config.cache;
	import java.io.Serializable;
	import java.util.HashMap;
	public class ConfigCache implements Serializable {
	  private static final long serialVersionUID = -4806648601087890853L;
	
	  private static ConfigCache instance;
	  private HashMap<String, String> configs = new HashMap<>();
	  private transient ConfigSource source;
	  private ConfigCache() {
	  }
	
	  public static ConfigCache getInstance() {
	  if(instance == null) {
	  instance = new ConfigCache();
	  }
	  return instance;
	  }
	
	  public String get(String chave) {
	  String valor = configs.get(chave);
	  if(valor == null && source != null) {
	  valor = source.pesquisar(chave);
	  configs.put("chave", valor);
	  }
	  return valor;
	  }
	
	  public void setSource(ConfigSource source) {
	  this.source = source;
	  }
	
	  public void clear() {
	  configs.clear();
	  }
	
	}
	
	// Fim do arquivo: ConfigCache.java
		/**
		 * Returns the singleton instance of ConfigCache.
		 *
		 * @return the singleton ConfigCache instance
		 */
		public static ConfigCache getInstance() {
		  if(instance == null) {
		  instance = new ConfigCache();
		  }
		  return instance;
		}
		
		/**
		 * Retrieves a configuration value by key.
		 * If not found in cache, searches using the configured source.
		 *
		 * @param chave the configuration key
		 * @return the configuration value or null if not found
		 */
		public String get(String chave) {
		  String valor = configs.get(chave);
		  if(valor == null && source != null) {
		  valor = source.pesquisar(chave);
		  configs.put("chave", valor);
		  }
		  return valor;
		}
		
		/**
		 * Sets the configuration source for retrieving values.
		 *
		 * @param source the ConfigSource to use for retrieving configuration values
		 */
		public void setSource(ConfigSource source) {
		  this.source = source;
		}
		
		/**
		 * Clears all cached configuration values.
		 */
		public void clear() {
		  configs.clear();
		}
			/**
			 * [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.
			 */
			
			--- 	
			/**
			 * Returns the singleton instance of ConfigCache.
			 *
			 * @return the singleton ConfigCache instance
			 */
			public static ConfigCache getInstance() {
			  if(instance == null) {
			  instance = new ConfigCache();
			  }
			  return instance;
			}
			/**
			 * Retrieves a configuration value by key.
			 * If not found in cache, searches using the configured source.
			 *
			 * @param chave the configuration key
			 * @return the configuration value or null if not found
			 */
			public String get(String chave) {
			  String valor = configs.get(chave);
			  if(valor == null && source != null) {
			  valor = source.pesquisar(chave);
			  configs.put("chave", valor);
			  }
			  return valor;
			}
			/**
			 * Sets the configuration source for retrieving values.
			 *
			 * @param source the ConfigSource to use for retrieving configuration values
			 */
			public void setSource(ConfigSource source) {
			  this.source = source;
			}
			/**
			 * Clears all cached configuration values.
			 */
			public void clear() {
			  configs.clear();
			}
				/**
				 * Cache singleton para configurações da aplicação.
				 * Implementa cache em memória com fallback para fonte de configuração externa.
				 *
				 * @author Sistema SARSB
				 * @version 1.0
				 * @since 1.0
				 */
				--- 	
				/**
				 * Retorna a instância singleton de ConfigCache.
				 *
				 * @return a instância singleton ConfigCache
				 * @since 1.0
				 */
				public static ConfigCache getInstance() {
				  if(instance == null) {
				  synchronized(ConfigCache.class) {
				  if(instance == null) {
				  instance = new ConfigCache();
				  }
				  }
				  }
				  return instance;
				}
				/**
				 * Recupera um valor de configuração por chave.
				 * Se não encontrado no cache, busca usando a fonte configurada.
				 *
				 * @param chave a chave de configuração
				 * @return o valor de configuração ou {@code null} se não encontrado
				 * @since 1.0
				 */
				public String get(String chave) {
				  String valor = configs.get(chave);
				  if(valor == null && source != null) {
				  valor = source.pesquisar(chave);
				  if(valor != null) {
				  configs.put(chave, valor); // Corrigido: usar 'chave' em vez de string literal
				  }
				  }
				  return valor;
				}
				/**
				 * Define a fonte de configuração para recuperar valores.
				 *
				 * @param source a ConfigSource para usar na recuperação de valores de configuração
				 * @since 1.0
				 */
				public void setSource(ConfigSource source) {
				  this.source = source;
				}
				/**
				 * Limpa todos os valores de configuração em cache.
				 *
				 * @since 1.0
				 */
				public void clear() {
				  configs.clear();
				}

