**SpringBoot\_Add Users with Roles and Authorities Continued**

**Updated SeedData.java**

package org.studyeasy.SpringBlog.config;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.stereotype.Component;

import org.studyeasy.SpringBlog.models.Account;

import org.studyeasy.SpringBlog.models.Authority;

import org.studyeasy.SpringBlog.models.Post;

import org.studyeasy.SpringBlog.services.AccountService;

import org.studyeasy.SpringBlog.services.AuthorityService;

import org.studyeasy.SpringBlog.services.PostService;

import org.studyeasy.SpringBlog.utils.constant.Privilages;

import org.studyeasy.SpringBlog.utils.constant.Roles;

@Component

public class SeedData implements CommandLineRunner {

    @Autowired

    private PostService postService;

    @Autowired

    private AccountService accountService;

    @Autowired

    private AuthorityService authorityService;

    @Override

    public void run(String... args) throws Exception {

        for (Privilages auth : Privilages.values()) {

            Authority authority = new Authority();

            authority.setId(auth.getId());

            authority.setName(auth.getPrivillage());

            authorityService.save(authority);

        }

        // Account

        Account account01 = new Account();

        Account account02 = new Account();

        Account account03 = new Account();

        Account account04 = new Account();

        account01.setEmail("user@user.com");

        account01.setPassword("pass987");

        account01.setFirstname("User");

        account01.setLastname("lastname");

        account02.setEmail("admin@admin.com");

        account02.setPassword("pass987");

        account02.setFirstname("Admin");

        account02.setLastname("lastname");

        account02.setRole(Roles.ADMIN.getRole());

        account03.setEmail("editor@editor.com");

        account03.setPassword("pass987");

        account03.setFirstname("Editor");

        account03.setLastname("lastname");

        account03.setRole(Roles.EDITOR.getRole());

        account04.setEmail("super\_editor@editor.com");

        account04.setPassword("pass987");

        account04.setFirstname("Editor");

        account04.setLastname("lastname");

        account04.setRole(Roles.EDITOR.getRole());

        Set<Authority> authorities = new HashSet<>();

        authorityService.findById(Privilages.RESET\_ANY\_USER\_PASSWORD.getId()).ifPresent(authorities::add);

        authorityService.findById(Privilages.ACCESS\_ADMIN\_PANEL.getId()).ifPresent(authorities::add);

        account04.setAuthorities(authorities);

        accountService.save(account01);

        accountService.save(account02);

        accountService.save(account03);

        accountService.save(account04);

        // Post

        List<Post> posts = postService.getAll();

        if (posts.size() == 0) {

            Post post01 = new Post();

            post01.setTitle("Post01");

            post01.setBody("Post 01 Body ..........................");

            // added

            post01.setAccount(account01);

            postService.save(post01);

            Post post02 = new Post();

            post02.setTitle("Post02");

            post02.setBody("Post 02 Body ..........................");

            // added

            post02.setAccount(account02);

            postService.save(post02);

        }

    }

}

**Updated AuthorityService.java**

package org.studyeasy.SpringBlog.services;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.studyeasy.SpringBlog.models.Authority;

import org.studyeasy.SpringBlog.repositories.AuthorityRepository;

@Service

public class AuthorityService {

    @Autowired

    private AuthorityRepository authorityRepository;

    public Authority save(Authority authority){

        return authorityRepository.save(authority);

    }

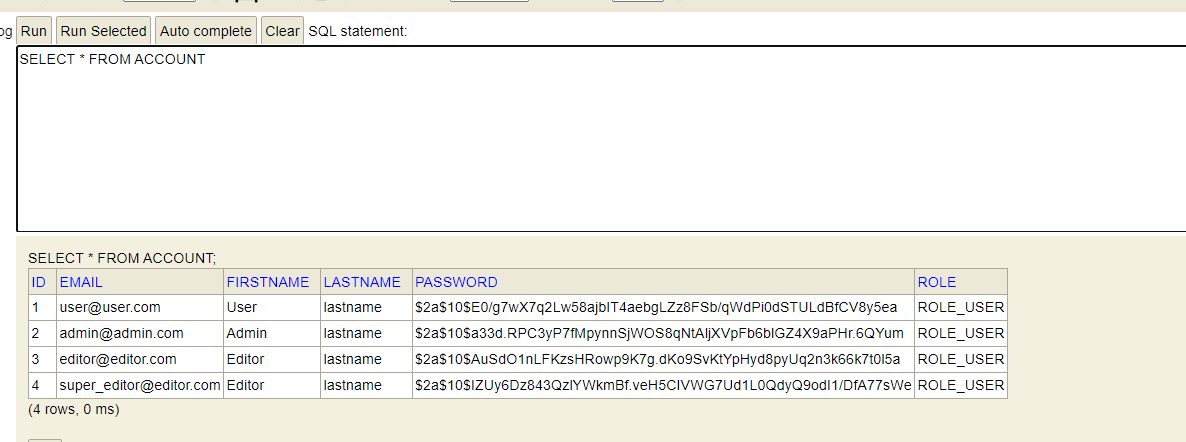
    public Optional<Authority> findById(Long id){

