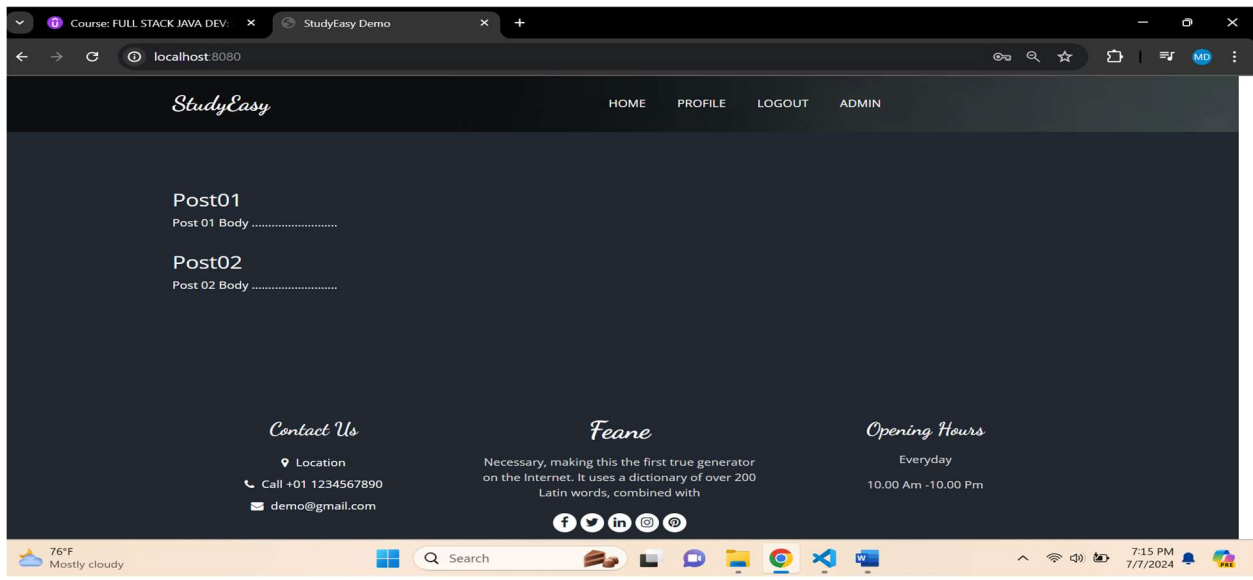
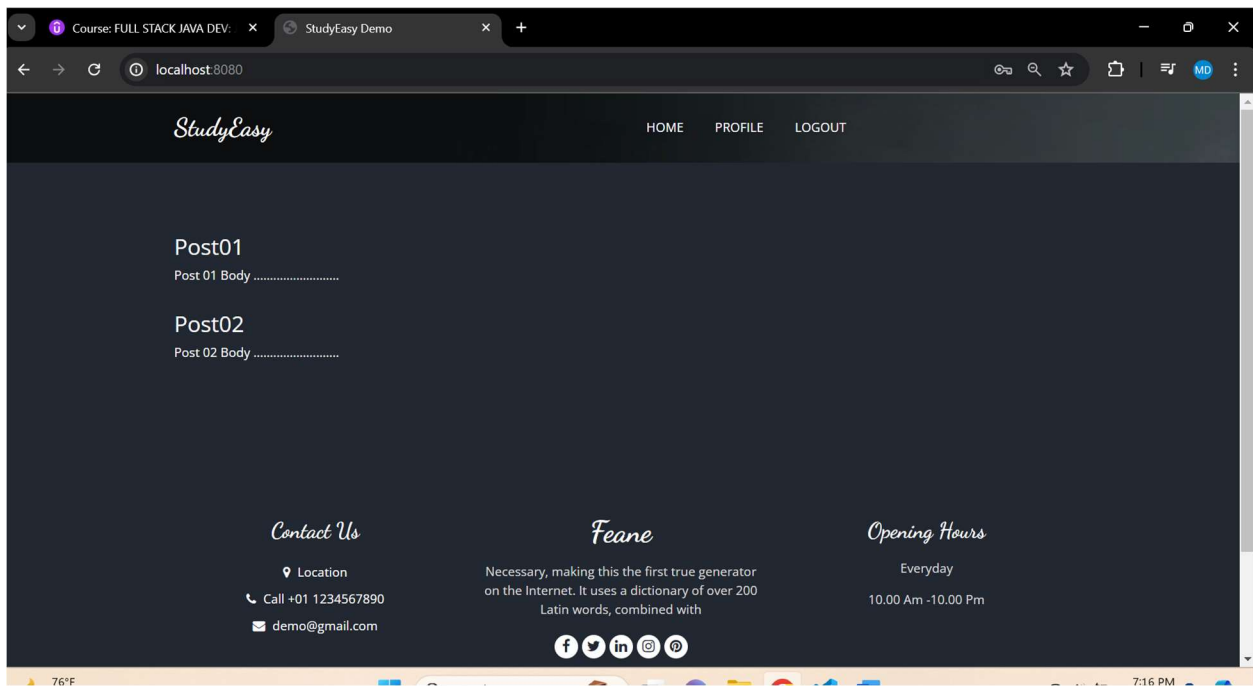