ID	Documentation A	Documentation B	Documentation C	Documentation 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 {
	
	  response.sendError(HttpServletResponse.SC_UNAUTHORIZED, ACESSO_NEGADO);
	
	  }
	
	}
	
	// Fim do arquivo: Http401UnauthorizedEntryPoint.java
		/**
		 * 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_UNAUTHORIZED, 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
			 */
			@Override
			public void commence(HttpServletRequest request, HttpServletResponse response,
			  AuthenticationException authenticationException) throws IOException, ServletException {
			
			response.sendError(HttpServletResponse.SC_UNAUTHORIZED, ACESSO_NEGADO);
			}
				/**
				 * 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
				 */
				
				--- 	
				
				/**
				 * Manipula ponto de entrada de autenticação enviando resposta HTTP 401 Unauthorized.
				 *
				 * @param request a requisição HTTP servlet
				 * @param response a resposta HTTP servlet
				 * @param authenticationException a exceção de autenticação que acionou este ponto de entrada
				 * @throws IOException se ocorrer exceção de entrada ou saída
				 * @throws ServletException se ocorrer exceção específica de servlet
				 * @since 1.0
				 */
				@Override
				public void commence(HttpServletRequest request, HttpServletResponse response,
				  AuthenticationException authenticationException) throws IOException, ServletException {
				
				response.sendError(HttpServletResponse.SC_UNAUTHORIZED, ACESSO_NEGADO);
				}

ID	Documentation A	Documentation B	Documentation C	Documentation D

6	// Arquivo: CustomUserDTO.java
	// Caminho:
	\br\gov\ana\sarsb\config\security\dto\CustomUserDTO.java
	
	package br.gov.ana.sarsb.config.security.dto;
	import java.util.Collection;
	import org.springframework.security.core.GrantedAuthority;
	import org.springframework.security.core.userdetails.User;
	import lombok.EqualsAndHashCode;
	import lombok.Getter;
	import lombok.Setter;
	
	@EqualsAndHashCode(callSuper = false)
	@Getter
	@Setter
	public class CustomUserDTO extends User {
	private static final long serialVersionUID = 1L;
	private String fullName;
	private String givenName;
	private String lastName;
	private String title;
	private String department;
	private String mail;
	/**
	* Constructs a CustomUserDTO with basic authentication information.
	*
	* @param username the username identifying the user whose data is represented by this UserDetails instance
	* @param password the password used to authenticate the user
	* @param authorities the authorities granted to the user
	*/
	public CustomUserDTO(String username, String password, Collection<? extends GrantedAuthority> authorities) {
	super(username, password, authorities);
	}
	/**
	* Constructs a CustomUserDTO with authentication information and additional user details.
	*
	* @param username the username identifying the user whose data is represented by this UserDetails instance
	* @param password the password used to authenticate the user
	* @param fullName the full name of the user
	* @param mail the email address of the user
	* @param authorities the authorities granted to the user
	*/
	public CustomUserDTO(String username, String password, String fullName, String mail, Collection<? extends GrantedAuthority> authorities) {
	super(username, password, authorities);
	this.fullName = fullName;
	this.mail = mail;
	}
	}
	// Fim do arquivo: CustomUserDTO.java
		/**
		* Constructs a CustomUserDTO with basic authentication information.
		*
		* @param username the username identifying the user whose data is represented by this UserDetails instance
		* @param password the password used to authenticate the user
		* @param authorities the authorities granted to the user
		*/
		public CustomUserDTO(String username, String password, Collection<? extends GrantedAuthority> authorities) {
		super(username, password, authorities);
		}
		
		/**
		* Constructs a CustomUserDTO with authentication information and additional user details.
		*
		* @param username the username identifying the user whose data is represented by this UserDetails instance
		* @param password the password used to authenticate the user
		* @param fullName the full name of the user
		* @param mail the email address of the user
		* @param authorities the authorities granted to the user
		*/
		public CustomUserDTO(String username, String password, String fullName, String mail, Collection<? extends GrantedAuthority> authorities) {
		super(username, password, authorities);
		this.fullName = fullName;
		this.mail = mail;
		}
			/**
			* [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.
			*/
			