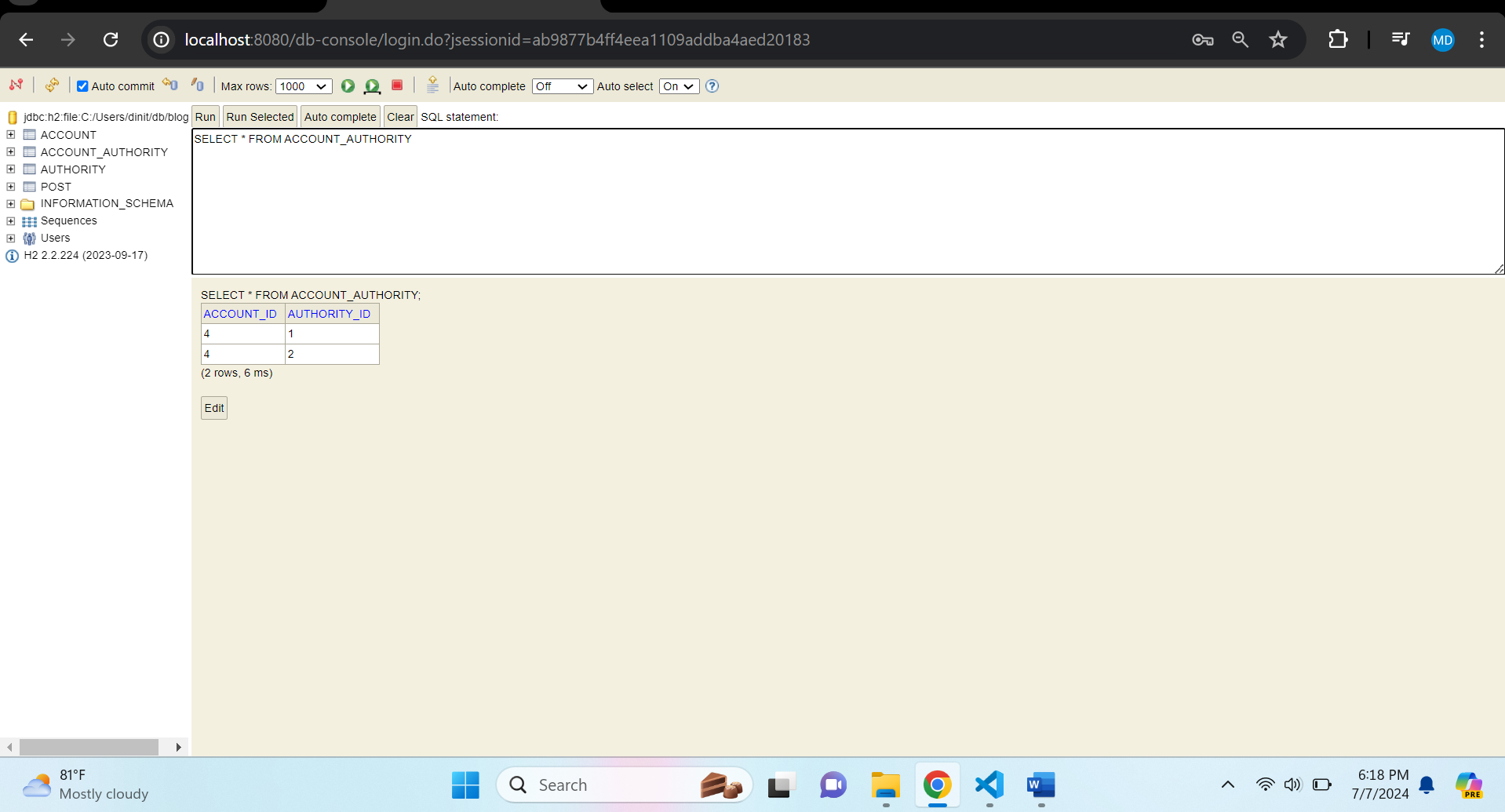
        return authorityRepository.findById(id);