## SpringBoot - Application has bugs Add security rules for Roles and Authorities

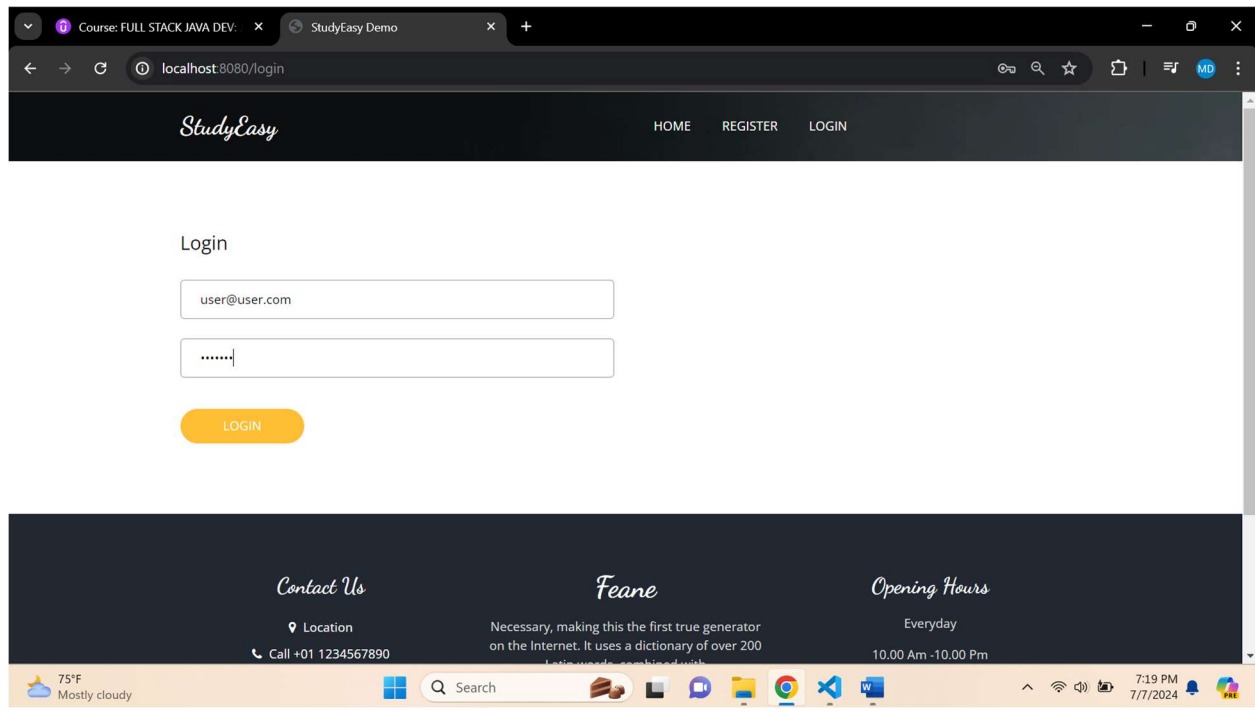
Super Editor can see admin but not editor.