			--- 	
			
			/**
			* Constructs a CustomUserDTO with basic authentication information.
			*
			* @param username the username identifying the user whose data is represented by this UserDetails instance
			* @param password the password used to authenticate the user
			* @param authorities the authorities granted to the user
			*/
			public CustomUserDTO(String username, String password, Collection<? extends GrantedAuthority> authorities) {
			super(username, password, authorities);
			}
			
			/**
			* Constructs a CustomUserDTO with authentication information and additional user details.
			*
			* @param username the username identifying the user whose data is represented by this UserDetails instance
			* @param password the password used to authenticate the user
			* @param fullName the full name of the user
			* @param mail the email address of the user
			* @param authorities the authorities granted to the user
			*/
			public CustomUserDTO(String username, String password, String fullName, String mail, Collection<? extends GrantedAuthority> authorities) {
			super(username, password, authorities);
			this.fullName = fullName;
			this.mail = mail;
			}
				/**
				* DTO personalizado para detalhes do usuário autenticado.
				* Estende UserDetails do Spring Security com informações adicionais do usuário.
				*
				* @author Sistema SARSB
				* @version 1.0
				* @since 1.0
				*/
				
				--- 	
				
				/**
				* Constrói um CustomUserDTO com informações básicas de autenticação.
				*
				* @param username o username identificando o usuário cujos dados são representados por esta instância UserDetails
				* @param password a senha usada para autenticar o usuário
				* @param authorities as autoridades concedidas ao usuário
				* @since 1.0
				*/
				public CustomUserDTO(String username, String password, Collection<? extends GrantedAuthority> authorities) {
				super(username, password, authorities);
				}
				
				/**
				* Constrói um CustomUserDTO com informações de autenticação e detalhes adicionais do usuário.
				*
				* @param username o username identificando o usuário cujos dados são representados por esta instância UserDetails
				* @param password a senha usada para autenticar o usuário
				* @param fullName o nome completo do usuário
				* @param mail o endereço de email do usuário
				* @param authorities as autoridades concedidas ao usuário
				* @since 1.0
				*/
				public CustomUserDTO(String username, String password, String fullName, String mail, Collection<? extends GrantedAuthority> authorities) {
				super(username, password, authorities);
				this.fullName = fullName;
				this.mail = mail;
				}

ID	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
		/**
		 * 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(AcoesDocContratoMatrizEnum attribute) {
		  if (attribute == null) {
		  return 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 null;
		  }
		  return AcoesDocContratoMatrizEnum.get(dbData);
		}
			/**
			 * 🔍 [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.
			 */
			
			--- 	
			
			/**
			 * 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(AcoesDocContratoMatrizEnum attribute) {
			  if (attribute == null) {
			  return 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 null;
			  }
			  return AcoesDocContratoMatrizEnum.get(dbData);
			}
				/**
				 * Conversor JPA para enum AcoesDocContratoMatrizEnum.
				 * Converte entre valores de enum e representação de banco de dados.
				 *
				 * @author Sistema SARSB
				 * @version 1.0
				 * @since 1.0
				 */
				
				--- 	
				
				/**
				 * Converte atributo enum para valor de coluna do banco de dados.
				 *
				 * @param attribute o atributo enum para converter
				 * @return o código do enum como Long, ou {@code null} se attribute for {@code null}
				 * @since 1.0
				 */
				@Override
				public Long convertToDatabaseColumn(AcoesDocContratoMatrizEnum attribute) {
				  if (attribute == null) {
				  return null;
				  }
				  return attribute.getCodigo();
				}
				
				/**
				 * Converte valor de coluna do banco de dados para atributo enum.
				 *
				 * @param dbData o valor do banco de dados para converter
				 * @return o valor enum correspondente, ou {@code null} se dbData for {@code null}
				 * @since 1.0
				 */
				@Override
				public AcoesDocContratoMatrizEnum convertToEntityAttribute(Long dbData) {
				  if (dbData == null) {
				  return null;
				  }
				  return AcoesDocContratoMatrizEnum.get(dbData);
				}