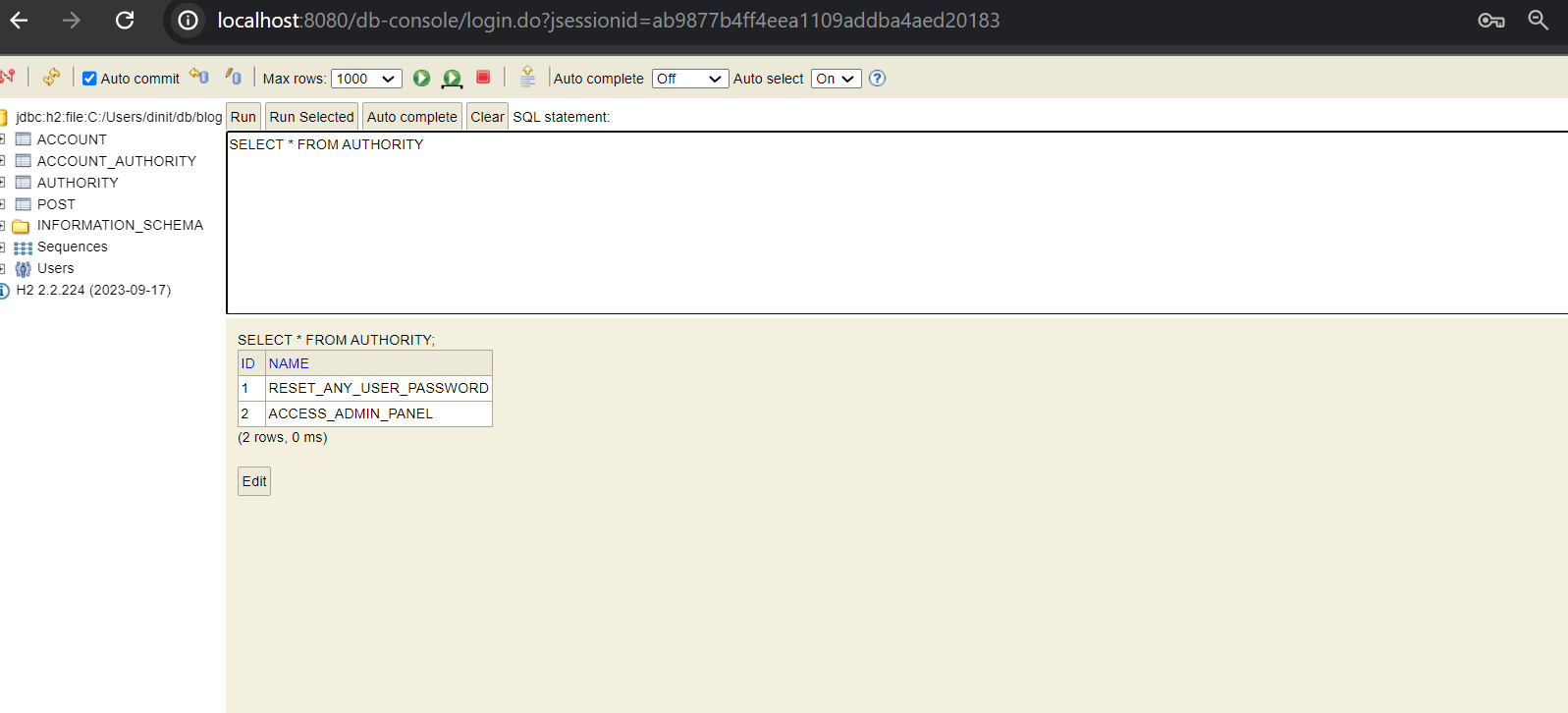
    }

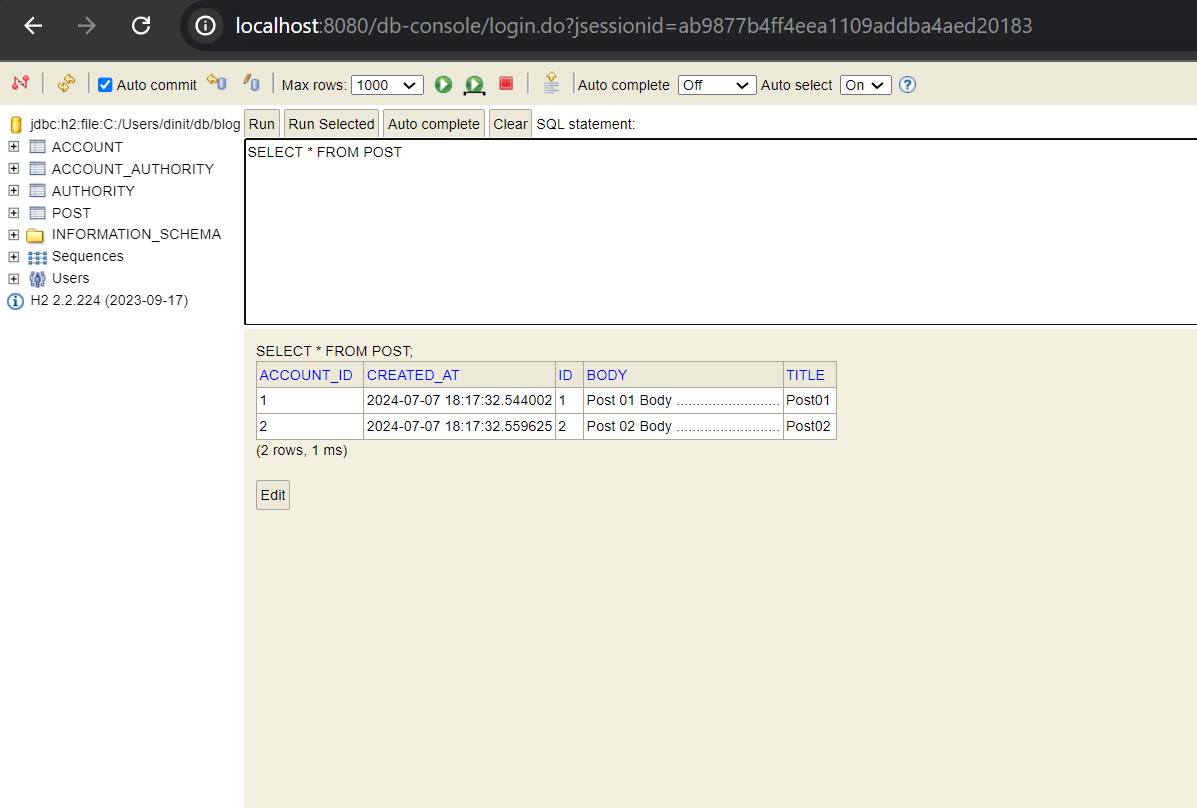
}}

**Output**

****

****

****

****

**Strings will make use of Lazy Loading. By default, it is not enabled.**

**application.properties -> Add below line**

#Enable Lazy Loading

spring.jpa.properties.hibernate.enable\_lazy\_load\_no\_trans = true

**AccountService.java**

for(Authority \_auth : account.getAuthorities()){

            grantedAuthority.add(new SimpleGrantedAuthority(\_auth.getName()));

