```java
package br.com.padon.application.security;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.http.HttpMethod;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfiguration;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.config.annotation.web.configurers.AbstractHttpConfigurer;
import org.springframework.security.config.http.SessionCreationPolicy;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.SecurityFilterChain;
import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;
import org.springframework.web.cors.CorsUtils;


@Configuration
@EnableWebSecurity
public class SecurityService {

    @Autowired
    private SecurityFilter securityFilter;

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity httpSecurity) throws Exception {
        return httpSecurity
                .csrf(AbstractHttpConfigurer::disable)
                .sessionManagement(s -> s.
```

```
31 sessionCreationPolicy(SessionCreationPolicy.STATELESS))
32                 .authorizeHttpRequests(a -> a
33                         .requestMatchers(CorsUtils::
   isPreFlightRequest).permitAll()
34                         .requestMatchers(HttpMethod.POST,
   "/login").permitAll()
35                         .requestMatchers(HttpMethod.POST,
   "/funcionario/create").permitAll()
36                         .requestMatchers(HttpMethod.GET,
   "/loja/get").permitAll()
37                         .anyRequest().authenticated()
38                 )
39                 .addFilterBefore(securityFilter,
   UsernamePasswordAuthenticationFilter.class)
40                 .build();
41     }
42
43     @Bean
44     public AuthenticationManager authenticationManager(
   AuthenticationConfiguration auth) throws Exception {
45         return auth.getAuthenticationManager();
46     }
47
48     @Bean
49     public PasswordEncoder passwordEncoder() {
50         return new BCryptPasswordEncoder();
51     }
52 }
53
```