ID	Documentation A	Documentation B	Documentation C	Documentation D

8	// Arquivo: FormularioAditivoDTO.java
	// Caminho:
	\br\gov\ana\sarsb\model\dto\agenciareguladora\aditivo\FormularioAditivoDTO.java
	package br.gov.ana.sarsb.model.dto.agenciareguladora.aditivo;
	import java.util.Date;
	import br.gov.ana.sarsb.model.entity.FormularioAditivoAgencia;
	import lombok.AllArgsConstructor;
	import lombok.Builder;
	import lombok.Getter;
	import lombok.NoArgsConstructor;
	import lombok.Setter;
	@Builder
	@AllArgsConstructor
	@NoArgsConstructor
	@Getter
	@Setter
	public class FormularioAditivoDTO {
	private Long id;
	private Long numeroRecibo;
	private String cnpjAgenciaReguladora;
	private Date dataEnvio;
	public FormularioAditivoDTO
	toFormularioAditivoDTO(FormularioAditivoAgencia entity) {
	return FormularioAditivoDTO.builder()
	.id(entity.getId())
	.numeroRecibo(entity.getNumeroRecibo())
	.cnpjAgenciaReguladora(entity.getAgenciaReguladora().getCnpj())
	.dataEnvio(entity.getDataEnvio())
	.build();
	}
	}
	// Fim do arquivo: FormularioAditivoDTO.java
	// Arquivo: FormularioSmsruDTO.java
	// Caminho:
	\br\gov\ana\sarsb\model\dto\agenciareguladora\aditivo\FormularioSmsruDTO.java
	package br.gov.ana.sarsb.model.dto.agenciareguladora.aditivo;
	import br.gov.ana.sarsb.model.entity.FormularioAditivoAgencia;
	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
		/**
		* Converts a FormularioAditivoAgencia entity to
		FormularioAditivoDTO.
		*
		* @param entity the FormularioAditivoAgencia entity to
		convert
		* @return a new FormularioAditivoDTO with data from the
		entity
		*/
		public FormularioAditivoDTO
		toFormularioAditivoDTO(FormularioAditivoAgencia entity)
		{
		return FormularioAditivoDTO.builder()
		.id(entity.getId())
		.numeroRecibo(entity.getNumeroRecibo())
		
		.cnpjAgenciaReguladora(entity.getAgenciaReguladora().g
		etCnpj())
		.dataEnvio(entity.getDataEnvio())
		.build();
		}
			/**
			* [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.
			*/
			
			---------- 	
			--- 	
			
			/**
			* Converts a FormularioAditivoAgencia entity
			to FormularioAditivoDTO.
			*
			* @param entity the
			FormularioAditivoAgencia entity to convert
			* @return a new FormularioAditivoDTO with
			data from the entity
			*/
			public FormularioAditivoDTO
			toFormularioAditivoDTO(FormularioAditivoAg
			encia entity) {
			return FormularioAditivoDTO.builder()
			.id(entity.getId())
			
			.numeroRecibo(entity.getNumeroRecibo())
			
			.cnpjAgenciaReguladora(entity.getAgenciaRe
			guladora().getCnpj())
			.dataEnvio(entity.getDataEnvio())
			.build();
			}
				/**
				* DTO para formulários de aditivo de agência reguladora.
				* Fornece conversão entre entidade e DTO.
				*
				* @author Sistema SARSB
				* @version 1.0
				* @since 1.0
				*/
				
				---------- 	
				
				/**
				* Converte uma entidade FormularioAditivoAgencia para
				FormularioAditivoDTO.
				*
				* @param entity a entidade FormularioAditivoAgencia para
				converter
				* @return um novo FormularioAditivoDTO com dados da
				entidade
				* @throws IllegalArgumentException se entity for {@code null}
				* @since 1.0
				*/
				public FormularioAditivoDTO
				toFormularioAditivoDTO(FormularioAditivoAgencia entity) {
				if (entity == null) {
				throw new
				IllegalArgumentException("FormularioAditivoAgencia não pode
				ser null");
				}
				
				return FormularioAditivoDTO.builder()
				.id(entity.getId())
				.numeroRecibo(entity.getNumeroRecibo())
				
				.cnpjAgenciaReguladora(entity.getAgenciaReguladora().getCnpj())
				.dataEnvio(entity.getDataEnvio())
				.build();
				}