        }

@Service

public class AccountService implements UserDetailsService{

    @Autowired

    private AccountRepository accountRepository;

    @Autowired

    private PasswordEncoder passwordEncoder;

    public Account save(Account account){

        account.setPassword(passwordEncoder.encode(account.getPassword()));

        account.setRole(Roles.USER.getRole());

        return accountRepository.save(account);

    }

    @Override

    public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException {

        Optional<Account> optionalAccount = accountRepository.findOneByEmailIgnoreCase(email);

        if(optionalAccount.isEmpty()){

            throw new UsernameNotFoundException("Account not found!..");

        }

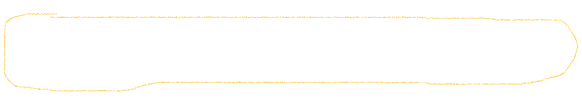
        Account account = optionalAccount.get();

        List<GrantedAuthority> grantedAuthority = new ArrayList<>();

        grantedAuthority.add(new SimpleGrantedAuthority(account.getRole()));



        for(Authority \_auth : account.getAuthorities()){

            grantedAuthority.add(new SimpleGrantedAuthority(\_auth.getName()));



        }

        return new User(account.getEmail(),account.getPassword(),grantedAuthority);

    }

}

**Added Line**

**In WebSecurityConfig.java, search below line**

.logoutSuccessUrl("/logout?success")

**And change to**

.logoutSuccessUrl("/")

**In header.html, add below line**

 <!-- Making Changes -->

                        <span sec:authorize="isAuthenticated()">

                            <li class="nav-item">

                                <a class="nav-link" href="/logout">Logout</a>

                            </li>

                        </span>

                        <span sec:authorize="hasRole('ROLE\_ADMIN')">

                            <li class="nav-item">

                                <a class="nav-link" href="/admin">Admin</a>

                            </li>

                        </span>

**Create new Controller -> AdminController.java**

@Controller

public class AdminController {

    @GetMapping("/admin")

    public String admin(Model model) {

        return "admin";

    }

}

**Create new file in templates -> admin.html**

<!DOCTYPE html>

<html>

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

<body class="sub\_page">

  <div class="hero\_area">

    <div class="bg-box">

      <img src="images/nav-bg.jpg" alt="">

    </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  ">

        Admin...

