|  |
| --- |
| namespace AuthenticationAndAuthorization.Controllers  {  [ApiController]  [Route("api/[controller]")]  public class AuthController : ControllerBase  {  [HttpPost("login")]  public IActionResult Login([FromBody] LoginModel model)  {  if (IsValidUser(model.Username,model.Password))  {  var token = GenerateJwtToken(model.Username);  return Ok(new { Token = token });  }  return Unauthorized();  }  private bool IsValidUser(string username, string password)  {    return (username == "testuser" && password == "password123");  }  private string GenerateJwtToken(string username)  {  var claims = new[]  {  new Claim(ClaimTypes.Name, username)  };  var key = new  SymmetricSecurityKey(Encoding.UTF8.GetBytes("ThisIsAVeryStrongAndSecretKeyForYourJwtTokenGenerationThatShouldBeAtLeast32BytesLongAndRandomandthisisnot32bitslong"));  var creds = new SigningCredentials(key, SecurityAlgorithms.HmacSha256);  var token = new JwtSecurityToken(  issuer: "MyAuthServer",  audience: "MyApiUsers",  claims: claims,  expires: DateTime.UtcNow.AddMinutes(60),  signingCredentials: creds);  return new JwtSecurityTokenHandler().WriteToken(token);  }  }  } |

**MICROSERVICE-JWT Handson**

**Question 1: Implement JWT Authentication in ASP.NET Core Web API**

AuthController.cs

Program.cs

|  |
| --- |
| using Microsoft.IdentityModel.Tokens;  using System.Text;  var builder = WebApplication.CreateBuilder(args);  // Add services to the container.  builder.Services.AddControllers();  // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle  builder.Services.AddEndpointsApiExplorer();  builder.Services.AddSwaggerGen();  builder.Services.AddAuthentication("Bearer")  .AddJwtBearer("Bearer", options =>  {  options.TokenValidationParameters = new TokenValidationParameters  {  ValidateIssuer = true,  ValidateAudience = true,  ValidateLifetime = true,  ValidateIssuerSigningKey = true,  ValidIssuer = builder.Configuration["Jwt:Issuer"],  ValidAudience = builder.Configuration["Jwt:Audience"],  IssuerSigningKey = new  SymmetricSecurityKey(Encoding.UTF8.GetBytes(builder.Configuration["Jwt:Key"]))  };  });  builder.Services.AddAuthorization();  var app = builder.Build();  // Configure the HTTP request pipeline.  if (app.Environment.IsDevelopment())  {  app.UseSwagger();  app.UseSwaggerUI();  }  app.UseHttpsRedirection();  app.UseAuthentication();  app.UseAuthorization();  app.MapControllers();  app.Run(); |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Question 2: Secure an API Endpoint Using JWT**

ProtectedController.cs

|  |
| --- |
| namespace AuthenticationAndAuthorization.Controllers  {  [Route("api/[controller]")]  [ApiController]  public class ProtectedController : ControllerBase  {  [HttpGet("data")]  [Authorize]  public IActionResult GetProtectedData()  {  var username = User.Identity?.Name;  return Ok($"Hello, {username}! You have accessed protected data.");  }    }  } |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Question 3: Add Role-Based Authorization**

**ProtectedController.cs**

|  |
| --- |
| namespace AuthenticationAndAuthorization.Controllers  {  [Route("api/[controller]")]  [ApiController]  public class ProtectedController : ControllerBase  {  [HttpGet("data")]  [Authorize]  public IActionResult GetProtectedData()  {  var username = User.Identity?.Name;  return Ok($"Hello, {username}! You have accessed protected data.");  }  [HttpGet("admin-only")]  [Authorize(Roles = "Admin")]  public IActionResult GetAdminOnlyData()  {  var adminUsername = User.Identity?.Name;  return Ok($"Welcome, {adminUsername}! This is highly confidential data accessible only by Admins.");  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Question 4: Validate JWT Token Expiry and Handle Unauthorized Access**

ProtectedController.cs

|  |
| --- |
| namespace AuthenticationAndAuthorization.Controllers  {  [Route("api/[controller]")]  [ApiController]  public class ProtectedController : ControllerBase  {  [HttpGet("data")]  [Authorize]  public IActionResult GetProtectedData()  {  var username = User.Identity?.Name;  return Ok($"Hello, {username}! You have accessed protected data.");  }  [HttpGet("admin-only")]  [Authorize(Roles = "Admin")]  public IActionResult GetAdminOnlyData()  {  var adminUsername = User.Identity?.Name;  return Ok($"Welcome, {adminUsername}! This is highly confidential data accessible only by Admins.");  }  }  } |

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.