ID	Documentation A	Documentation B	Documentation C	Documentation D

9	// Arquivo: InstrumentoCobrancaTemEtapasDTO.java
	// Caminho:
	\br\gov\ana\sarsb\model\dto\municipio\InstrumentoCobrancaTemEtapasDTO.java
	
	package br.gov.ana.sarsb.model.dto.municipio;
	
	import br.gov.ana.sarsb.model.entity.TipoEtapaServico;
	import lombok.AllArgsConstructor;
	import lombok.Builder;
	import lombok.Getter;
	import lombok.NoArgsConstructor;
	import lombok.Setter;
	
	@Builder
	@AllArgsConstructor
	@NoArgsConstructor
	@Getter
	@Setter
	public class InstrumentoCobrancaTemEtapasDTO {
	
	private TipoEtapaServico tipoEtapaServico;
	
	public String carregaDetalhamento() {
	return tipoEtapaServico.getDescricao();
	}
	
	}
	
	// Fim do arquivo: InstrumentoCobrancaTemEtapasDTO.java
		/**
		* Loads the detailed description of the service stage.
		*
		* @return the description of the service stage type
		*/
		public String carregaDetalhamento() {
		return tipoEtapaServico.getDescricao();
		}
			/**
			* [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.
			*/
			
			---------- 	
			--- 	
			
			/**
			* Loads the detailed description of the service stage.
			*
			* @return the description of the service stage type
			*/
			public String carregaDetalhamento() {
			return tipoEtapaServico.getDescricao();
			}
				/**
				* DTO para etapas de instrumentos de cobrança.
				* Fornece detalhamento das etapas de serviço.
				*
				* @author Sistema SARSB
				* @version 1.0
				* @since 1.0
				*/
				
				---------- 	
				
				/**
				* Carrega a descrição detalhada da etapa de serviço.
				*
				* @return a descrição do tipo de etapa de serviço, ou string vazia se tipoEtapaServico for {@code null}
				* @since 1.0
				*/
				public String carregaDetalhamento() {
				return tipoEtapaServico != null ? tipoEtapaServico.getDescricao() : "";
				}

ID	Documentation A	Documentation B	Documentation C	Documentation D

10	// Arquivo: SecurityResource.java
	// Caminho: \br\gov\ana\sarsb\config\security\boundary\SecurityResource.java
	package br.gov.ana.sarsb.config.security.boundary;
	import java.util.Collection;
	import org.springframework.security.access.prepost.PreAuthorize;
	import org.springframework.security.core.Authentication;
	import org.springframework.security.core.GrantedAuthority;
	import org.springframework.web.bind.annotation.GetMapping;
	import org.springframework.web.bind.annotation.RequestMapping;
	import org.springframework.web.bind.annotation.ResponseBody;
	import org.springframework.web.bind.annotation.RestController;
	@RestController
	@RequestMapping(path = "/security")
	@PreAuthorize("isAuthenticated()")
	public class SecurityResource {
	/**
	* Retrieves the collection of granted authorities (profiles/roles) for the currently authenticated user.
	* This endpoint returns the security authorities associated with the authenticated user's session.
	*
	* @param authentication the authentication object containing user credentials and authorities
	* @return a collection of granted authorities representing the user's profiles/roles
	*/
	@GetMapping(value = "/currentuser/profiles")
	@ResponseBody
	public Collection<? extends GrantedAuthority> profiles (Authentication authentication) {
	return authentication.getAuthorities();
	}
	}
	// Fim do arquivo: SecurityResource.java
	// Arquivo: UserResource.java
	// Caminho: \br\gov\ana\sarsb\config\security\boundary\UserResource.java
	package br.gov.ana.sarsb.config.security.boundary;
	import org.springframework.http.MediaType;
	import org.springframework.security.access.prepost.PreAuthorize;
	import org.springframework.security.core.Authentication;
	import org.springframework.security.core.userdetails.UserDetails;
	import org.springframework.web.bind.annotation.GetMapping;
	import org.springframework.web.bind.annotation.RequestMapping;
	import org.springframework.web.bind.annotation.RestController;
	import br.gov.ana.sarsb.config.security.dto.CustomUserDTO;
	import br.gov.ana.sarsb.config.security.dto.UserInfoDTO;
	@RestController
	@RequestMapping(path = "/userinfo")
	@PreAuthorize("isAuthenticated()")
	public class UserResource {
	