    </div>

  </section>

  <!-- end about section -->

  <!-- footer section -->

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

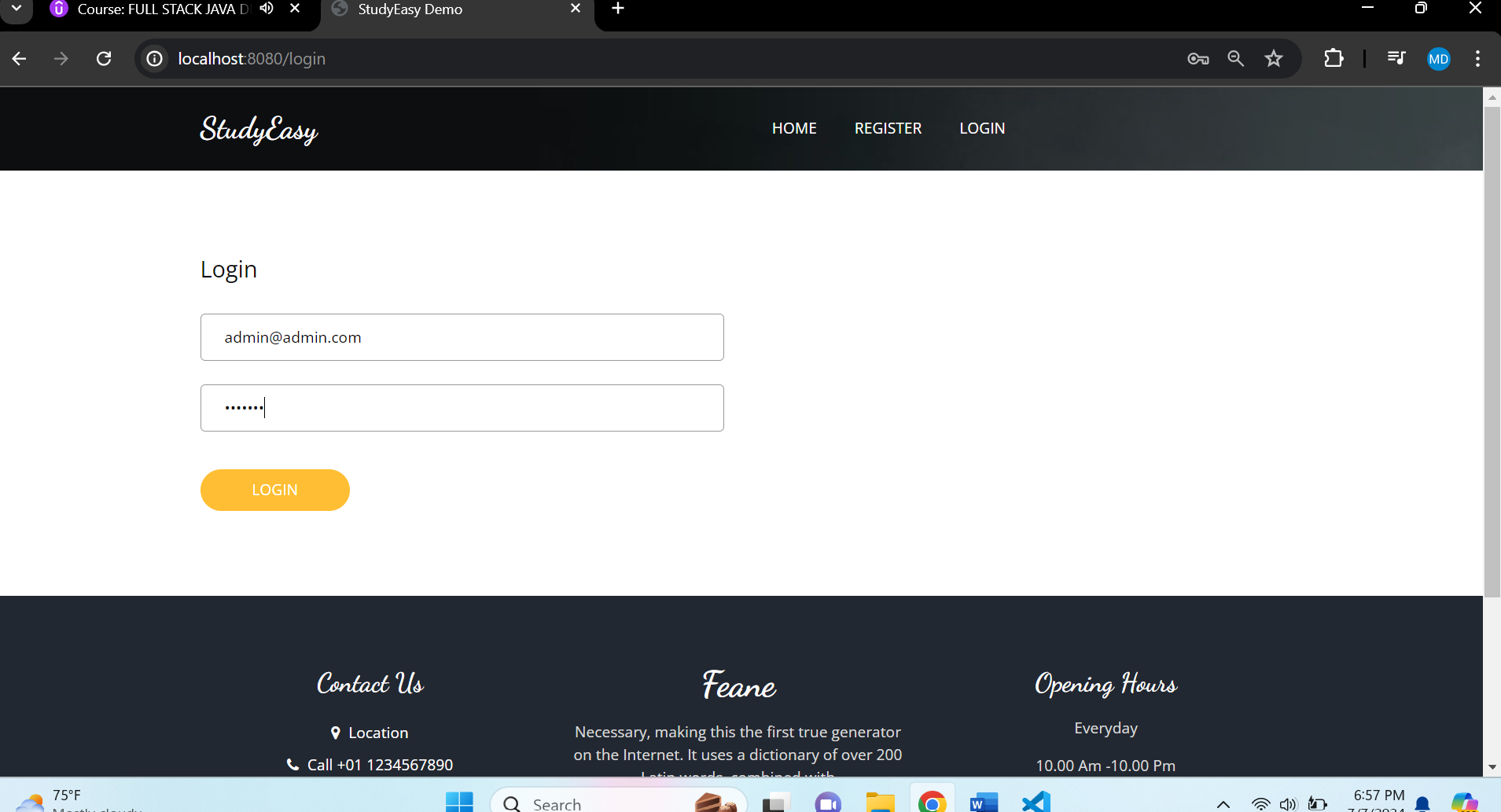
    <!-- footer section -->

</body>

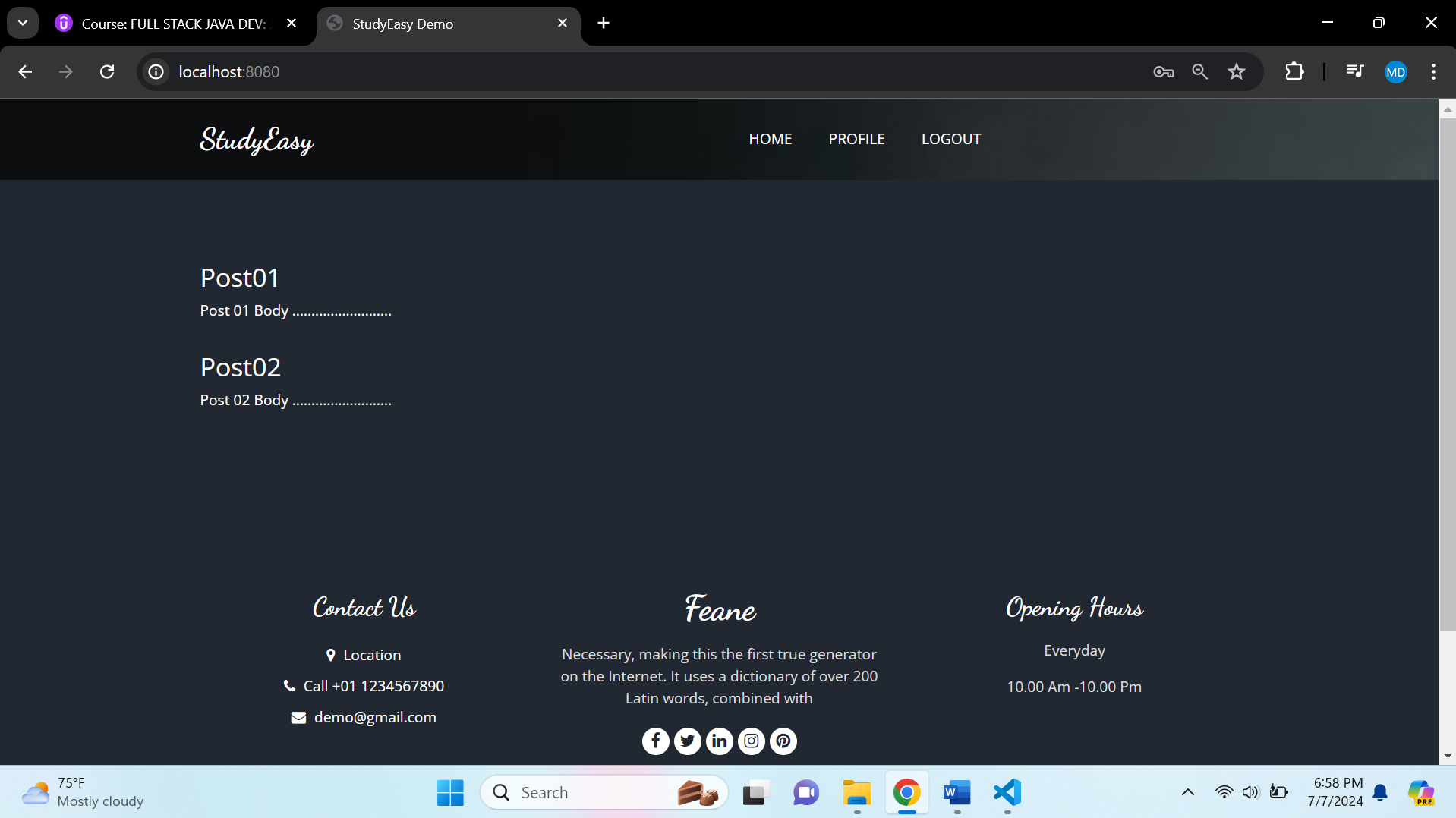
</html>

**Output:-**

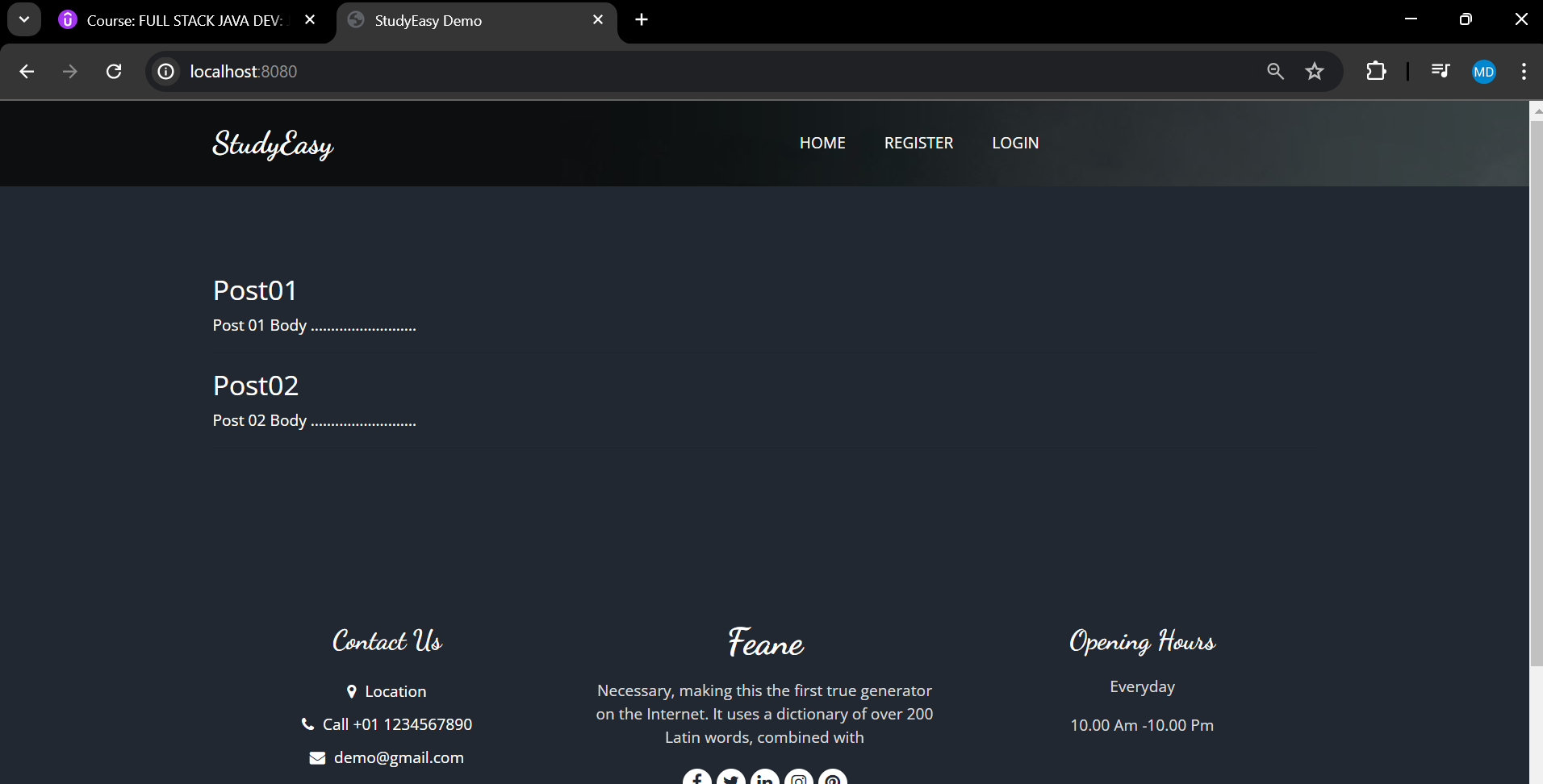