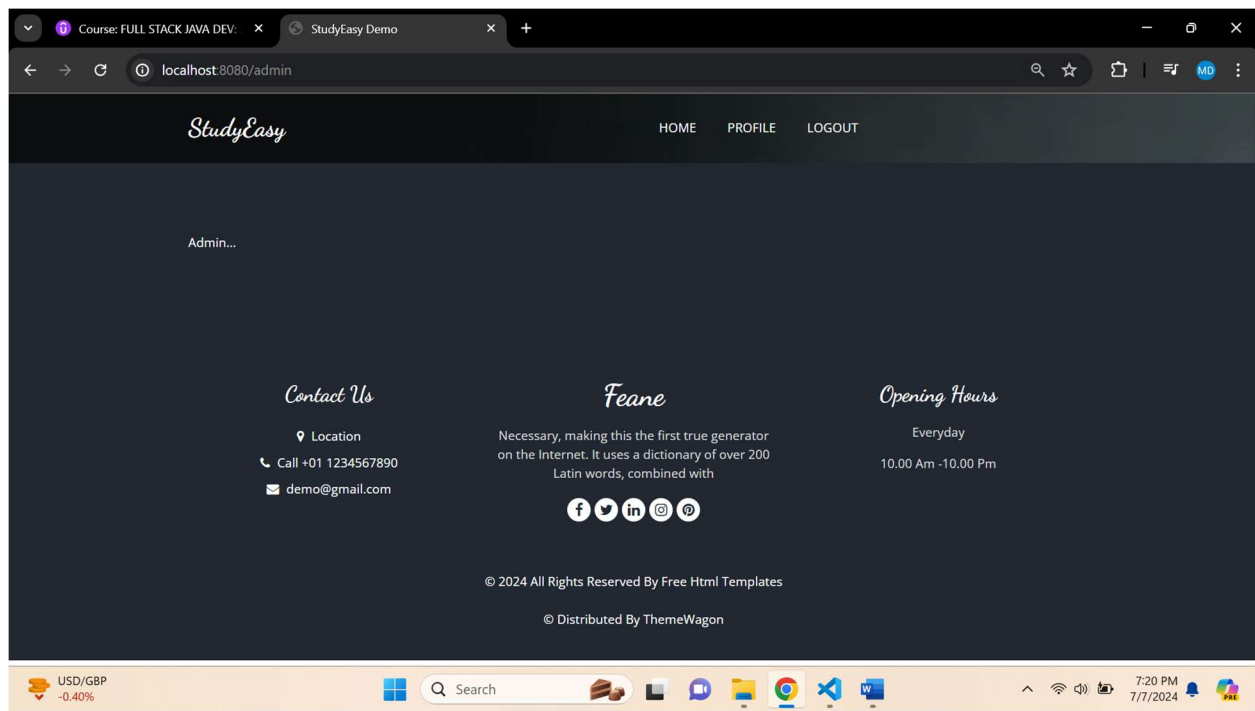
Editor cannot see admin.