	@GetMapping(produces = MediaType.APPLICATION_JSON_VALUE)
	public UserInfoDTO getUserInfo(Authentication authentication) {
	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;
	}
	}
	// Fim do arquivo: UserResource.java
		/**
		* Retrieves user information from the authenticated user's session.
		*
		* @param authentication the authentication object containing user details
		* @return UserInfoDTO with user's name, email, and username, or null if user is not a CustomUserDTO
		*/
		@GetMapping(produces = MediaType.APPLICATION_JSON_VALUE)
		public UserInfoDTO getUserInfo(Authentication authentication) {
		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;
		}
			/**
			* [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.
			*/
			
			---------- 	
			
			/**
			* Retrieves user information from the authenticated user's session.
			*
			* @param authentication the authentication object containing user details
			* @return UserInfoDTO with user's name, email, and username, or null if user is not a CustomUserDTO
			*/
			@GetMapping(produces = MediaType.APPLICATION_JSON_VALUE)
			public UserInfoDTO getUserInfo(Authentication authentication) {
			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;
			}
				/**
				* Recurso REST para informações do usuário autenticado.
				* Fornece endpoints para consulta de dados do usuário logado.
				*
				* @author Sistema SARSB
				* @version 1.0
				* @since 1.0
				*/
				
				---------- 	
				
				/**
				* Recupera informações do usuário a partir da sessão do usuário autenticado.
				*
				* @param authentication o objeto de autenticação contendo detalhes do usuário
				* @return UserInfoDTO com nome, email e username do usuário, ou {@code null} se o usuário não for CustomUserDTO
				* @since 1.0
				*/
				@GetMapping(produces = MediaType.APPLICATION_JSON_VALUE)
				public UserInfoDTO getUserInfo(Authentication authentication) {
				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;
				}

ID	Documentation A	Documentation - B

4	// Arquivo: AjaxLogoutSuccessHandler.java
	// Caminho: \br\gov\ana\sarsb\config\security\handler\AjaxLogoutSuccessHandler.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.apache.commons.lang3.StringUtils;
	import org.springframework.beans.factory.annotation.Autowired;
	import org.springframework.security.core.Authentication;
	import org.springframework.security.oauth2.provider.token.TokenStore;
	import org.springframework.security.web.authentication.AbstractAuthenticationTargetUrlRequestHandler;
	import org.springframework.security.web.authentication.logout.LogoutSuccessHandler;
	import org.springframework.stereotype.Component;
	
	@Component
	public class AjaxLogoutSuccessHandler extends AbstractAuthenticationTargetUrlRequestHandler
	implements LogoutSuccessHandler {
	
	public static final String BEARER_AUTHENTICATION = "Bearer ";
	
	private static final String HEADER_AUTHORIZATION = "authorization";
	
	@Autowired
	private TokenStore tokenStore;
	
	/**
	* 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(HttpServletRequest request, HttpServletResponse 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: AjaxLogoutSuccessHandler.java
		/**
		* 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(HttpServletRequest request, HttpServletResponse 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);
		}

ID	Documentation - C	Documentation - D

4	/**
	* [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 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(HttpServletRequest request, HttpServletResponse 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);
	}
		/**
		* Manipulador de sucesso de logout para requisições AJAX.
		* Gerencia invalidação de tokens OAuth2 durante o processo de logout.
		*
		* @author Sistema SARSB
		* @version 1.0
		* @since 1.0
		*/
		
		--- 	
		
		/**
		* Manipula logout bem-sucedido invalidando tokens OAuth2 e definindo status HTTP apropriado.
		*
		* Este método extrai o token Bearer do cabeçalho Authorization, remove tanto o token de acesso
		* quanto seu token de refresh associado do armazenamento de tokens, e retorna status HTTP 200 OK.
		*
		* @param request a requisição HTTP servlet contendo o cabeçalho de autorização com token Bearer
		* @param response a resposta HTTP servlet que será definida com status 200 OK
		* @param authentication o objeto de autenticação representando o usuário sendo deslogado
		* @throws IOException se ocorrer exceção de entrada ou saída durante processamento de request/response
		* @throws ServletException se ocorrer exceção específica de servlet durante processamento
		* @since 1.0
		*/
		@Override
		public void onLogoutSuccess(HttpServletRequest request, HttpServletResponse 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);
		}