// IAuthRepository.cs

public interface IAuthRepository

{

bool ValidateCredentials(string username, string password);

}

// AuthRepository.cs

using System;

using System.DirectoryServices.AccountManagement;

public class AuthRepository : IAuthRepository

{

public bool ValidateCredentials(string username, string password)

{

bool isValid = false;

try

{

using (var context = new PrincipalContext(ContextType.Domain, "YOUR\_LDAP\_SERVER"))

{

isValid = context.ValidateCredentials(username, password);

}

}

catch (Exception ex)

{

// Handle any exceptions (e.g., LDAP server unreachable)

// Log the error

// Optionally, you might want to display a user-friendly error message

}

return isValid;

}

}

public class AccountController : Controller

{

private readonly IAuthRepository \_authRepository;

public AccountController(IAuthRepository authRepository)

{

\_authRepository = authRepository;

}

// Actions...

}

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult Login(LoginViewModel model)

{

if (ModelState.IsValid)

{

if (\_authRepository.ValidateCredentials(model.Username, model.Password))

{

// Authentication successful

// Redirect to home page or any other page

return RedirectToAction("Index", "Home");

}

else

{

ModelState.AddModelError("", "Invalid username or password.");

}

}

// If we got this far, something failed; redisplay form

return View(model);

}

public void ConfigureServices(IServiceCollection services)

{

services.AddTransient<IAuthRepository, AuthRepository>();

// Other configurations...

}

// LoginViewModel.cs

using System.ComponentModel.DataAnnotations;

public class LoginViewModel

{

[Required]

public string Username { get; set; }

[Required]

[DataType(DataType.Password)]

public string Password { get; set; }

}

<!-- Views/Account/Login.cshtml -->

@model LoginViewModel

<h2>Login</h2>

@using (Html.BeginForm("Login", "Account", FormMethod.Post))

{

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(m => m.Username)

@Html.TextBoxFor(m => m.Username, new { @class = "form-control" })

@Html.ValidationMessageFor(m => m.Username, "", new { @class = "text-danger" })

</div>

<div class="form-group">

@Html.LabelFor(m => m.Password)

@Html.PasswordFor(m => m.Password, new { @class = "form-control" })

@Html.ValidationMessageFor(m => m.Password, "", new { @class = "text-danger" })

</div>

<button type="submit" class="btn btn-primary">Login</button>

}

// Controllers/AccountController.cs

using Microsoft.AspNetCore.Mvc;

using System.DirectoryServices.AccountManagement;

public class AccountController : Controller

{

private readonly IAuthRepository \_authRepository;

public AccountController(IAuthRepository authRepository)

{

\_authRepository = authRepository;

}

// GET: /Account/Login

public IActionResult Login()

{

return View();

}

// POST: /Account/Login

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult Login(LoginViewModel model)

{

if (ModelState.IsValid)

{

if (\_authRepository.ValidateCredentials(model.Username, model.Password))

{

// Authentication successful

// Redirect to home page or any other page

return RedirectToAction("Index", "Home");

}

else

{

ModelState.AddModelError("", "Invalid username or password.");

}

}

// If we got this far, something failed; redisplay form

return View(model);

}

}