## Login as User -> Then access Admin page by using URL



## You will be able to access Admin Page



## Add this below line in HomeController.java

```
@GetMapping("/editor")
public String editor(Model model){
    return "editor";
}
```

## editor.html

```
<!DOCTYPE html>
<html>

<head th:replace="/fragments/head :: head">

<body class="sub_page">

    <div class="hero_area">
        <div class="bg-box">
            
        </div>
        <!-- header section strats -->
        <header th:replace="/fragments/header :: header">
            <!-- end header section -->
        </div>

        <!-- about section -->

        <section class="about_section layout_padding">
            <div class="container">
                Editor...
            </div>
        </section>

        <!-- end about section -->

        <!-- footer section -->
        <footer th:replace="/fragments/footer :: footer">
            <!-- footer section -->
        </body>

</html>
```

## WebSecurityConfig.java

```
package org.studyeasy.SpringBlog.security;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.Customizer;
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecurity;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.SecurityFilterChain;

@Configuration
@EnableWebSecurity
@EnableMethodSecurity(prePostEnabled = true, securedEnabled = true)
public class WebSecurityConfig {
    private static final String[] WHITELIST = {
        "/",
        "/login",
        "/register",
        "/db-console/**",

        // Allow all the css and other resources
        "/css/**",
        "/fonts/**",
        "/images/**",
        "/js/**"
    };

    @Bean
    public static PasswordEncoder passwordEncoder() {
        return new BCryptPasswordEncoder();
    }

    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http) throws
Exception {
        http
            .authorizeHttpRequests(auth -> auth
```

```

        .requestMatchers(WHITELIST)
        .permitAll()
        .anyRequest()
        .authenticated()

        .formLogin(formLogin -> formLogin
            .loginPage("/login")
            .loginProcessingUrl("/login")
            .usernameParameter("email")
            .passwordParameter("password")
            .defaultSuccessUrl("/", true)
            .failureUrl("/login?error")
            .permitAll())

        .logout(logout -> logout
            .logoutUrl("/logout")
            .logoutSuccessUrl("/")
            .permitAll())

        .httpBasic(Customizer.withDefaults()); // Enable
HTTP Basic authentication

        // TODO: remove these files after upgrading the DB from h2 infile
DB
        http.csrf(csrf -> csrf.disable());
        // Disable frame options
        http.headers(headers -> headers.frameOptions(frameOptions ->
frameOptions.disable()));

        return http.build();
    }
}

```

## Make some modifications in WebSecurityConfig.java

### Change below line

```

http

        .authorizeHttpRequests(auth -> auth
            .requestMatchers(WHITELIST)
            .permitAll()
            .anyRequest()
            .authenticated())

```

## To

```
.authorizeHttpRequests(auth -> auth
    .requestMatchers(WHITELIST).permitAll()
    .requestMatchers("/profile/**").authenticated()
    .requestMatchers("/admin/**").hasRole("ADMIN")
    .requestMatchers("/editor/**").hasAnyRole("ADMIN", "EDITOR")
    .requestMatchers("/test").hasAuthority(Privileges.ACCESS_ADMIN_PANEL.getPrivilege()))
```