**Home Page -> Login as Admin**

****

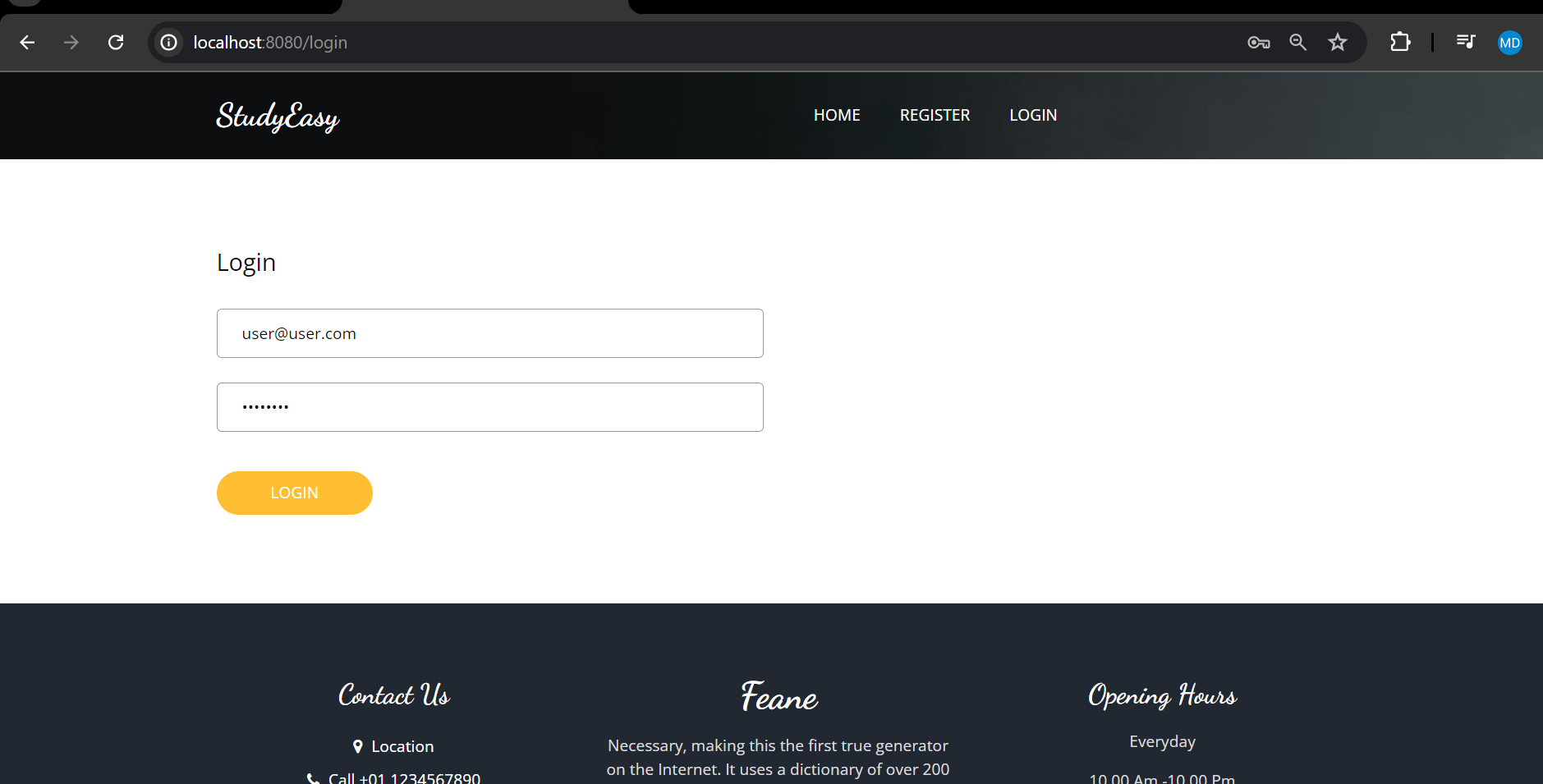
**After clicking Login Button, you get below page.**