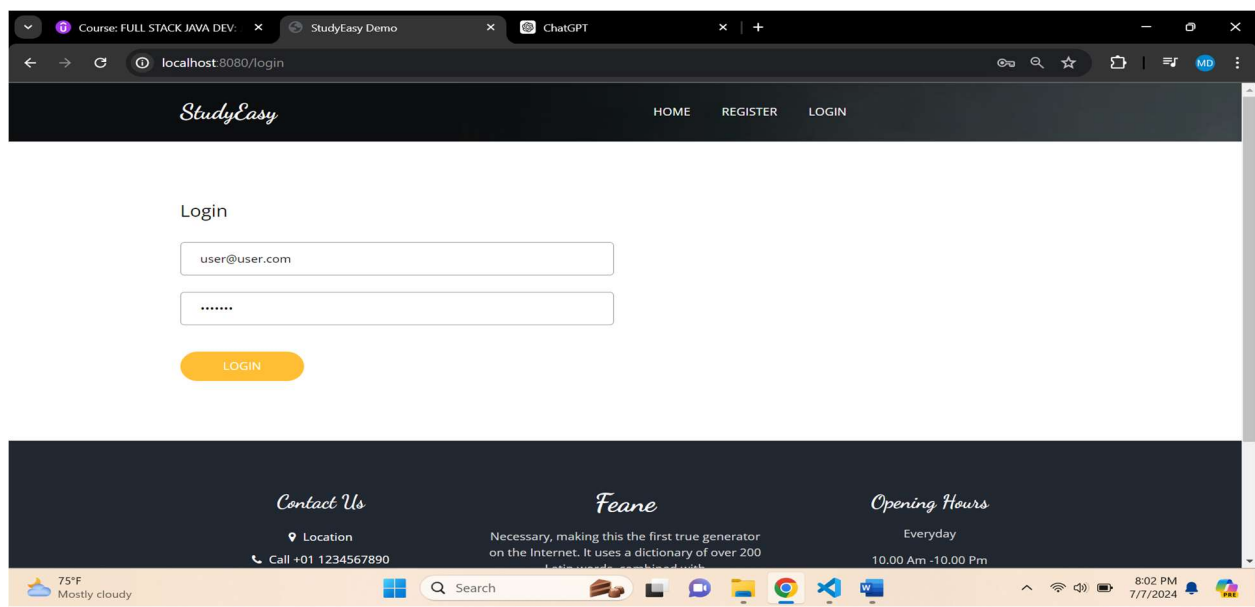
## In AccountController.java, add below line

```
@GetMapping("/profile")
public String profile(Model model) {
    return "profile";
}

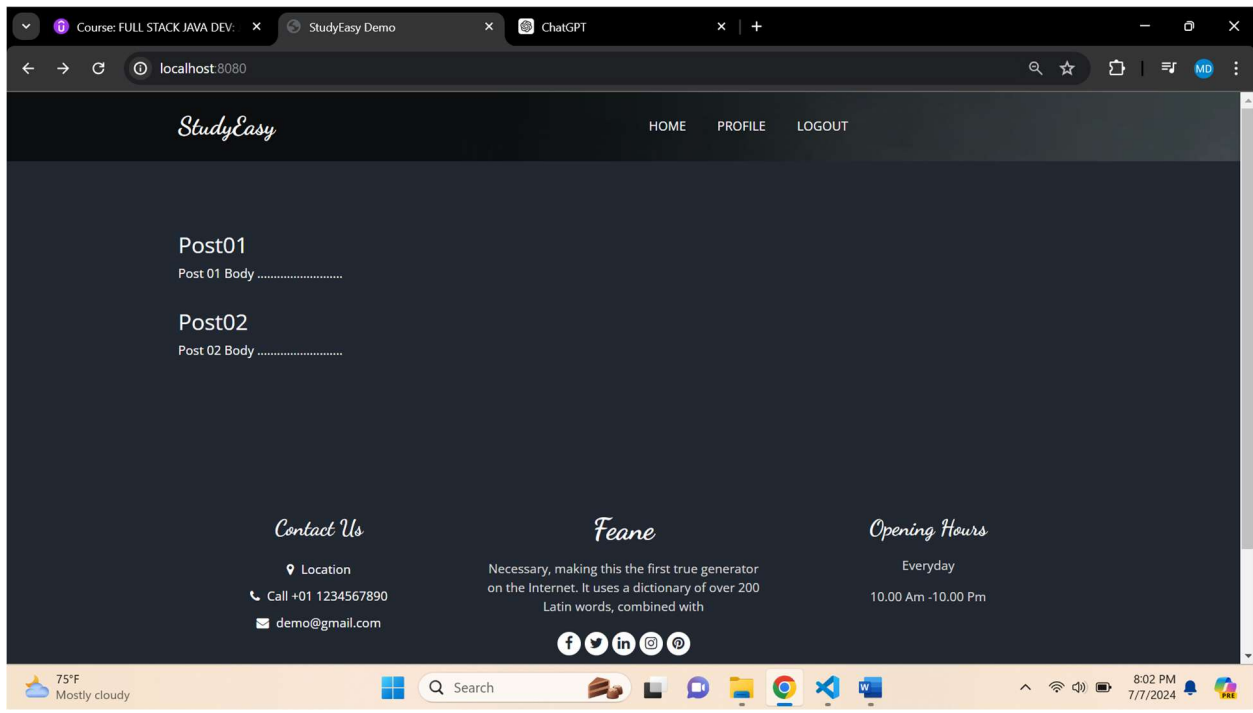
@GetMapping("/test")
public String test(Model model) {
    return "test";
}
```

## Output

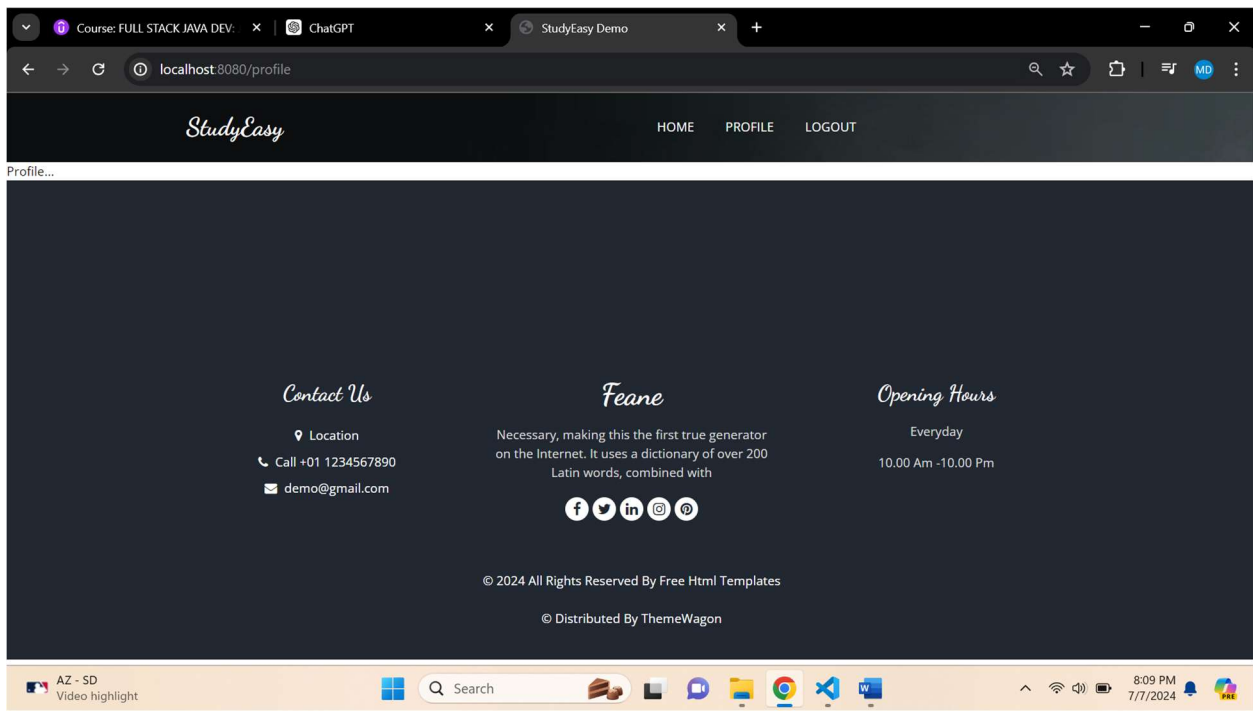
### Login as User



You get below page



Click on profile link -> You get below page



User -> Has access to profile

**Admin -> access to admin but editor access**

**Supereditor -> access to admin and editor**

**Editor -> no access to admin**

**hasManyRoles and hasManyAuthorities -> many roles and authorities.**