****

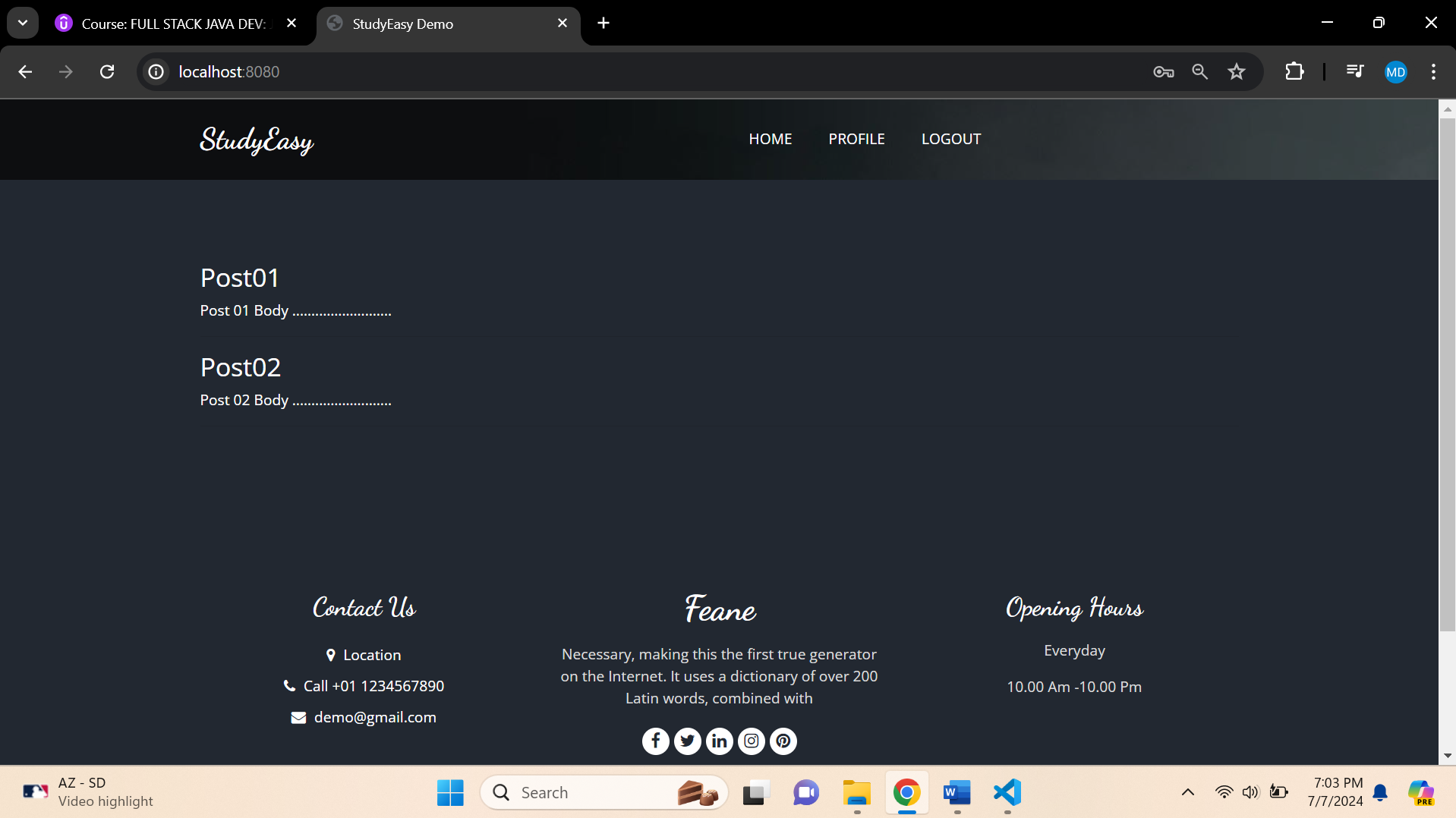
**Click Logout**

****

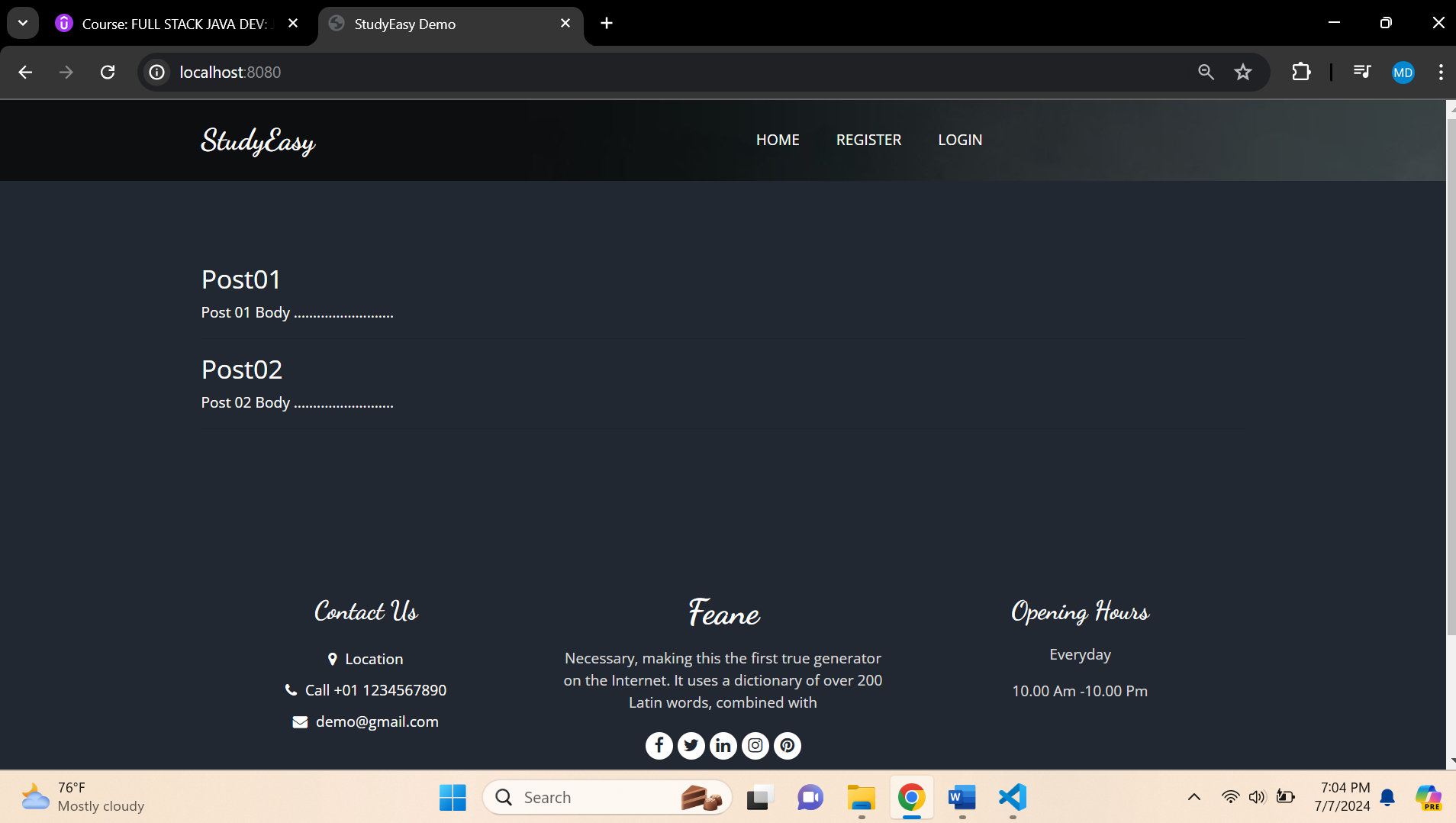
**Login Page -> Login as User**

****

**After clicking Login Button, you get below page.**

****

**Click Logout**

****